ePublisher

ePublisher Platform

Documentation

Published date: 12/16/2023

webworks®

Technical Communication Software

Table of Contents

Markdown++ Source Documents

Introduction

Markdown is a text-based authoring format created by John Gruber. It's a simple, light-weight and robust language that puts emphasis on efficiency and readability.

Quick Links

Getting Started with Markdown

Learning Markdown

Learning Markdown++

Markdown Cheat Sheet

These sections will go into detail on authoring in Markdown and Markdown++, how-tos for syntax features, as well as how to prepare documents for publishing in ePublisher.

Getting Started with Markdown

Markdown is a text-based authoring format. Any text editor can be used to create Markdown documents. Here are some popular ones to try out:

- Notepad++. Simple, lightweight notepad app with syntax highlighting based on file extension.
- Visual Studio Code. Code-centric text editor. Has community-made extensions, including many for Markdown.
- Obsidian. A Markdown-specific note taking app. Has themes, a writing mode, and a preview mode.
- Typora. A Markdown editor with many authoring experience features.

After choosing a text editor, Markdown and Markdown++ documents can be created using the `.md` extension.

Learning Markdown

This section will detail the features of Markdown, how to write them, and how to use them in ePublisher. For quick reference material, see the Markdown Cheat Sheet.

Any text can be Markdown. Common prose parses into Markdown with no issue. Simple text documents can be formatted into Markdown documents quickly and easily because of this.

Quick Links

Paragraphs

The basic organization of block-level text, the paragraph is the building block of a Markdown document.

Syntax

A Paragraph is created by writing any text content on a line. It is the default block-level element, meaning all content is considered a Paragraph if the content does not have any recognizeable block-level syntax.

Basics

Any amount of text will do to create a Paragraph. Start the line with non-space characters to avoid indentation-related parsing issues.

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
 eiusmod tempor incididunt ut labore et dolore magna aliqua.
```

Separate with Empty Lines

Keep an empty line between Paragraphs that should be separated. This is a general good rule of thumb for all Markdown content.

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
 eiusmod tempor incididunt ut labore et dolore magna aliqua.


Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris
 nisi ut aliquip ex ea commodo consequat.
```

Multi-Line Paragraphs

Multiple lines not separated by an empty line will be treated as parts of the same Paragraph. The lines will be consolidated and separated by space in the output.

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
 eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris
 nisi ut aliquip ex ea commodo consequat.
```

Preserve Line Breaks

Ending a line with a space character at the end will preserve the line break within the Paragraph. Useful for poetry, or other types of content where line structure is important.

```
Nature's first green is gold,

Her hardest hue to hold.

Her early leaf's a flower;

But only so an hour.
```

Markdown++

A custom Paragraph Style can be given to a Paragraph using a Markdown++ style tag on the line directly above the Paragraph.

```
<!--style:CustomParagraph-->

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
 eiusmod tempor incididunt ut labore et dolore magna aliqua.
```

To learn more about Markdown++ tagging, see Learning Markdown++.

ePublisher Style Information

Default Style Properties

Style Type: **Paragraph**

Style Name: **Paragraph**

Property	Value
font family	Arial
font size	12pt
line height	1.2em
padding top	0pt
padding bottom	6pt

If a custom style name is assigned to a Paragraph, that style name will still inherit all of the listed default style information.

Titles

Also referred to as a *setext heading*, Titles are useful to communicate the central idea of a document. Titles are most useful as the leading content of a set of text material.

Syntax

Titles are created by writing a single line of content followed by a line containing at least 1 of either `=` or `-` characters. The second line shouldn't contain text other than these two characters.

Basics

The most basic example, a line of content with a following line with a `=` character.

```
My Document Title

=
```

Titles can be written in the same way using `-` characters.

```
My Document Title

-
```

Any Amount of Characters

The amount of `=` or `-` characters that are used is not important. Having a matching amount of characters on both lines can be a nice touch for readability, though.

```
My Document Title

=================
```

Markdown++

A custom Paragraph Style can be given to a Title using a Markdown++ style tag on the line directly above the Title.

```
<!--style:CustomTitle-->

My Document Title

=================
```

> To learn more about Markdown++ tagging, see Learning Markdown++.

ePublisher Style Information

Style Behavior

The style name a Title will get is dependent on the characters used in the second line. **Title 1** is given to Titles that use `=` characters, and **Title 2** is given to Titles that use `-` characters.

Default Style Properties

Style Type: **Paragraph**

Style Name: **Title 1**, **Title 2**

Property	Value
font family	Arial
font size	24pt
font weight	bold
line height	1.2em
padding top	0pt
padding bottom	12pt

Default Style Options

Option	Value
Table of Contents level	1

If a custom style name is assigned to a Title, that style name will still inherit all of the listed default style information.

Headings

Originally named the *ATX heading*, a Heading communicates a central idea for a topic. Headings should contain the main idea for a section, and have useful keywords to make the section easy to find in a search.

Syntax

Headings are created by starting a line of content with the `#` character. The `#` characters and the text content of the Heading need to be separated by a space character. The amount of `#` characters used indicates the level of heading which will be created.

Basics

Create a Heading 1 with a single `#`, a space, and some text.

```
# Heading 1
```

More `#` characters can be added to the Heading to increase the heading level, up to 6.

```
# Heading 1


## Heading 2


### Heading 3


#### Heading 4


##### Heading 5


###### Heading 6
```

Markdown++

A custom Paragraph Style can be given to a Heading using a Markdown++ style tag on the line directly above the Heading.

```
<!--style:CustomHeading-->
```

```
# Heading 1
```

To learn more about Markdown++ tagging, see Learning Markdown++.

Heading Behavior

Heading Alias

Each created Heading gets an alias that can be used to link to it from another place in the publication.

To determine the alias value, ePublisher takes the text of the Heading, lower-cases it, removes all non-alphanumeric characters, and replaces space with `-` characters.

The below Heading will get the alias value `lets-go-to-the-moon`.

```
# Let's Go to the Moon!
```

Any time the text of a Heading is changed, the alias will also change. It's recommended to use a Custom Alias to avoid having to change link paths when Headings change.

ePublisher Style Information

Style Behavior

The style name ePublisher will create for a Heading will is dependent on the number of `#` characters used at the front of the line. One `#` character creates the style name **Heading 1**, two `#` characters creates **Heading 2**, etc.

Default Style Properties

Style Type: **Paragraph**

Style Name: **Heading 1**, **Heading 2**, **Heading 3**, **Heading 4**, **Heading 5**, **Heading 6**

Heading 1

Property	Value
font family	Arial
font size	21pt
font weight	bold
line height	1.2em
padding top	0pt

Property	Value
padding bottom	12pt

Heading 2

Property	Value
font family	Arial
font size	18pt
font weight	bold
line height	1.2em
padding top	0pt
padding bottom	12pt

Heading 3

Property	Value
font family	Arial
font size	15pt
font weight	bold
line height	1.2em
padding top	0pt
padding bottom	12pt

Heading 4, Heading 5, Heading 6

Property	Value
font family	Arial
font size	12pt
font weight	bold
line height	1.2em
padding top	0pt
padding bottom	12pt

Default Style Options

Heading 1

Option	Value
Table of Contents level	2

Heading 2

Option	Value
Table of Contents level	3

Heading 3

Option	Value
Table of Contents level	4

Heading 4

Option	Value
Table of Contents level	5

Heading 5

Option	Value
Table of Contents level	6

Heading 6

Option	Value
Table of Contents level	none

If a custom style name is assigned to a Heading, that style name will still inherit all of the listed default style information for the matching Heading syntax.

Lists

Lists are a structural feature in Markdown. They're useful for many things, such as itemizing a collection of information, providing steps to a procedure, or numbering sections of information.

Syntax

There are two types of lists that can be created: Ordered Lists and Unordered Lists.

Ordered Lists are created by starting a line with a number or letter, followed by a single `.`, then space (two is recommended), then some text content.

Unordered Lists are created by starting a line with any `-`, `*`, or `+` character, followed by a space, then some text content.

Beyond these differences, both types of Lists have the same behavior when it comes to syntax and authoring.

Basics

Ordered List

Create a simple Ordered List using a number and a `.` character. Each list item is written on it's own line.

```
1.  list item one

2.  list item two

3.  list item three
```

Ordered Lists can also be created using letters and `.`.

```
a.  list item one

b.  list item two

c.  list item three
```

Roman numerals are fine, too. Remember to keep the vertical spacing consistent.

```
i.   list item one

ii.  list item two

iii. list item three
```

Re-using the same letter or number is OK.

```
1.  list item one

1.  list item two

1.  list item three
```

```
a.  list item one

a.  list item two

a.  list item three
```

Unordered List

Create a simple Unordered List using `-`. Each list item is written on it's own line.

```
- list item one

- list item two

- list item three
```

Unordered Lists can also be created using `*`.

```
* list item one

* list item two

* list item three
```

The `+` character can be used as well.

```
+ list item one

+ list item two

+ list item three
```

One Empty Line Between List Items

Put a single empty line between list items to give room. More than one empty line will break the list into two.

```
- list item one
```

```
- list item two


- list item three
```

Don't Use Unlike Characters

Using non-matching characters on the same list level will break the list in two.

```
- list item one

* list item one

+ list item one
```

```
1.  list item one

a.  list item one
```

Multi-Line Content in List Items

List Item content can span multiple lines. Use a blank line to separate elements. Make sure all lines of content retain the same vertical spacing.

```
1.  ### Cities in the US

    Here is a sample of some cities in the United States.

    | Name   | State    |

    |--------|----------|

    | Austin | Texas    |

    | Tulsa  | Oklahoma |


2.  list item two
```

```
- ### Cities in the US

  Here is a sample of some cities in the United States.
```

```
  | Name   | State    |
  |--------|----------|
  | Austin | Texas    |
  | Tulsa  | Oklahoma |

- list item two
```

Nested List Items

To nest List items, make sure the vertical spacing of the nested List item matches up with the content of the parent List item.

```
1.  list item one

    1.  nested list item one

2.  list item two

3.  list item three
```

```
- list item one

  - nested list item one

- list item two

- list item three
```

Nesting Different Types of Lists

Nesting Lists of different types is acceptable. Use the same spacing rules as usual.

```
1.  list item one

    - nested list item one
```

```
    - nested list item two

2.  list item two

3.  list item three
```

```
- list item one

  1.  nested list item one

  2.  nested list item two

- list item two

- list item three
```

Markdown++

A custom Paragraph Style can be given to a List using a Markdown++ style tag on the line directly above the List.

```
<!--style:CustomOList-->

1.  A customized ordered list, style name "CustomOList"

2.  CustomOList item two

3.  CustomOList item three
```

```
<!--style:CustomUList-->

- A customized unordered list, style name "CustomUList"

- CustomUList item two

- CustomUList item three
```

Customizing Nested Lists

Nested Lists can be customized as well.

```
1.  A default list, style name "OList"


    <!--style:CustomOList-->

    a.  A customized list, style name "CustomOList"


    b.  CustomOList item two


2.  OList item two
```

```
- A default list, style name "UList"


  <!--style:CustomUList-->

  - A customized list, style name "CustomUList"


  - CustomUList item two


- UList item two
```

Default Until Customized

Nested Lists are treated as standalone; they will not inherit the outermost stylename if customized.

```
<!--style:CustomOList-->

1.  A customized list, style name "CustomOList"


    a.  A default list, style name "OList"


    b.  OList item two


2.  CustomOList item two
```

```
<!--style:CustomUList-->
```

```
- A customized list, style name "CustomUList"


  - A default list, style name "UList"


  - UList item two


- CustomUList item two
```

Add a style tag to each list individually for consistency with custom styles.

```
<!--style:CustomOList-->

1.  A customized list, style name "CustomOList"


    <!--style:CustomOList-->

    a.  A customized list, style name "CustomOList"


    b.  CustomOList item two


2.  CustomOList item two
```

```
<!--style:CustomUList-->

- A customized list, style name "CustomUList"


  <!--style:CustomUList-->

  - A customized list, style name "CustomUList"


  - CustomUList item two


- CustomUList item two
```

Nested Content in Lists

The tagging convention can be used for other Markdown elements inside List items. The resulting style name will be appended with the style name of the containing List.

```
1.  <!--style:CustomParagraph-->

    A customized paragraph, style name "OList CustomParagraph"


2.  list item two
```

```
- <!--style:CustomParagraph-->

  A customized paragraph, style name "UList CustomParagraph"


- list item two
```

To learn more about Markdown++ tagging, see Learning Markdown++.

ePublisher Style Information

Style Behavior

To allow full styling of Lists, ePublisher creates a number of style names when a list is detected inside a Markdown source document.

List Style

The List Style is the first style that ePublisher adds to the Style Designer when a list is detected in a source document. The default name is **OList** for ordered lists, and **UList** for unordered lists, but could also be a custom name if the style tag syntax is used on the list.

This style applies to the container area surrounding the lists's items. It's style rules can also apply to list items or nested content, if the same rule isn't already applied on a nested style.

Customizing the List Style

By adding a Markdown++ custom style tag, the List Style name can be changed. The example below changes the List Style name to **CustomUList**:

```
<!--style:CustomUList-->

- This is a custom list

- unordered

- named "CustomUList"
```

List Item Style

The list items inside a list also get a style name. To determine the List Item Style's name, ePublisher takes the List Style and adds `Item` to the end, separated by a space. The default name is **OList Item** for ordered list items, and **UList Item** for unordered list items, but could also be a custom name if the style tag syntax is used on the list.

Customizing the List Item Style

By adding a Markdown++ custom style tag, the List Item Style name can be changed. The example below adds the List Item Style **CustomUList Item**, because the List Style name has been set to **CustomUList**:

```
<!--style:CustomUList-->

- This is a custom list

- unordered

- named "CustomUList"
```

List items can't be styled individually. This will break the list into two separate lists. All list items in a given list are styled by the same List Item Style.

Nested Styles

Nested content inside of list items also get a new style name. To determine the Nested Style's name, take the List Style and add the style name of the nested content to the end, separated by a space.

The example below populates the Style Designer with 3 Paragraph Styles: **UList** (the List Style), **UList Item** (the List Item Style), and **UList Paragraph** when scanned into ePublisher.

```
- This is a simple list

- unordered

- default style names
```

Customizing Nested Styles

By adding a Markdown++ custom style tag, the Nested Style name can be changed. The example below changes the Nested Style name to **UList CustomParagraph**:

```
- <!--style:CustomParagraph-->

  This is a custom paragraph.
```

Custom Style Names can be used on both the list and nested content simultaneously. This example creates the style names **CustomUList**, **CustomUList Item**, and **CustomUList CustomParagraph**:

```
<!--style:CustomUList-->

- <!--style:CustomParagraph-->

  This is a custom paragraph inside a blockquote.
```

Default Style Properties

Style Type: **Paragraph**

Style Name: **OList**, **OList Item**, **UList**, **UList Item**

OList

Property	Value
padding top	0pt
padding right	0pt
padding bottom	0pt
padding left	0pt
margin top	0pt
margin right	0pt
margin bottom	0pt
margin left	0pt
tag	ol

UList

Property	Value
padding top	0pt
padding right	0pt
padding bottom	0pt

Property	Value
padding left	0pt
margin top	0pt
margin right	0pt
margin bottom	0pt
margin left	0pt
tag	ul

OList Item, UList Item

Property	Value
padding top	0pt
padding right	0pt
padding bottom	0pt
padding left	0pt
margin top	0pt
margin right	0pt
margin bottom	0pt
margin left	36pt
tag	li

If a custom style name is assigned to a List, that style name will still inherit all of the listed default style information.

Tables

Tables lay out multiple lines of detailed data in an organized way. In Markdown, Tables are used to display cells of inline content. This often means that table structue is kept simple.

If a Table with complex structure is needed, it can be created as an HTML fragment in a Block HTML element.

Syntax

Markdown Tables consist of 3 things:

- A **header row**, which contains header cell content separated by `|` characters.
- An **alignment row**, that indicates the alignment of the body cells' text. Each cell in this row contains at least 3 `-` characters, and an optional `:` character to indicate alignment. Each cell is separated by a `|` character.
    - Default alignment only uses `-` characters; 3 or more.
    - Left align the column by starting the cell with `:` and filling in the rest with `:` characters; 3 or more.
    - Right align the column by starting the cell with 3 or more `-` characters, ending with a `:` character.
    - Center align the column by starting and ending the cell with `:` characters. Put `-` characters between them; 3 or more.
- 1 or more **body rows**, that contain body cell content separated by `|` characters.

Each row's content should be confined to a single line. The table will not parse properly if rows have multi-line content.

Optionally, all lines in the table can start and end with `|` characters. Be sure to apply them to all lines if they are to be used.

Basics

Two basic Tables; one with wrapping `|` characters, one without.

```
| name | age | city |

|---|---|---|

| Bob | 42 | Dallas |

| Mary | 37 | El Paso |
```

```
 name | age | city
```

```
---|---|---
 Bob | 42 | Dallas
 Mary | 37 | El Paso
```

Line up the `|` characters in each row for a nice touch for readibilty.

```
| name | age | city    |
|------|-----|---------|
| Bob  | 42  | Dallas  |
| Mary | 37  | El Paso |
```

```
 name | age | city
------|-----|------
 Bob  | 42  | Dallas
 Mary | 37  | El Paso
```

Left-align the text of cells in a column by starting the alignment cell with `:`. The first column is left-aligned in this example:

```
| name | age | city    |
|:-----|-----|---------|
| Bob  | 42  | Dallas  |
| Mary | 37  | El Paso |
```

```
 name | age | city
:-----|-----|------
 Bob  | 42  | Dallas
 Mary | 37  | El Paso
```

Right-align the text of cells in a column by ending the alignment cell with `:`. The first column is right-aligned in this example:

```
| name | age | city    |
|-----:|-----|---------|
| Bob  | 42  | Dallas  |
| Mary | 37  | El Paso |
```

```
 name | age | city
-----:|-----|------
 Bob  | 42  | Dallas
 Mary | 37  | El Paso
```

Center-align the text of cells in a column by starting and ending the alignment cell with `:`. The first column is center-aligned in this example:

```
| name | age | city    |
|:----:|-----|---------|
| Bob  | 42  | Dallas  |
| Mary | 37  | El Paso |
```

```
 name | age | city
:----:|-----|------
 Bob  | 42  | Dallas
 Mary | 37  | El Paso
```

Each column gets it's own alignment. Mix them together as needed.

```
| name | age | city    |
|:-----|:---:|--------:|
| Bob  | 42  | Dallas  |
| Mary | 37  | El Paso |
```

```
 name | age | city

:-----|:---:|--------:

 Bob  | 42  | Dallas

 Mary | 37  | El Paso
```

Markdown In Tables

Inline Markdown elements, like bold, italic, and even inline HTML, can be used with cell text content.

```
| name     | age | city    |

|----------|-----|---------|

| **Bob**  | 42  | Dallas  |

| **Mary** | 37  | El Paso |
```

```
 name     | age | city

----------|-----|---------

 **Bob**  | 42  | Dallas

 **Mary** | 37  | El Paso
```

Markdown++

A custom Table Style can be given to a Table using a Markdown++ style tag on the line directly above the Table.

```
<!--style:CustomTable-->

| name | age | city    |

|------|-----|---------|

| Bob  | 42  | Dallas  |

| Mary | 37  | El Paso |
```

```
<!--style:CustomTable-->
```

```
 name | age | city

------|-----|------

 Bob  | 42  | Dallas

 Mary | 37  | El Paso
```

Content in Cells

Inline text content can be customized using the inline tag convention.

```
| name                           | age | city     |

|--------------------------------|-----|----------|

| <!--style:CustomText-->*Bob*   | 42  | Dallas   |

| <!--style:CustomText-->*Mary*  | 37  | El Paso  |
```

```
 name                           | age | city

--------------------------------|-----|------

 <!--style:CustomText-->*Bob*   | 42  | Dallas

 <!--style:CustomText-->*Mary*  | 37  | El Paso
```

To learn more about Markdown++ tagging, see Learning Markdown++.

ePublisher Style Information

Style Behavior

In order to style a Table and it's cells in detail, a few different styles are needed in ePublisher. A Table gets 3 styles when ePublisher detects one in a document.

The example below populates the Style Designer with 1 Table Style called **Table**, and 2 Paragraph Styles: **Table Cell Head**, and **Table Cell Body** when scanned into ePublisher.

```
> # Heading 1 element inside a blockquote

>

> This is a Paragraph element inside of a blockquote.
```

```
>
```

Table Style

The Table Style is the first style that ePublisher adds to the Style Designer when a table is detected in a source document. The default name is **Table**, but could also be a custom name if the style tag syntax is used on the table.

This style applies table-specific style rules to the entire table. It is the only style in Markdown++ that creates an entry in the **Table Styles** area in the Style Designer.

Customizing the Table Style

By adding a Markdown++ custom style tag, the Table Style name can be changed. The examples below change the Table Style name to **CustomTable**:

```
<!--style:CustomTable-->

| name | age | city    |

|------|-----|---------|

| Bob  | 42  | Dallas  |

| Mary | 37  | El Paso |
```

```
<!--style:CustomTable-->

 name | age | city

------|-----|------

 Bob  | 42  | Dallas

 Mary | 37  | El Paso
```

Header & Body Cell Styles

Every cell on a header row gets a Header Cell Style. Each cell on body rows get a Body Cell Style as well. To determine the Header Cell Style's name, ePublisher takes the Table Style and adds `Cell Head` to the end for Header Cell Styles, and `Cell Body` to the end for Body Cell Styles. The default names are **Table Cell Head** and **Table Cell Body**, but these will also be customized if the Table Style has a custom name.

Customizing Header & Body Cell Styles

By adding a Markdown++ custom style tag, the Header & Body Cell Style names can be changed. The example below changes the style names to **CustomTable Cell Head** and **CustomTable Cell Body** because the Table Style has been given the custom style name **CustomTable**:

```
<!--style:CustomTable-->

| name | age | city    |

|------|-----|---------|

| Bob  | 42  | Dallas  |

| Mary | 37  | El Paso |
```

```
<!--style:CustomTable-->

 name | age | city

------|-----|------

 Bob  | 42  | Dallas

 Mary | 37  | El Paso
```

Default Style Properties

Style Type: **Table**, **Paragraph**

Style Name: **Table**, **Table Cell Head**, **Table Cell Body**

Table

Property	Value
border top color	#222222
border top style	solid
border top width	1px
border right color	#222222
border right style	solid
border right width	1px
border bottom color	#222222
border bottom style	solid

Property	Value
border bottom width	1px
border left color	#222222
border left style	solid
border left width	1px

Table Cell Head

Property	Value
font family	Arial
font size	11pt
font weight	bold
padding top	6pt
padding right	6pt
padding bottom	6pt
padding left	6pt

Table Cell Body

Property	Value
font family	Arial
font size	11pt
padding top	6pt
padding right	6pt
padding bottom	6pt
padding left	6pt

If a custom style name is assigned to a Table, the style names will still inherit all of the listed default style information.

Blockquotes

Blockquotes are unique block-level elements that can contain other block-level elements. They have a diverse set of usages due to this, such as capturing a sequence of conversation, or being a container for an important presentation of concepts.

Syntax

Blockquotes are created by starting a line with the `>` character. A space between text content and the `>` character is optional, but recommended. Any Markdown or Markdown++ convention is acceptable as text content inside of Blockquotes.

Basics

A basic Blockquote containing a single Paragraph

```
> A paragraph inside a blockquote.
```

Markdown In Blockquotes

Other Markdown elements, like headings and lists, can be used inside Blockquotes. Use the same spacing and indentation rules as usual when inside Blockquotes.

```
> ### How to Publish Content with ePublisher

>

> Here's some steps to publish your content with ePublisher.

>

> 1. Open ePublisher

> 2. Add source documents

> 3. Select Format

> 4. Click **Generate All**
```

Nested Blockquotes

Other Blockquotes can also be nested inside of Blockquotes, and so on.

```
> First level blockquote

>
```

```
> > Second level nested blockquote.

> >

> > > Third level nested blockquote.

> > >
```

Markdown++

A custom Paragraph Style can be given to a Blockquote using a Markdown++ style tag on the line directly above the Blockquote.

```
<!--style:CustomBlockquote-->

> A customized blockquote, style name "CustomBlockquote"
```

Customizing Nested Blockquotes

Nested Blockquotes can be customized as well.

```
> A default blockquote, style name "Blockquote"

>

> <!--style:CustomBlockquote-->

> > A customized blockquote, style name "CustomBlockquote"

> >
```

Default Until Customized

Nested Blockquotes are treated as standalone; they will not inherit the outermost stylename if customized.

```
<!--style:CustomBlockquote-->

> A customized blockquote, style name "CustomBlockquote"

>

> > A default blockquote, style name "Blockquote"

> >
```

Add a style tag to each blockquote individually for consistency with custom styles.

```
<!--style:CustomBlockquote-->

> A customized blockquote, style name "CustomBlockquote"

>

> <!--style:CustomBlockquote-->

> > A customized blockquote, style name "CustomBlockquote"

> >
```

Markdown in Blockquotes

The tagging convention can be used for other Markdown elements inside Blockquotes. These style names will inherit the Blockquote's style name as a prefix. See Nested Styles for more info.

```
> <!--style:CustomParagraph-->

> A customized paragraph, style name "Blockquote CustomParagraph"

>

> <!--style:CustomList-->

> - an unordered list

> - customized

> - style name "Blockquote CustomList"

>
```

To learn more about Markdown++ tagging, see Learning Markdown++.

ePublisher Style Information

Style Behavior

Blockquotes are considered containers; they *contain* other block-level elements, like Paragraphs, Lists, and Tables. Because of this, ePublisher creates a number of different styles when it detects blockquotes in source documents.

Blockquote Style

The Blockquote Style is the first style that ePublisher adds to the Style Designer when a blockquote is detected in a source document. The default name is **Blockquote**, but could also be a custom name if the style tag syntax is used on the blockquote.

This style applies to the container area surrounding the blockquote's content. It's style rules can also apply to nested content, if the same rule isn't already applied on the nested style.

Customizing the Blockquote Style

By adding a Markdown++ custom style tag, the Blockquote Style name can be changed. The example below changes the Blockquote Style name to **CustomBlockquote**:

```
<!--style:CustomBlockquote-->

> This is a custom named blockquote.

>
```

Nested Styles

Nested content inside of Blockquotes also get a new style name. To determine the Nested Style's name, take the Blockquote Style and add the style name of the nested content to the end, separated by a space.

The example below populates the Style Designer with 3 Paragraph Styles: **Blockquote** (the Blockquote Style), **Blockquote Heading 1**, and **Blockquote Paragraph** when scanned into ePublisher.

```
> # Heading 1 element inside a blockquote

>

> This is a Paragraph element inside of a blockquote.

>
```

Customizing Nested Styles

By adding a Markdown++ custom style tag, the Nested Style name can be changed. The example below changes the Nested Style name to **Blockquote CustomParagraph**:

```
> <!--style:CustomParagraph-->
```

```
> This is a custom paragraph.

>
```

Custom Style Names can be used on both the blockquote and nested content simultaneously. This example creates the style names **CustomBQ**, and **CustomBQ CustomParagraph**:

```
<!--style:CustomBQ-->

> <!--style:CustomParagraph-->

> This is a custom paragraph inside a blockquote.

>
```

Default Style Properties

Style Type: **Paragraph**

Style Name: **Blockquote**

Property	**Value**
background color	#efefef
border left style	solid
border left color	#DFE2E5
border left width	3pt
padding top	12pt
padding right	12pt
padding bottom	12pt
padding left	12pt

If a custom style name is assigned to a Blockquote, that style name will still inherit all of the listed default style information.

Code Fences

A Code Fence preserves all text content it encapsulates and presents it exactly how it was written. Code Fences are useful for presenting information that needs to be written explicitly, like examples of code. They can provide users with content that can be copied and used for their own purposes, too.

Syntax

Code Fences are created in three steps:

1. Start with a line containing ```` ``` ```` or `~~~`.
2. A following line or lines of text content.
3. End with a line containing ```` ``` ```` or `~~~`, matching the starting line.

Basics

A simple example using ```` ``` ```` tags. Any amount of text can be written between the two tags, as long as the tags match and are written correctly.

````
```

function addTwoNumbers(num1, num2) {

  return num1 + num2;

}

```
````

Code Fences can be created using `~~~`, too.

```
~~~

function addTwoNumbers(num1, num2) {

  return num1 + num2;

}

~~~
```

No Parsing in Code Fences

Markdown written inside of Code Fences will render as plain text.

```
# Heading 1 in Plain Text
```

HTML will also render as plain text when written inside Code Fences.

```
<p>HTML in plain text</p>
```

Markdown++

A custom Paragraph Style can be given to a Code Fence using a Markdown++ style tag on the line directly above the Code Fence.

```
<!--style:CustomCodeFence-->
```

```
function addTwoNumbers(num1, num2) {
  return num1 + num2;
}
```

To learn more about Markdown++ tagging, see Learning Markdown++.

ePublisher Style Information

Default Style Properties

Style Type: **Paragraph**

Default Style Name: **Code Fence**

Property	Value
background color	#efefef
font family	Consolas

Property	Value
font size	11pt
margin top	6pt
margin bottom	6pt
padding top	12pt
padding right	12pt
padding bottom	12pt
padding left	12pt
overflow	auto
white space	pre

If a custom style name is assigned to a Code Fence, that style name will still inherit all of the listed default style information.

Code Blocks

A Code Block preserves all text content it encapsulates and presents it exactly how it was written. Code Blocks are useful for presenting information that needs to be written explicitly, like examples of code. They can provide users with content that can be copied and used for their own purposes, too.

Syntax

Code Blocks are created by adding 4 spaces before text content. A Code Block can consist of one or more lines created in this manner.

Basics

Starting a line with 4 spaces will create a basic Code Block.

```
    var firstName, lastName;
```

Multi-Line Code Blocks

Multiple lines can be used; start all lines with at least 4 spaces.

```
    var firstName, lastName;



    firstName = "John";

    lastName = "Doe";
```

Space is Preserved.

Any spaces after the first 4 will be used as indentation for the content of the Code Block.

```
    function addTwoNumbers(num1, num2) {

      return num1 + num2;

    }
```

No Parsing in Code Blocks

Markdown written inside of Code Blocks will render as plain text.

```
    # Heading 1 in Plain Text
```

HTML will also render as plain text when written inside Code Blocks.

```
    <p>HTML in plain text</p>
```

Markdown++

A custom Paragraph Style can be given to a Code Block using a Markdown++ style tag on the line directly above the Code Block.

```
<!--style:CustomCodeBlock-->

    function addTwoNumbers(num1, num2) {

      return num1 + num2;

    }
```

To learn more about Markdown++ tagging, see Learning Markdown++.

ePublisher Style Information

Default Style Properties

Style Type: **Paragraph**

Style Name: **Code Block**

Property	Value
background color	#efefef
font family	Consolas
font size	11pt
margin top	6pt
margin bottom	6pt
padding top	12pt
padding right	12pt
padding bottom	12pt
padding left	12pt
overflow	auto
white space	pre

If a custom style name is assigned to a Code Block, that style name will still inherit all of the listed default style information.

Horizontal Rules

A Horizontal Rule provides a visual separation between sections of content. They're useful to separate unrelated ideas on a single page.

Syntax

A Horizontal Rule is created by using at least 3 `-`, `_`, or `*` characters. These should be the only characters on the line, but any combination of them is acceptable.

Basics

A simple Horizontal Rule using `-` characters.

```
---
```

An example using `*` characters.

```
***
```

And one with `_` characters.

```
___
```

3 or More Characters

More than 3 characters can be used, if desired.

```
----------
```

Spaces OK

Spaces are acceptable between the characters.

```
- - - - - - -
```

Mixed Characters

Combinations of the 3 characters is fine to use as well.

```
-*_*-*_*-*_*-
```

```
-_-_-_-_-_-_-
```

```
* - * - * - *
```

Markdown++

A custom Paragraph Style can be given to a Horizontal Rule using a Markdown++ style tag on the line directly above the Horizontal Rule.

```
<!--style:CustomHR-->

---
```

To learn more about Markdown++ tagging, see Learning Markdown++.

ePublisher Style Information

Default Style Properties

Style Type: **Paragraph**

Style Name: **Horizontal Rule**

Property	Value
border top color	#222222
border top style	inset
border top width	1px
border right color	#222222
border right style	inset
border right width	1px
border bottom color	#222222
border bottom style	inset
border bottom width	1px
border left color	#222222
border left style	inset
border left width	1px
display	block
margin top	6pt
margin bottom	6pt
tag	hr

If a custom style name is assigned to a Horizontal Rule, that style name will still inherit all of the listed default style information.

Block HTML

The common markup language for web technology, HTML, can be used in Markdown documents on the block level. Refer to W3Schools' HTML Tutorial to learn more about how to write and use HTML.

Syntax

Block HTML is created by writing a valid HTML fragment on a line or set of lines. HTML syntax must be the first thing on the line to be considered Block HTML.

Basics

Simple Block HTML using a `p` element.

```
<p>A simple paragraph element.</p>
```

Multi-Line HTML

HTML can span multiple lines. Keep it compact. An empty line will break the fragment in two, so it is best used to separate the fragment from other content.

```
<table>

  <tr>

    <th>Name</th>

    <th>Age</th>

    <th>Country</th>

  </tr>

  <tr>

    <td>John Doe</td>

    <td>35</td>

    <td>USA</td>

  </tr>

  <tr>

    <td>Jane Doe</td>
```

```
    <td>32</td>
    <td>USA</td>
  </tr>
</table>
```

No Markdown in Block HTML

Markdown syntax can't be used inside of Block HTML. The entire HTML fragment is passed straight to the output as-is.

```
<p>No **Markdown** here.</p>
```

Markdown++

A custom Paragraph Style can be given to Block HTML using a Markdown++ style tag on the line directly above the Block HTML.

```
<!--style:CustomHTML-->

<p>HTML block given the style name "CustomHTML"</p>
```

To learn more about Markdown++ tagging, see Learning Markdown++.

ePublisher Style Information

Style Behavior

All HTML fragments are wrapped in a container element, which is given a style name. The default name is **HTML**, but can also be a custom name if the style tag is used directly above an HTML fragment.

HTML is unavailable for publishing in PDF or PDF XSL-FO output due to incompatibility with those technologies. ePublisher will remove any HTML content it detects before generating PDF output.

Default Style Properties

Style Type: **Paragraph**

Style Name: **HTML**

Property	Value
display	block

Property	Value
overflow	auto

If a custom style name is assigned to a Block HTML, that style name will still inherit all of the listed default style information.

Bold, Italic, Strikethrough, Code

Inline text can be styled to put emphasis or formatting on certain phrases. Markdown offers wrappers for Bold, Italic, Strikethrough, and Code.

Syntax

Bold text is created by wrapping a set of text with a pair of either `**` or `__` characters.

Italic text is created by wrapping text with a pair of either `*` or `_`.

Strikethrough text is created by wrapping text between a pair of `~~` characters.

Code spans are created by wrapping text between a pair of `` ` `` characters.

Basics

Two simple examples for Bold text. Notice either `*` or `_` can be used, but there must be two on each side of the wrap. The start and end characters must also match.

```
Here's **bold** and here's also __bold__.
```

Italic text is written similarly, using one `*` or `_` instead of two.

```
Here's *italic* and here's also _italic_.
```

Same rules apply to Strikethrough text, using `~~`.

```
Using ~~strikethrough~~ text.
```

Code spans follow the same rules, too.

```
Defining a `technical term`.
```

Mixing Styles of Text

Combinations of these can be used together. Make sure the innermost pair of tags is closed before closing an outer pair. Using unlike characters for different pairs helps with readability. (Using `*` for bold, `_` for italic, etc.)

```
We can write **bold and _italic_**.
```

Spanning Multiple Lines

Inline text decorators can span multiple lines, as long as there are no empty lines between the start and end tags.

```
Writing a sentence that **has

bold text** across lines.
```

Markdown++

A custom Character Style can be given to Inline Text using a Markdown++ style tag directly before the start tag of the Inline Text.

```
Styling <!--style:CustomBold-->**inline text. Style name
 "CustomBold"**.
```

To learn more about Markdown++ tagging, see Learning Markdown++.

ePublisher Style Information

Default Style Properties

Style Type: **Character**

Style Name: **Bold**, **Italic**, **Strikethrough**, **Code**

Bold

Property	Value
font weight	bold

Italic

Property	Value
font style	italic

Strikethrough

Property	Value
text decoration	line-through

Code

Property	Value
background color	#efefef
font family	Consolas
white space	pre

If a custom style name is assigned to Inline HTML, that style name will still inherit all of the listed default style information.

Links

Links are an inline Markdown convention used to connect users to other locations and resources in a set of information.

Syntax

Link syntax looks interesting, but is simple enough once written a few times. Write the link's displayed text in between `[` and `]` characters, and directly next to it write the link's URL between `(` and `)` characters. Optionally, a title can be given to the Link, written next to the link URL, separated by a space and wrapped in `"` characters.

Basics

A basic Link example.

```
[Link Text](path/to/my_doc.md)
```

Titles are optional. Keep the URL and title separate with a space. Wrap the title in `"` characters.

```
[Link Text](path/to/my_doc.md "Link Title")
```

Links can be the only thing on a line or mixed in anywhere inline text can go.

```
To see more, follow the [Link](path/to/my_doc.md).
```

Relative paths, absolute paths, web links, and Aliases are all valid path values.

```
[Link Text](../my_doc.md)

[Link Text](D:/Markdown/Docs/my_doc.md)

[Link Text](https://www.webworks.com)

[Link Text](#my-doc)
```

Using Link References

Links can make use of Link References to simplify URL management for documents with many different link paths.

```
[Link Text][0]


[0]: my_image.png
```

Titles are also available and written the same way using Link References.

```
[Link Text][0]


[0]: my_image.png "Link Title"
```

Markdown++

A custom Character Style can be given to an Image using a Markdown++ style tag immediately before the Link syntax.

```
<!--style:CustomLink-->[Link Text](path/to/my_doc.md)
```

To learn more about Markdown++ tagging, see Learning Markdown++.

Link Behavior

Links in ePublisher have a variety of ways to connect to other resources in a publication. What they connect to depends on what is used as a path value. All path values inside links also apply to path values in Link References.

Web Links

Write a fully qualified web URL in the path area to link to an external resource. Make sure to start the URL with `http://` or `https://`.

```
[WebWorks Website](https://www.webworks.com)
```

Link to Other Documents

Write the file path for the intended file in the path area to link to another document. Relative paths need to resolve from the file the link is written in. Absolute paths can also be used.

```
[link text](path/to/my_doc.md)
```

```
[link text](C:/Users/me/path/to/my_doc.md)
```

Link to Topics in Other Documents

To link to a specific section in a document, write the file path followed immediately with the alias for the topic. This can be either a Heading Alias or a Custom Alias. Relative paths need to resolve from the file the link is written in. Absolute paths can also be used.

The examples link to the alias `#my-alias` in `my_doc.md`.

```
[link text](path/to/my_doc.md#my-alias)
```

```
[link text](C:/Users/me/path/to/my_doc.md#my-alias)
```

Link to Topics in Same Document

Use an alias by itself to link to a topic in the current document. This can be either a Heading Alias or a Custom Alias.

The example links to the alias `#my-alias` in the current document.

```
[link text](#my-alias)
```

ePublisher Style Information

Default Style Properties

Style Type: **Character**

Style Name: **Link**

Property	Value
text decoration	underline
color	#0078d7

If a custom style name is assigned to a Link, that style name will still inherit all of the listed default style information.

Images

Images are an inline Markdown convention used to display graphics in a document.

Syntax

The image syntax is identical to the Link syntax, with the addition of a `!` character at the beginning. Alt text is written between `![` and `]` characters. Inside of `(` and `)`, the URL path to the image should be written, then, optionally, a title wrapped in `"` characters.

Basics

A basic Image example.

```
![alt text](path/to/my_image.png)
```

Titles are optional. Keep the URL and title separate with a space.

```
![alt text](path/to/my_image.png "Image Title")
```

Images can be the only thing on a line or mixed in anywhere inline text can go.

```
Images can go anywhere text can: ![alt text](path/to/my_image.png)
```

Relative paths and absolute paths can both be used.

```
![alt text](../my_image.png)


![alt text](D:/Images/my_image.png)
```

Using Link References

Images can also make use of [Link References][md-link-reference] in the same way Links do.

```
![alt text][0]


[0]: my_image.png
```

Titles are also available and written the same way using Link References.

```
![alt text][0]
```

```
[0]: my_image.png "Image Title"
```

Markdown++

A custom Graphic Style can be given to an Image using a Markdown++ style tag immediately before the Image syntax.

```
<!--style:CustomImage-->![alt text](path/to/my_image.png)
```

To learn more about Markdown++ tagging, see Learning Markdown++.

ePublisher Style Information

Default Style Properties

Style Type: **Graphic**

Style Name: **Image**

If a custom style name is assigned to an Image, that style name will still inherit all of the listed default style information.

Link References

Link References accompany Links and Images, and are used to keep path values in a separated location from text content. They're useful for readability because they simplify links and images in inline text, and can be written in a standalone location for editing en masse.

This section requires familiarity with Links and Images to teach referencing concepts.

Syntax

A Link Reference must be the only thing on a line. The **link key** is written between `[` and `]:`. The URL path is written next, separated by a space. Optionally, a title can be written after the URL, separated by a space.

Once a Link Reference has been written, the **link key** from it can be used with a Link or Image. Replace the Link/Image's parenthesis `()` section with the link key between `[` and `]`.

Basics

A Link Reference example used with an accompanying Link. The Link is written first and makes use of the link key, in this example `0`.

```
[Link Text][0]


[0]: path/to/my_doc.md
```

Titles are optional. Keep the URL and title separate with a space.

```
[Link Text][0]


[0]: path/to/my_doc.md "Link Title"
```

A Link Reference example with an accompanying Image.

```
![alt text][0]


[0]: path/to/my_image.png
```

Make sure to keep the Reference on it's own line. The Link or Image can be used anywhere text is allowed, though.

```
For more info, check the [Link][0].


[0]: my_doc.md
```

Use Unique Values for Link Keys

Any text will work for the link key, but something unique that can be searched for will help in the authoring process. Link keys must be one-of-a-kind as well. In the case of overlapping link keys, the last link key written will be the accepted one.

```
[wwdoc_0001]: my_doc.md

[wwdoc_0002]: doc2.md

[wwdoc_0003]: doc3.md


[wwimg_0001]: img1.png

[wwimg_0002]: img2.png

[wwimg_0003]: img3.png
```

Inline HTML

The common markup language for web technology, HTML, can be used in Markdown documents mixed with text and other inline elements. Refer to W3Schools' HTML Tutorial to learn more about how to write and use HTML.

Syntax

Inline HTML is created by writing a valid HTML fragment in an area where other inline content exists.

Basics

Simple Inline HTML using a `strong` element.

```
Write words with <strong>bold</strong> emphasis.
```

Markdown and Inline HTML

Markdown syntax can be mixed with Inline HTML.

```
We can <span>write **bold** text</span>.
```

Markdown++

A custom Character Style can be given to Inline HTML using a Markdown++ style tag directly before the Inline HTML.

```
Styling <!--style:CustomHTML--><span>inline HTML. Style name
 "CustomHTML".</span>
```

To learn more about Markdown++ tagging, see Learning Markdown++.

ePublisher Style Information

Style Behavior

All HTML fragments are wrapped in a container element, which is given a style name. The default name is **HTML**, but can also be a custom name if the style tag is used directly before an HTML fragment.

HTML is unavailable for publishing in PDF or PDF XSL-FO output due to incompatibility with those technologies. ePublisher will remove any HTML content it detects before generating PDF output.

Default Style Properties

Style Type: **Character**

Style Name: **HTML**

If a custom style name is assigned to Inline HTML, that style name will still inherit all of the listed default style information.

Learning Markdown++

This section will detail the features of Markdown++, how to write them, and how to use them in ePublisher. For quick reference material, see the Markdown Cheat Sheet.

Markdown++ is a superset of Markdown, meaning that any convention available in Markdown also applies to Markdown++. Learn about Markdown before reading this section.

Quick Links

Markdown++ Basics

Custom Styles

Aliases

Markers

Conditional Text

File Includes

Variables

Markdown++ Basics

Markdown++ is a superset of Markdown. Because of this, *all Markdown files are also Markdown++ files*. Any tools used for Markdown also work well for Markdown++.

Filling out Markdown with a full designing & publishing experience, while also maintaining readability, is a major design goal of Markdown++. Another goal is to preserve the integrity of rendering and previews across the many Markdown tools out there.

Markdown++ uses the HTML Comment tag with a set of commands inside them for most of it's features. Using these enables Markdown++ syntax to be transparent when documents are rendered or previewed, and aids in quick learning by using a well-established pattern.

Learning the HTML Comment tag opens the door to learning most of the features Markdown++ offers.

Syntax

Write an HTML Comment by starting with `<!--` and ending with `-->`. Any text can be written between these two patterns. Keeping the entire comment on a single line is required by Markdown++.

Any unrecognized text inside of a comment gets treated as a regular HTML comment and carries through to the output.

Basics

A simple comment tag with a Custom Style Name command. Keep the tag on a single line.

```
<!--style:CustomStyle-->
```

Apply commands to block-level elements by adding the tag to the line directly above the block element. The style **CustomParagraph** is added to a paragraph below.

```
<!-- style:CustomParagraph -->

A customized paragraph, named "CustomParagraph".
```

Apply commands to inline elements by adding the tag directly before the inline syntax. Don't put space between the comment tag and the inline syntax. The style **CustomBold** is added to bold text below.

```
Customizing some <!--style:CustomBold-->**bold text**.
```

Whitespace OK

In general, it is safe to include any amount of whitespace *between the comment tags*. Use it as necessary for readability.

```
<!-- style: CustomStyle -->
```

Multiple Commands

Any number of commands can be put inside the comment. Separate the commands with a `;` character. This example applies two commands to a Heading 1: a Custom Style Name, and a Custom Alias.

```
<!-- style:CustomHeading1 ; #custom-heading1 -->

# Heading 1
```

Start & End Tags

Some features, like Conditional Text, require start & end tags. The example wraps condition tags around content meant only for printed publications.

```
<!--condition:print_only-->

## Print Only


This ccontent is meant for print only.

<!--/condition-->
```

Custom Styles

The Custom Style command overrides the default Style Name of a Markdown element with a user-defined Style Name. Using this feature enables a virtually limitless amount of styles for designing & publishing in ePublisher.

Syntax

The Custom Style command is created by writing `style:` followed by the name of the intended style.

Basics

A basic Custom Style command applied to a Paragraph.

```
<!--style:CustomParagraph-->

This paragraph has it's style name customized to "CustomParagraph".
```

Put the tag on the line above any block-level element to customize it. Make sure there are no empty lines between the tag and the block element. The tag needs to be the only thing on it's line.

```
<!--style:CustomHeading1-->

# This Heading has been renamed to "CustomHeading1".
```

For Custom Styles on inline text content, put the tag directly before the starting syntax. No space should be put between the tag and the inline syntax. A string of bold text is customized with the Style Namme **CustomBold** below.

```
This paragraph has customized <!--style:CustomBold-->**bold text**.
```

Mix with Other Commands

Custom Styles can be in the same comment tag with other commands. Separate them with a `;` character. A Custom Style and Custom Alias are written in the same tag below.

```
<!-- style:CustomStyle ; #custom-alias -->
```

Custom Style Command Behavior

Through the Custom Style command, it is possible to get name entries for almost any type of style into ePublisher's Style Designer. How to do so varies based on style type.

Custom Paragraph Style

Add the style tag to the line directly above a block-level element to give it a custom Paragraph Style. This applies to any block-level element, except for Tables.

```
<!--style:CustomParagraph-->

This is a custom paragraph called "CustomParagraph".
```

```
<!--style:CustomHeading1-->

# This is a custom paragraph called "CustomParagraph".
```

```
<!--style:UList-->

- custom list

- unordered

- called "CustomUList"
```

```
<!--styleCustomBlockquote-->

> This is a custom blockquote

>
```

```
<!--style:CustomHTML-->

<div>

  <p>This is customized HTML. Named "CustomHTML".</p>

</div>
```

Custom Character Style

Add a style tag directly before inline syntax to create a custom Character Style. Remember, no space between the tag and the inline syntax. This applies to any inline syntax, except for Images.

```
This is customized <!--style:CustomBold-->**bold text**.
```

```
This is customized <!--style:CustomItalic-->*italic text*.
```

```
This is a customized <!--style:CustomLink-->[link](my_doc.md).
```

```
This is a customized <!--style:CustomHTML--><span>Inline HTML</span>.
```

Custom Graphic Style

Add a style tag directly before image syntax to create a custom Graphic Style. Remember, no space between the tag and the image syntax.

```
<!--style:CustomImage-->![alt text](my_image.png)
```

```
<!--style:CustomImage-->![alt text][link_key]
```

Custom Table Style

Like custom Paragraph Styles, add the tag to the line directly above Table syntax to give it a custom Table Style.

```
<!--style:CustomTable-->

| name | age | city    |

|------|-----|---------|

| Bob  | 42  | Dallas  |

| Mary | 37  | El Paso |
```

```
<!--style:CustomTable-->

 name | age | city
```

```
------|-----|------

 Bob   | 42  | Dallas

 Mary  | 37  | El Paso
```

Custom Page Style

Custom Page Styles need to be created through the Custom Markers command. Refer to the linked section for details.

```
<!--markers:{"PageStyle": "CustomPage"}-->

# Topic Heading
```

Custom Marker Style

Custom Marker Styles need to be created through the Custom Markers command. Refer to the linked section for details.

```
<!--markers:{"Keywords": "topic, heading, markers"}-->

# Topic Heading
```

Custom Aliases

Use a Custom Alias to give an element a unique pointer for linking to in other places in the publication. The Custom Alias is a powerful tool that can simplify link management and speed up authoring and editing.

Syntax

Create a Custom Alias by starting with a single `#` character, followed any alphanumeric characters, `-`, or `_`. Space characters cannot be used in an alias; the Alias will cut off before the space.

Basics

A basic Custom Alias applied to a Heading 1.

```
<!--#custom-alias-->

# Custom Aliased Heading
```

Inline syntax can be given an Alias as well.

```
This <!--#bold-keyword-->**bold text** has a custom alias.
```

Mix with Other Commands

Custom Aliases can be in the same comment tag with other commands. Separate them with a `;` character. A Custom Style and Custom Alias are written in the same tag below.

```
<!-- style:CustomStyle ; #custom-alias -->
```

Custom Alias Behavior

Custom Aliases create an entry in the document's WIF that enable linking to the element it was created with.

Using a Custom Alias

The first step in using a Custom Alias is to create one by adding the Alias tag to the location that will be linked to. Below, the Custom Alias `#my-alias` is created, associated with a Heading 1.

```
<!--#my-alias-->
```

```
# A Topic Heading
```

After that, the Alias `#my-alias` can be used in the path value for Links and Link References. Once the link is clicked, it will take readers to the location the Alias was created.

Refer to the linked sections for detailed instructions on using these elements.

Some quick examples for Links:

```
[link text](#my-alias)
```

```
[link text](my_doc.md#my-alias)
```

Some quick examples for Link References:

```
[link text][link_key]

[link_key]: #my-alias
```

```
[link text][link_key]

[link_key]: my_doc.md#my-alias
```

Markers in Markdown++

Markers are pieces of metadata that can be inserted into a document to add different features or change behavior of the publication. Markers are useful for many things, like adding keywords to a topic to improve search relevance, or instructing ePublisher to pass text through to the output without any processing.

Syntax

The Markers command has two main parts. Start the command with `markers:`, and follow it with a JSON Object Literal. The Marker name is written in the object key area, and the Marker value goes into the object value area. Make sure all keys and all values are wrapped in `"`, and separated by `:`. Multiple key/value pairs for Markers can be written, separated with `,`.

Basics

A basic Markers command with a Keywords marker.

```
<!--markers:{"Keywords": "webworks"}-->

# About WebWorks
```

Multiple Markers can be added in one command. Separate the key/value pairs with `,`.

```
<!--markers:{"Keywords": "webworks"}-->

# About WebWorks
```

The Markers command is available for inline-level syntax as well.

```
Add a custom <!--markers:{"Keywords": "inline, marker"}-->**marker**.
```

Mix with Other Commands

Markers can be in the same comment tag with other commands. Separate them with a `;` character. A Custom Style and Markers are written in the same tag below.

```
<!-- style:CustomStyle ; markers:{"Keywords": "webworks"} -->
```

Markers Behavior

Markers associate a piece of data with an element on the page. To learn more about ePublisher's built-in Markers and what they do, see the Markers reference table.

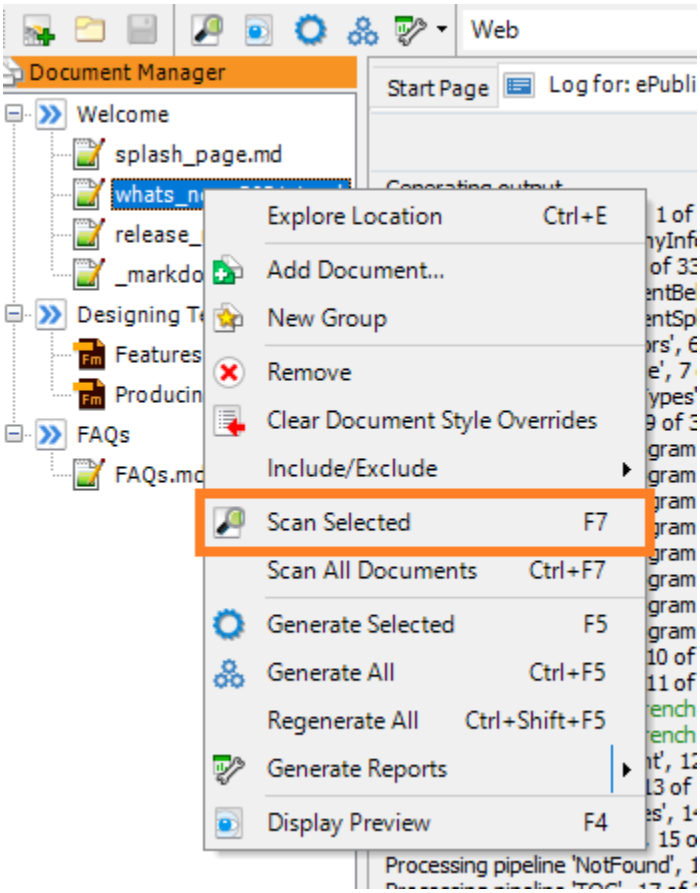
Using a Marker

First thing to do is write the Marker in the intended area of the content. The Marker needs to be tagged to a content element, like a Paragraph or Heading.

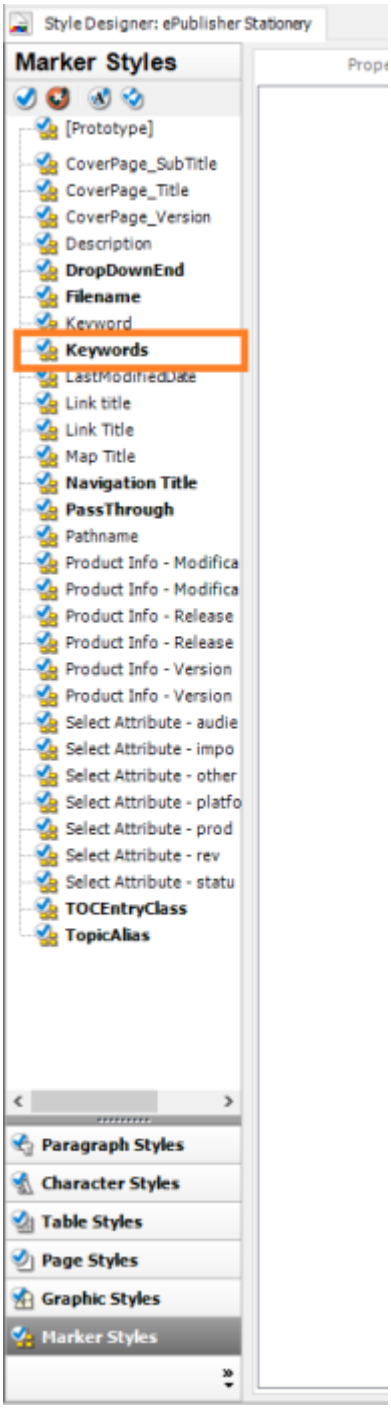
Below, a Keywords Marker is tagged to a Heading 1.

```
<!--markers:{"Keywords": "markers, content, create"}-->

# Using Markers in Source Content
```

Next, scan the document in ePublisher. This will add the Marker in the Marker Styles area of the Style Designer.



With the Marker added to ePublisher, output can now be generated with new Keywords added. This Marker will improve search relevance in outputs with searching capabilities, like Reverb 2.0.


Style Designer: ePublisher Stationery
Marker Styles
[Prototype]
CoverPage_SubTitle
CoverPage_Title
CoverPage_Version
Description
DropDownEnd
Filename
Keyword
Keywords
LastModifiedDate
Link title
Link Title
Map Title
Navigation Title
PassThrough
Pathname
Product Info - Modifica
Product Info - Modifica
Product Info - Release
Product Info - Release
Product Info - Version
Product Info - Version
Select Attribute - audie
Select Attribute - impo
Select Attribute - other
Select Attribute - platfo
Select Attribute - prod
Select Attribute - rev
Select Attribute - statu
TOCEntryClass
TopicAlias
Paragraph Styles
Character Styles
Table Styles
Page Styles
Graphic Styles
Marker Styles

Conditions

Conditions allow an author to create sections of content that are available in certain contexts, and excluded in others. They can be useful for the Edit/Review process, or to switch out sections meant only for print or the Web.

Syntax

Conditional text is written between two HTML Comment Tags. In the first tag, write one or more conditions between `<!--condition:` and `-->`. The second closing tag is always written as `<!--/condition-->`. Between the tags, any valid Markdown++ content can be written. Condition names must only use alphanumeric characters, `-`, and `_`. Do not use spaces in Condition names.

See Condition Operators for details on advanced conditional logic syntax.

Basics

A Condition containing a single Paragraph, using the Condition `print_only`.

```
<!--condition:print_only-->

This paragraph is meant only for print.

<!--/condition-->
```

Conditions can contain multiple block-level elements.

```
<!--condition:print_only-->

# The Print Section


This sections is meant only for print.


It will not be visible if `print_only` is set to `Hidden`.

<!--/condition-->
```

Conditions can be used with inline content, too.

```
Go to the Section <!--condition:print_only-->on page 304<!--/
condition--> for more details.
```

Operators

Complex logic can be used with Conditions using a set of Operators. This block is hidden when `production` is set `Visible` using the `!` (logical NOT) operator.

```
<!--condition:!production-->

This paragraph is not meant for production publications.

<!--/condition-->
```

Multiple Conditions

Multiple Conditions can be used in a single conditional block Operators. This example only show the block of text if `print_only` AND `production` are set to `Visible`.

```
<!--condition:print_only production-->

This paragraph is meant only for print and production.

<!--/condition-->
```

Conditions Behavior

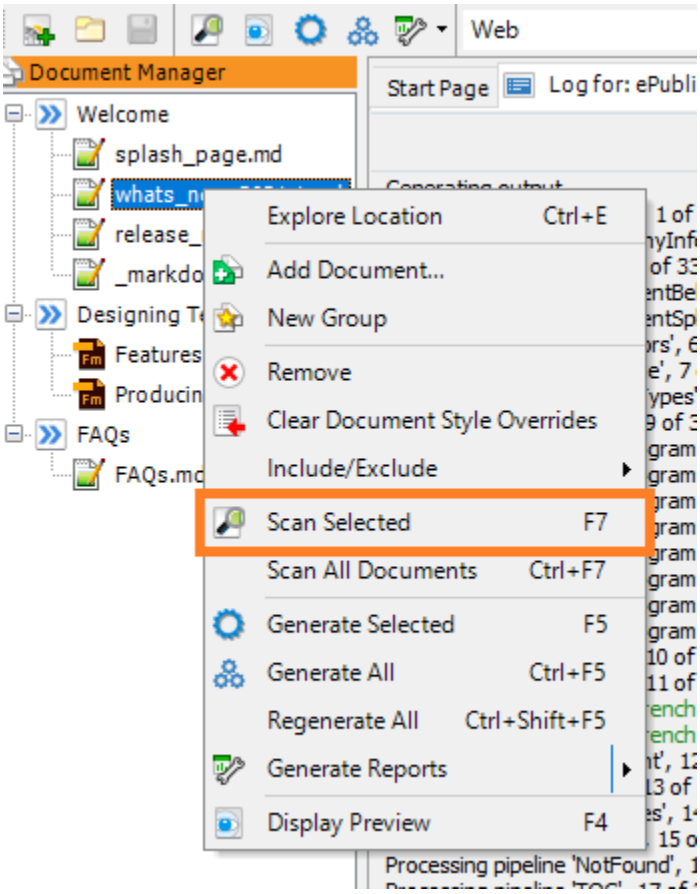
Conditions are rendered or removed from the document when publishing based on values set in the Conditions Window. Conditions are considered unset, and therefore always render, if they haven't been scanned into ePublisher. Conditions are also considered unset if they still have the default value `Use document condition` in the Conditions Window.

Using Conditions

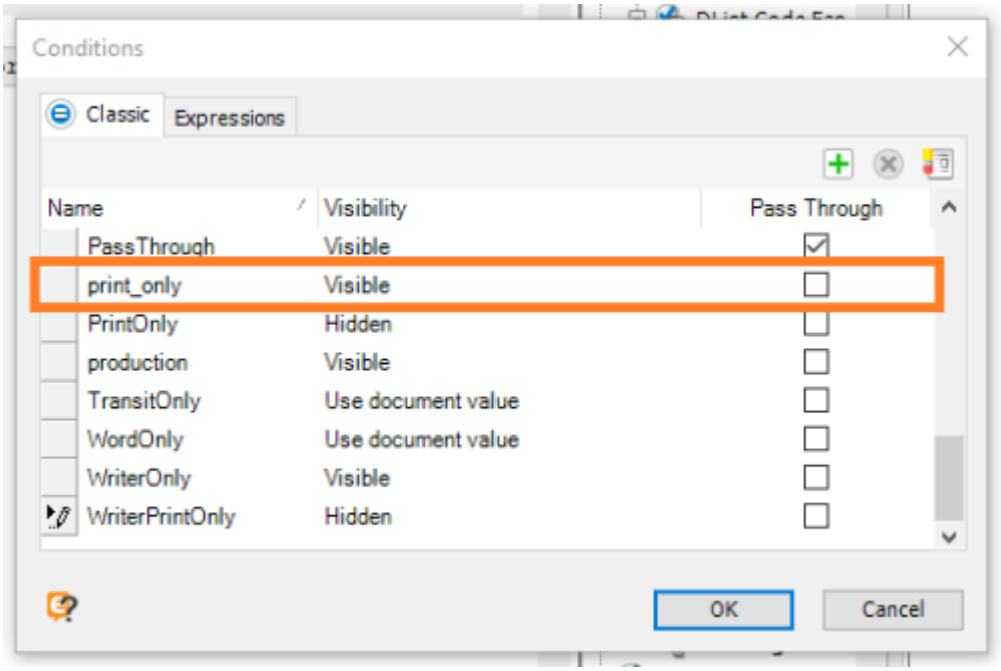
First, create a condition by writing it in a source document. Below, and block of conditional text is created with the Condition `print_only`.

```
<!--condition:print_only-->

# The Print Section


This sections is meant only for print.


It will not be visible if `print_only` is set to `Hidden`.

<!--/condition-->
```

Next, scan the document in ePublisher. This will add the Condition `print_only` to the Conditions Window.



Once scanned, the `print_only` Condition's value can be changed to either `Visible` or `Hidden`. The Condition is considered unset, and will always render, if the value is left with the default, `Use document value`.



Condition Operators

Using Operators, an author can create additional logic to determine whether conditional text should be rendered or hidden.

In a conditional statement, the block of text renders if the entire statement evaluates to `true`. The block is hidden if the statement evaluates to `false`.

In this context, `Visible` is considered `true`, and `Hidden` is considered `false`. `Use document value` disables the conditional statement and always renders the block.

Combine Condition names and Operators to create complex statements to determine if the content should be rendered or removed in the publication.

The Space Operator - Logical AND

The space character in a conditional statement is a Logical AND. Meaning, if the statements on both sides of ` ` evaluate to `true`, the statement passes.

The conditional text below is rendered if `print_only` AND `production` are set to `Visible`.

```
<!--condition:print_only production-->

This paragraph is meant only for print and production.

<!--/condition-->
```

The `,` Operator - Logical OR

The `,` character in a conditional statement is a Logical OR. Meaning, if a statement on either side of `,` evaluates to `true`, the statement passes.

The conditional text below is rendered if one of `print_only` OR `production` are set to `Visible`.

```
<!--condition:print_only,production-->

This paragraph is meant for either print or production.

<!--/condition-->
```

The `!` Operator - Logical NOT

The `!` character in a conditional statement is a Logical NOT. Adding `!` to the beginning of a Condition reverses it's truthiness. Meaning, a Condition with `!` on the front of it's name evaluates `Visible` to `false` and `Hidden` to `true`.

The conditional text below is removed if `production` is set to `Visible`.

```
<!--condition:!production-->

This paragraph is not meant for production.

<!--/condition-->
```

Conditions and Includes

Conditions can be used as expected in File Includes since they are processed at the same time as Includes. Additionally, Includes can be used inside of conditions to import entire documents based on certain Conditions.

File Includes

A File Include statement points to another Markdown++ file and imports the file's contents at the location of the statement. This enables multi-file structure in a single document.

Syntax

An Include statement is created by writing a path to a Markdown++ document between `<!--include:` and `-->`. Relative paths and absolute paths are both valid path values. Web paths are not supported. The include statement must be the only thing on a line.

Basics

A basic Include statement. The Include must be written on it's own line to work properly.

```
<!--include:my_file.md-->
```

Relative paths and absolute paths are fine to use.

```
<!--include:my_file.md-->
```

```
<!--include:C:/Users/Me/Docs/my_file.md-->
```

Multiple includes can be used in the same document.

```
<!--include:my_file.md-->


<!--include:doc_2.md-->
```

File Includes Behavior

When ePublisher detects a File Include statement, the file is read, and the Include tag is replaced with the content of the file. If the no file is found at the path given, the Include tag will be passed through to the output as an HTML Comment.

Using a File Include

To use an Include statement, all that needs to be done is write the tag where the file's content is to be imported. Below, an include statement is written below a Title.

```
Learning ePublisher

==================


<!--include:epublisher_basics.md-->
```

This can even be done inside of documents used in an Include statement, as long as it is not a [Recursive Include][mdp-includes-recursion]. Use this feature to create Map Files for many Markdown++ documents, or create documents needed for content re-use.

Recursive Includes

If an Include statement tries to insert a document that has already been inserted by a parent file, ePublisher's generation log will display a message like this one:

```
[Warning]

Skipping recursive include file:

 'C:\Users\Me\Documents\include_doc.md'

 in file: 'C:\Users\Me\main_doc.md'
```

This message displays because ePublisher cannot insert the document. Doing so would create a recursive loop and would break the generation. If this message is recieved, it's time to look at the layout of Includes in the source documents.

The message can be useful to track down the file in error. The first file path refers to the file in the attempted Include statement. The second file path refers to where the Include occured.

Variables

Variables represent a shorthand to store a value that can be re-used across a set of documents. They're useful to store content that only needs to be written once, but is used in the same way in many places. Store values like product names, copyright text, publication dates, and more inside of Variables.

Syntax

Variables are the only syntax in Markdown++ that doesn't use the HTML Comment Tag.

To write one, start with `$`, write the variable name using alphanumeric characters, `-`, and `_`. End the Variable with `;`. Do not use spaces in the Variables's name.

Basics

A simple example of writing a Variable, called `product_name`.

```
$product_name;
```

Variables can be intermixed with other text content.

```
Document last published on $publish_date;.
```

The full range of Markdown features is available with Variables as well, both around them and written in their values.

```
The documentation for our product, **$product_name;**.
```

Variable Behavior

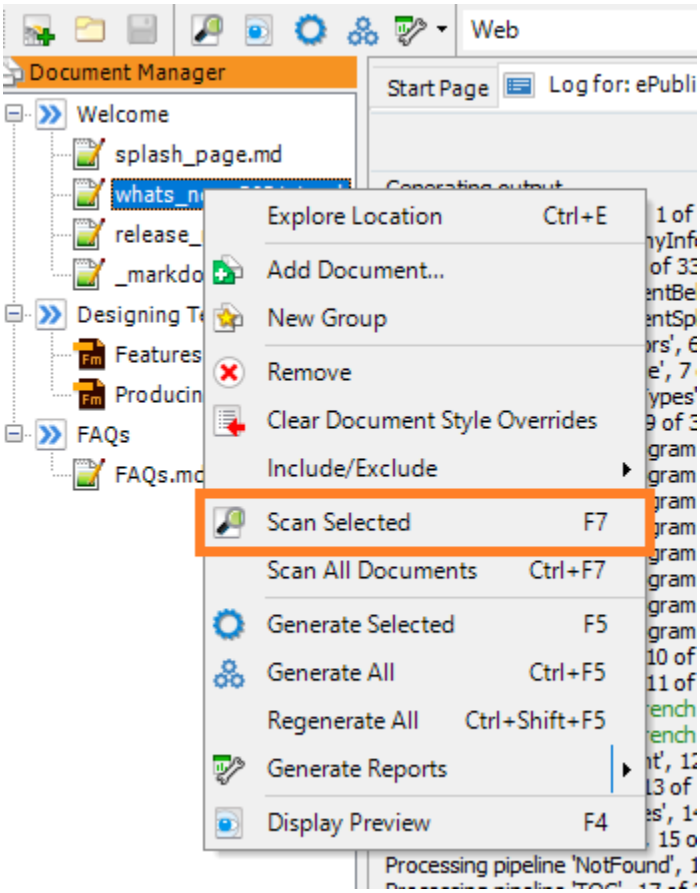
Variables, once scanned into ePublisher, can be given values that are saved to an ePublisher Project.

Using Variables

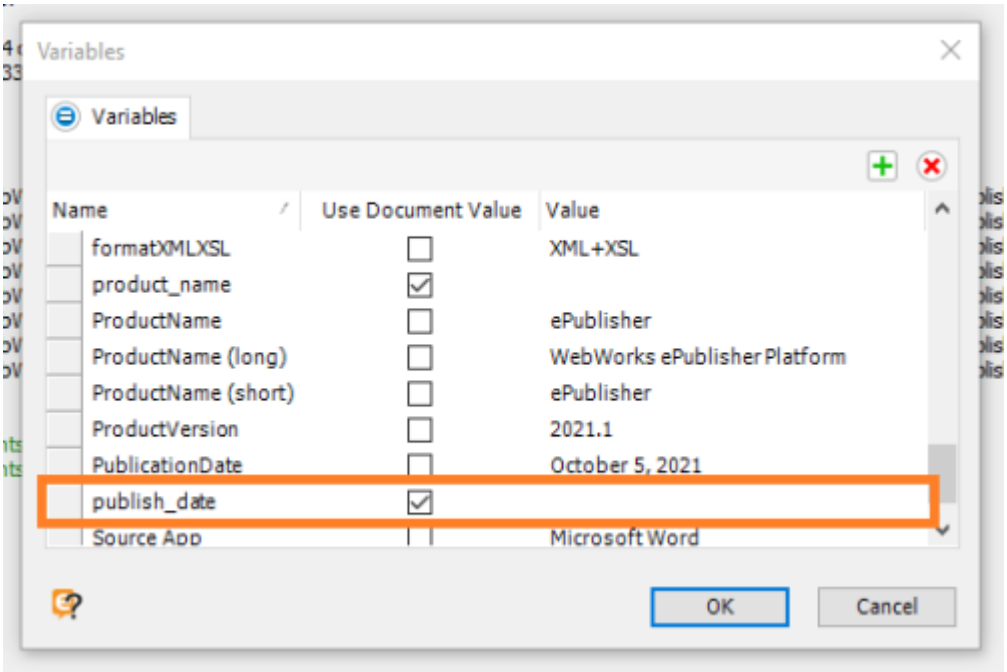
First, a Variable must be created by writing it into a document. Here, we create a Variable called `publish_date`.

```
Document last published on $publish_date;.
```

Next, scan the document in ePublisher. This will add the Variable to the Variables Window.



The Variable can now be given a value, typed in the input field next to the Variable's name in the Variable Window.



Use Document Value

The **Use Document Value** checkbox inside the Variables Window does not apply to Markdown++ Variables, since their values are instead maintained in ePublisher. There's no change in behavior based on if the box is checked or not. This feature applies to legacy source document types, such as FrameMaker and Word.

See Online Help for Markdown++ cheatsheet.

Adobe FrameMaker

If you want to implement online content features in your generated output, you need to prepare your Adobe FrameMaker source documents for output generation. This section explains how to prepare your Adobe FrameMaker source documents.

Adobe FrameMaker Formats and Standards

Adobe FrameMaker provides a comprehensive publishing solution with XML-based structured authoring. You can develop the templates you need to deliver polished technical documentation for large product libraries. FrameMaker allows you to create both structured and unstructured content. You can also create DITA-compliant content.

This section describes the design considerations for a FrameMaker catalog and template files. By effectively designing your FrameMaker template, and by consistently applying formats throughout your source documents, you can streamline single-sourcing processes and reduce your production and maintenance costs. This section does not describe all FrameMaker processes, but it focuses on the design considerations related to ePublisher.

Standards for Single-Sourcing

To define your FrameMaker standards, create a FrameMaker file with all the elements you need in it, including formats, markers, conditions, variables, tables, and master pages. You can use this file to import and update your standards. For example, you can use this file to update variables and conditions across your FrameMaker source files. You can also use this file as a source document in your Stationery design project. To create a template file, start with one of the default templates provided with FrameMaker that most closely matches the format you want. Then, customize the default file to meet your specific needs. The following sections describe various template areas and considerations, and how to effectively design your FrameMaker template file to support single-sourcing with ePublisher.

Planning for Importing Elements Across Files

You need to carefully plan your FrameMaker standards so you can import all elements, such as formats, page layouts, variables, and conditions across all source files. To avoid issues, do not reuse an element for two different purposes in two different files. For example, if the footer text differs between the front matter and the main chapters of a book, do not use the same variable to define the footer in both files. Otherwise, you cannot import that variable across files.

To avoid conflicts when importing master pages, use different page layout names for special pages in all files, such as Title, TOC right, TOC left, and Index first. In addition, delete unneeded formats, variables, and conditions to simplify your template use and maintenance.

Paragraph Formats in FrameMaker

Create paragraph formats for items based on function, not based on formatting. This approach allows you to modify formatting over time and the format names continue to apply. It also prepares you for structured writing in the future. If you are using DITA, paragraph formats are already defined.

Name your paragraph formats starting with naming conventions that group formats by function. For example, group procedure-related formats together by starting the format names with Procedure, such as ProcedureIntro, ProcedureStep1, ProcedureStep, ProcedureSubStep1, and ProcedureSubstep. You do not need to restart numbering using a step1 format. If you have a format that always proceeds a numbered list, such as a ProcedureIntro, you can restart the numbering with that format, which allows you to not use a step1. Either method is fine, but one can require less maintenance when updating steps in a procedure topic.

Note: Format names should not include a period in their name. The period can cause display issues when ePublisher creates the cascading style sheet entry that defines the appearance of the format.

To simplify formatting and save time for future maintenance and customization, set the default paragraph font for all formats, then customize specific formats that need customization. You may need multiple paragraph formats to define functions that support pagination settings, such as a BodyListIntro format that has **Keep with next paragraph** set.

In ePublisher, you can scan the source documents to list all the paragraph formats. Then, you can organize them in ePublisher to allow property inheritance and to streamline the customization process for your generated output.

To automate and simplify template use, define the paragraph format that follows each paragraph format. This process allows the writer to press Enter after writing a paragraph and the template creates the next paragraph with the format most commonly used next. For example, after a Heading format, the writer most often writes a body paragraph of content.

Common paragraph formats include:

- Anchors for images and tables. You may need multiple indents, such as Anchor, AnchorInList, and AnchorInList2.
- Body paragraphs. You may need multiple indents, such as Body, BodyInList, and BodyInList2.
- Headings, such as ChapterTitle, AppendixTitle, Heading1, Heading2, Heading3, and Heading4. You may also need specialized headings, such as Title, Subtitle, FrontMatterHeading1, FrontMatterHeading2, and FrontMatterHeading3.
- Bulleted lists. You may need multiple bullet levels, such as Bullet, Bullet2, Bullet3. You may also need a bullet item within a procedure, such as a ProcedureBullet and a bullet item within a table, such as a CellBullet. For more information, see “Bulleted and Numbered Lists in FrameMaker”.
- Numbered lists. You may need multiple levels, such as ProcedureStep that uses numbers and ProcedureSubstep that uses lowercase letters. You can use ProcedureStep1 and ProcedureSubstep1 to restart numbering, or you can use a common paragraph that precedes each list to restart numbering. You may also need numbered list items in tables, such as CellStep and CellStep1. Be

sure to consider related supporting formats, such as ProcedureIntro. For more information, see "Bulleted and Numbered Lists in FrameMaker".

- Examples, such as code or command syntax statements, usually in a fixed font. To keep the lines of a code example together, you can set the Example format to keep with next paragraph and use an ExampleLast format to identify the end of the example. You may also need multiple example levels, such as ExampleInList and ExampleInListLast.
- Paragraphs in tables, such as CellHeading, CellBody, CellBody2, CellStep, CellStep1, and CellBullet. Although you can reuse paragraph formats in tables and adjust the margins when those formats are in a table in FrameMaker, create unique paragraph formats for use in tables to give you complete control in ePublisher.
- Legal notice and copyright or trademark formats for inside the cover page.
- Table of contents and Index formats. However, these formats are defined on the reference pages rather than as paragraph formats.
- Definition lists, such as term and definition or description. You can use a two-column table for this purpose, but a definition list allows long terms, such as field labels in a user interface, to run across the page without wrapping. Then, the definition or description are indented below the term.
- Header and footer formats to control formatting.
- Notes, cautions, tips, and warnings. You can use the numbering property of a paragraph format to insert default text at the beginning of a paragraph, such as Note, Caution, Tip, or Warning. In ePublisher, you can use the **Bullet** properties for the paragraph style to add an image to the left of each note, caution, tip, or warning.
- Page breaks, which can be identified with a small-font paragraph format with **Keep with previous paragraph** set and a large space below the paragraph that pushes the next paragraph to the next page. This paragraph format is hidden in online content. This approach allows you to put page breaks in the content where needed to achieve the cleanest printed output without customizing the pagination settings for individual paragraphs. However, you need to review all these paragraphs each release and remove unneeded PageBreak paragraphs. This approach increases maintenance, but it prevents format customization for pagination.

ePublisher projects use custom marker types, paragraph formats, and character formats to define online features. You need to give the list of markers and formats to the writers so they know how to implement each online feature. The writers use the markers and formats you create to define online features.

The Stationery defines the custom markers and formats. To reduce complexity, you can use the format names defined in the documentation, or you can define the online feature to a different format. The following list identifies additional paragraph formats you may need to support ePublisher online content features:

- Paragraph or character formats to support multiple languages, such as bidirectional languages and text.
- Dropdown paragraph format that identifies the start of an expand/collapse section. You can end the section with a paragraph format defined to end the section, or with a DropDownEnd marker.
- Popup paragraph formats that define several aspects of popup window content:

- Popup paragraph format identifies the content to display in a popup window and in a standard help topic. This format is applied to the first paragraph of popup content.
- Popup Append paragraph format identifies the content to display in a popup window and in a standard help topic. This format is applied to additional popup paragraphs when you have more than one paragraph of content to include in a popup window.
- Popup Only paragraph format identifies the content to display only in a popup window. This format is applied to the first paragraph of popup content.
- Popup Only Append paragraph format identifies the content to display only in a popup window. This format is applied to additional popup paragraphs when you have more than one paragraph of content to include in a popup window.

- Related topics paragraph format that identifies a link to a related topic, such as a concept topic related to a task or a task related to a concept.
- See Also paragraph format that identifies the text you want to include in an inline See Also link.

For more information about enabling a specific online feature, see “Designing, Deploying, and Managing Stationery”.

Character Formats in FrameMaker

Create character formats for items based on function, not based on formatting or appearance. This approach allows you to modify formatting over time and the format names continue to apply. It also prepares you for structured writing in the future. If you are using DITA, character formats are already defined.

For character formats, use As Is to start with as base, which allows you to apply multiple character formats to the same text. It also allows each character format to define only the aspects of the formatting required for that character format. Customize each format for your specific need by specifying only the properties required for that format.

Common character formats include:

- Book titles in cross references
- Emphasized text
- Command names
- File and folder names
- User interface items
- Optional steps or if clauses used to introduce optional steps
- Links
- New terms
- Step numbers, which allows you to apply formatting to the number for a step
- Text the user must type
- Variables

ePublisher projects use custom marker types, paragraph formats, and character formats to define online features. You need to give the list of marker types and formats to the writers so they know how to implement each online feature. The writers use the markers and formats you create to define online features.

The Stationery defines the custom marker types, paragraph formats, and character formats. To reduce complexity, you can use the format names defined in the documentation, or you can define the online feature to a different format. The following list identifies additional character formats you may need to support ePublisher online content features:

- Link character format, which identifies the text to include in the link. Include the marker and text in the Link character format.
- Multiple language support, such as bidirectional languages and text, can require a paragraph or character format with Bidi support enabled.
- Abbreviation character format identifies abbreviation alternate text for browsers to display for abbreviations, such as SS#, when a user hovers over the abbreviation in output. Screen readers

also can read the abbreviation alternate text. This character format is used in combination with the AbbreviationTitle marker type.

- Acronym character format identifies acronym alternate text for browsers to display for acronyms, such as HTML, when a user hovers over the acronym in output. Screen readers can also read the acronym alternate text. This character format is used in combination with the AcronymTitle marker type.
- Citation character format identifies the source of a quote using a fully-qualified Uniform Resource Identifier (URI) when a user hovers over the quote in output. Screen readers can also read the URI for the quote. This character format is used in combination with the Citation marker type.
- See Also character format identifies the text you want to include in a See Also button. This format controls the appearance of the text on the button.

For more information about enabling a specific online feature, see “Designing, Deploying, and Managing Stationery”.

Bulleted and Numbered Lists in FrameMaker

ePublisher uses a table-like structure with two columns to display any paragraph format with a hanging indent, such as bulleted and numbered list items, in generated output. ePublisher uses the numbers, characters, formats, and fonts from the source documents for the bullets or numbers. Since some fonts are not available on all computers, you should use character formats in ePublisher to override the formatting of the bullets or numbers. You can also use an image in ePublisher for bullets.

For bulleted lists, you may need multiple bullet levels, such as Bullet, Bullet2, and Bullet3. You may also need a format for a bullet within a procedure, such as a ProcedureBullet, and a bullet within a table, such as a CellBullet. Make sure you consider all supporting formats you may need, such as a ListIntro format for the paragraph that introduces the bulleted list, which should be set to stay with the list (Keep with Next).

For numbered lists, you may need multiple levels, such as a ProcedureStep that uses numbers and a ProcedureSubstep that uses lowercase letters. To restart numbering, you can use a ProcedureStep1 and a ProcedureSubstep1 format. If you have a common paragraph format that precedes each list, you can use that paragraph format to restart numbering, which would eliminate the need for a ProcedureStep1 format. You may also need a numbered list item in tables, such as CellStep and CellStep1. Make sure you consider all supporting formats you may need, such as a ProcedureIntro format

Note: Be aware of paragraphs that have a hanging indent. The hanging indent can cause incorrect alignment of text on the first line of your generated output. For more information see "Defining the Appearance of Numbered Lists".

Image Formats and Considerations in FrameMaker

If ePublisher cannot use an original image in the output, or if ePublisher determines it needs to modify the image based on how it is included in the source document, ePublisher rasterizes the image using the options you define for your graphic styles in Style Designer. For example, you can define the dots per inch (DPI) and format for the final images. Rasterization of an image can cause the image to be less clear in the output.

To avoid reduced image quality in your output, and to avoid an extended transformation time during the Image stage and pipeline, review the following considerations:

- When ePublisher encounters an anchored frame in your FrameMaker source documents, ePublisher checks for the following conditions:
  - Is the frame a different size than the original image?
  - Is there white space in the frame?
  - Is the image copied into the document, rather than imported by reference?
  - Is the original image a file format other than `.jpg`, `.gif`, `.png`, or `.svg`?
  - Are there additional elements in the frame, such as text boxes, multiple images, or callouts?

  If ePublisher determines that any of these conditions apply, ePublisher rasterizes the entire frame and applies the options you defined in Style Designer.
- To display images at full size in online output and avoid resizing, which can cause the image to be rasterized, set the **By reference graphics use document dimensions** option for your graphic styles to **Disabled**.
- If you want ePublisher to rasterize all images according to your Style Designer options, set the **By reference graphics** option to **Disabled** for all graphic styles.
- When ePublisher finds an image included by-reference that is the original size, is shrink-wrapped, and contains no callouts, ePublisher copies the image directly into the output folder in most cases, bypassing the graphic style options.
- To improve the image quality in your output, resize your images as needed using an image editing application before importing them, rather than adjusting the DPI or scale in FrameMaker. Otherwise, an image included by reference retains its original file size, and it is either scaled by the browser or rasterized according to the size of the anchored frame, which can result in a distorted image.
- For the best compatibility with most computer monitors, save and import your images at 96 DPI using a format that ePublisher does not rasterize.
- Image callouts are useful in many publications. However, text boxes and line drawings cause images to be rasterized, which can make images less clear in your output. Add and edit callouts in your image editing application and then import the single, final image to avoid the rasterization process.

- If you use `.svg` image files, you need to configure the `.svg` options to specify whether to rasterize these images. Some output formats and some browsers do not support `.svg` image files.
- You can add text boxes with GraphicStyle markers to your images without causing the image to be rasterized, since markers do not affect the appearance or format of an anchored frame.
- ePublisher does not include images from FrameMaker Master or Reference pages, and it does not include content outside the main text flow.
- Store image files and source documents on the local computer when generating output.

To achieve the best results when inserting images in FrameMaker

1. Create a unique paragraph format for images. Use the paragraph alignment properties to control the position of your images. Make sure the **Fixed** check box in the **Line Spacing** section is *not* selected for the paragraph format.
2. Insert an anchored frame in an empty paragraph of the format created in step 1.
3. Import your image file by reference into the anchored frame rather than copying it into the document. ePublisher supports only `.jpg`, `.gif`, `.png`, and `.svg` files. ePublisher rasterizes all other formats.
4. Import the image at the native resolution of the image.
5. Shrink-wrap the frame (type Esc+m+p with the frame selected) and change its **Anchoring Position** to **At Insertion Point** or **Below Current Line**.

Table Formats in FrameMaker

Table formats allow you to define standard tables and quickly apply those standards to tables in your source documents. When you define your table formats, be sure to consider the various types of tables you may need, such as with lines, without lines, checklists, and action/result tables. You can use a table without lines to layout content within an area on a page, such as a definition list with short terms. You can also create a table format for each indent position needed. For example, you can create a table format to use for tables within a bulleted list that is indented to align with the text of each bulleted list item.

ePublisher allows you to define how the header, footer, and main rows of a table appear in your generated output. To support these formatting properties, your tables must have each of these parts defined in your source documents. If a table does not have a header defined, ePublisher cannot apply the formatting defined for the header row.

ePublisher applies the paragraph and character formats you define for content within each cell. You can also configure ePublisher to ignore character formats in a table. You may need additional paragraph formats to use in tables, such as CellBody and CellBullet, so you can define the proper margins and appearance for your generated output. You cannot adjust paragraph formats to change their appearance when used within tables.

Cross Reference Formats in FrameMaker

Cross reference formats allow you to quickly use consistent cross references throughout your source documents. However, you probably want to change the appearance of your cross references in your generated output. For example, you may not want to include page numbers in your online content. ePublisher allows you to define how each cross reference format appears in your generated output.

Define the cross reference formats you need in your source documents, such as references to headings, steps, figures, tables, and chapters. Then, you can define each of these formats separately in ePublisher.

Markers in FrameMaker

FrameMaker uses markers to implement standard features, such as index entries and hypertext links. ePublisher recognizes these standard markers and uses them to implement these standard features in your generated output.

ePublisher projects also use custom marker types, paragraph formats, and character formats to define online features. You need to give the list of marker types and formats to the writers so they know how to implement each online feature. The writers use the markers and formats you create to define online features.

The Stationery defines the custom marker types, paragraph formats, and character formats. Markers with reserved names have their functions defined by default. You can use these default names, or you can create your own markers. To reduce complexity, use the default marker names, which are also used throughout the documentation. You can also use the format names defined in the documentation to reduce complexity. The following table lists the default custom marker types used to implement online features.

Marker Type	Description
AbbreviationTitle	Specifies abbreviation alternate text for browsers to display for abbreviations such as SS# when a user hovers over the abbreviation in output. Screen readers also can read the abbreviation alternate text. Used in combination with the Abbreviation character format.
AcronymTitle	Specifies acronym alternate text for browsers to display for acronyms such as HTML when a user hovers over the acronym in output. Screen readers can also read the acronym alternate text. Used in combination with the Acronym character format.
Citation	Specifies the source of a quote using a fully qualified Uniform Resource Identifier (URI) when a user hovers over the quote in output. Screen readers can also read the URI for the quote. Used in combination with the Citation character format.
Context Plugin	Specifies context plug-ins for Eclipse help systems. Other Eclipse plug-ins can use the context plug-in IDs to call the Eclipse help system. For more information, see “Using Markers to Specify Context Plug-ins in Eclipse Help”.
DropDownEnd	Marks the end of an expand/collapse section. Used in conjunction with an Expand/Collapse paragraph format.
Filename	Specifies the name of an output file for a page or an image.
GraphicScale	Specifies a percentage to use to resize an image, such as 50 or 75 percent, in generated output.
GraphicStyle	Specifies the name of a graphic style defined in a project to apply to an image. This marker type is an internal marker type that is not displayed in

Marker Type	Description
	Stationery Designer. You cannot create a marker type with a different name and assign it this functionality.
Hypertext	Specifies a link using the `newlink` and `gotolink` commands in Adobe FrameMaker. This marker type is a default Adobe FrameMaker marker type ePublisher automatically maps.
ImageAltText	Specifies alternate text for an image.This text is added to the `alt` attribute of the `img` tag in the output. Screen readers use this text when you create accessible content.
ImageAreaAltText	Specifies alternate text for clickable regions in an image map. This text is added to the `alt` attribute of the `img` tag in the output. Screen readers use this text when you create accessible content.
ImageLongDescByRef	Specifies the path to the file that contains the long description for an image. This text is added to the `longdesc` attribute of the `img` tag in the output. Screen readers read this description when you create accessible content.
ImageLongDescNotReq	Specifies that a long description is not required for an image, which bypasses this accessibility check for the image when you create accessible content.
ImageLongDescText	Specifies the long description for an image. This text is added to the `longdesc` attribute of the `img` tag in the output. Screen readers read this description when you create accessible content.
Keywords	Specifies the keywords to include in the `meta` tag for the topic. The `meta` tag improves searchability on the Web.
PageStyle	Specifies the name of a page style defined in the project to apply to a topic. This marker type is an internal marker type that is not displayed in Stationery Designer. You cannot create a marker type with a different name and assign it this functionality.
PassThrough	Specifies that ePublisher place the contents of the marker directly into the generated output without processing the content in any way. For example, you could use a PassThrough marker if you wanted to embed HTML code within your generated output.
Popup	Specifies the start of the content to include in a popup window. The content is displayed in a popup window when you hover over the link. When you click the link in some output formats, the topic where the popup text is stored, such as the glossary, is displayed.
PopupEnd	Marks the end of the content to include in a popup window.
PopupOnly	Specifies the start of the content to include in only a popup window. Browsers display the content in a popup window when you hover over or click the link.
RubiComposite	No longer supported.

Marker Type	Description
SeeAlsoKeyword	Specifies an internal identifier for a topic. SeeAlsoLink markers in other topics can list this identifier to create a link to this topic. Used in conjunction with a See Also paragraph format or character format.
SeeAlsoLink	Identifies an internal identifier from another topic to include in the list of See Also links in this topic. Used in conjunction with a See Also paragraph format or character format.
SeeAlsoLinkDisplayType	Specifies whether to display the target topics on a popup menu or in a window. By default, the links are displayed in the Topics Found window. To display a popup menu, set the value to `menu`. This marker type is supported only in HTML Help.
SeeAlsoLinkWindowType	Specifies the name of the window defined in the `.hhp` file, such as TriPane or Main, that the topic opens in when the user clicks the link. This marker type is supported only in HTML Help.
TableSummary	Specifies an alternate text summary for a table, which is used when you create accessible content. This text is added to the `summary` attribute of the `table` tag in the output. Screen readers read this description when you create accessible content.
TableSummaryNotReq	Specifies that a summary is not required for a table, which bypasses this accessibility check for that table.
TOCIconHTMLHelp	Identifies the image to use as the table of contents icon for a topic in the HTML Help output format.
TOCIconJavaHelp	Identifies the image to use as the table of contents icon for a topic in the Sun JavaHelp output format.
TOCIconOracleHelp	Identifies the image to use as the table of contents icon for a topic in the Oracle Help output format.
TOCIconWWHelp	Identifies the image to use as the table of contents icon for a topic in the WebWorks Help output format.
TopicAlias	Specifies an internal identifier for a topic that can be used to create a context-sensitive link to that topic.
TopicDescription	Specifies a topic description for a context-sensitive help topic in Eclipse help systems. For more information, see “Using Markers to Specify Topic Descriptions for Context-Sensitive Help Topics in Eclipse Help”.
WhatIsThisID	Identifies a What Is This help internal identifier for creating context-sensitive What Is This field-level help for Microsoft HTML Help.
WindowType	Specifies the name of the window defined in the help project that the topic should be displayed in. In Microsoft HTML Help, the window names are

Marker Type	Description
	defined in the `.hhp` file. This marker type is supported in Microsoft HTML Help and Oracle Help.

Variables and Conditions in FrameMaker

Variables allow you to define a phrase once and consistently use that phrase throughout your source documents. Then, if you ever need to change that phrase, you can change it in one location and apply that change throughout your source documents. For example, you can use variables for product names, book titles, and company names.

Conditions allow you to single-source information and include or exclude specific sets of information. You can apply a condition to a character, word, sentence, paragraph, or entire sections of content. Then, you can specify whether to show or hide the content with that condition applied to it. This capability allows you to create multiple version of content based on your specific needs. You can also use conditions to include and exclude notes to the writer or reviewer during the content development process. When you combine variables and conditions, you can customize information for multiple versions of a product while reducing your maintenance costs by reducing duplicate information.

When working with conditions, you can customize the appearance of content with a condition applied by using color, underline, overline, and strikethrough formatting for the condition. This formatting helps you maintain and work with the content. However, ePublisher does not display this formatting in the generated output. If you want to apply formatting in the generated output, use paragraph and character formats to define the appearance for the content.

To simplify consistently setting variables and conditions across your source documents, create a standard file with all the variables and conditions defined in it. The file can display the value of each variable and the show/hide state of each condition. Writers can set the variables and conditions as needed in this one file. Then, the writers can import the variables and conditions from this file into all the source documents.

Note: Use each variable and condition for the same purpose and value in all source documents. For example, if you want the footer in the preface file to be different from the footer in the chapter files, use a different variable to define the footer in each file. Otherwise, you cannot import variables across all the source documents.

Page Layouts in FrameMaker

A **master page** defines the layout of one or more pages and includes all design elements, such as headers, footers, background text, and graphics, for every page that uses that master page. A master page allows you to define the layout of multiple pages in one place, and apply that layout to multiple pages. If you want to adjust the page layout, you need change it only in one place. Each template has at least one master page.

You can create multiple master pages, such as one for odd pages, one for even pages, and one for the first page of each chapter. To simplify page management and being able to import master pages across files, do not redefine a master page to have a different layout in different files. For example, if you want odd pages to have a different footer in the front matter than in the chapters, create a master page for each case, such as ChapterOdd and PrefaceOdd.

You may need to create many master pages for special purposes, such as Title, LegalNotice, PrefaceOdd, PrefaceEven, ChapterFirst, ChapterOdd, ChapterEven, AppendixFirst, AppendixOdd, AppendixEven, IndexFirst, IndexOdd, and IndexEven.

Reference Pages, Table of Contents, and Indexes in FrameMaker

Reference pages define default images, lines, heading levels, and formatting for generated table of contents and index files. You can use reference pages to simplify content creation in your source documents. For more information about reference pages and formatting generated table of contents and indexes in your printed content, see the Adobe FrameMaker documentation and help.

Since graphics and lines defined on the reference pages are not in the main flow, ePublisher cannot include these items in the generated output. For images and lines, use anchored frames in your content to include the images by reference.

Since the appearance of online table of contents and indexes often differ from printed versions, you need to be able to deliver customized table of contents and indexes in your online content. Therefore, ePublisher does not need the table of contents and index formatting defined on the reference pages. ePublisher allows you to define the table of contents levels and appearance, as well as the appearance of the index in your generated output. ePublisher uses the index markers throughout your source files to build the online index. This powerful support allows you to deliver the online content you require.

Implementing Online Features in FrameMaker

Implement online features in your output by preparing your Adobe FrameMaker source documents with custom marker types, paragraph formats, and character formats defined by the Stationery designer for your Stationery. These markers and styles define the presentation and behavior or your online content. For example, markers can define the name of the file generated for a topic. Formats can define how content displays online.

Custom Marker Types in FrameMaker

ePublisher projects use the custom marker types to implement online features when generating output. Before you begin using custom marker types, talk to the Stationery designer and verify which online features your Stationery supports. Your Stationery only recognizes the custom marker types defined by the Stationery designer in your Stationery. If you try to implement online features using custom marker types not supported in your Stationery, ePublisher does not recognize these items when generating output. ePublisher correctly converts all standard Adobe FrameMaker marker types. In addition, ePublisher also supports several custom marker types you can use to implement online features in your generated output.

When the Stationery designer creates the Stationery, the Stationery designer can use the default name for a custom marker type or the Stationery designer can use a different name for the customer marker type. The following table lists the default names of custom marker types used to implement online features. Always verify with the Stationery designer the names of the custom marker types you should use when implementing online features before you use these items in your source documents.

Marker Type	Description
AbbreviationTitle marker type	Specifies abbreviation alternate text for browsers to display for abbreviations such as SS# when a user hovers over the abbreviation in output. Screen readers also can read the abbreviation alternate text. Used in combination with the Abbreviation character format. For more information, see “Assigning Alternate Text to Abbreviations in FrameMaker”
AcronymTitle marker type	Specifies acronym alternate text for browsers to display for acronyms such as HTML when a user hovers over the acronym in output. Screen readers can also read the acronym alternate text. Used in combination with the Acronym character format. For more information, see “Assigning Alternate Text to Acronyms in FrameMaker”.
Citation marker type	Specifies the source of a quote using a fully qualified Uniform Resource Identifier (URI) when a user hovers over the quote in output. Screen readers can also read the URI for the quote. Used in combination with the Citation character format. For more information, see “Providing Citations for Quotes in FrameMaker”.
Context Plugin marker type	Specifies context plug-ins for Eclipse help systems. Other Eclipse plug-ins can use the context plug-in IDs to call the Eclipse help system. For more information, see “Specifying Context Plug-ins in FrameMaker”.
DropDownEnd marker type	Marks the end of an expand/collapse section. Used in conjunction with an Expand/Collapse paragraph format. For more information, see “Creating Expand/Collapse Sections (Drop-Down Hotspots) in FrameMaker”.
Filename marker type	Specifies the name of an output file for a page or an image. For more information, see “Specifying Output File Names in FrameMaker”.
GraphicScale marker type	Specifies a percentage to use to resize an image, such as 50 or 75 percent, in generated output. For more information, see “Assigning Image Scales in FrameMaker”.

Marker Type	Description
GraphicStyle marker type	Specifies the name of a image style defined in a project to apply to an image. This marker type is an internal marker type that is not displayed in Stationery Designer. You cannot create a marker type with a different name and assign it this functionality. For more information, see “Assigning Image Styles in FrameMaker”.
Hypertext marker type	Specifies a link using the `newlink` and `gotolink` commands in Adobe FrameMaker. This marker type is a default Adobe FrameMaker marker type ePublisher automatically maps.
ImageAltText marker type	Specifies alternate text for an image.This text is added to the `alt` attribute of the `img` tag in the output. Screen readers use this text when you create accessible content. For more information, see “Assigning Alternate Text to Images in FrameMaker”.
ImageAreaAltText marker type	Specifies alternate text for clickable regions in an image map. This text is added to the `alt` attribute of the `img` tag in the output. Screen readers use this text when you create accessible content. For more information, see “Assigning Alternate Text to Image Maps in FrameMaker”.
ImageLongDescByRef marker type	Specifies the path to the file that contains the long description for an image. This text is added to the `longdesc` attribute of the `img` tag in the output. Screen readers read this description when you create accessible content. For more information, see “Using Text in External Files to Assign Long Descriptions to Images in FrameMaker”.
ImageLongDescNotReq marker type	Specifies that a long description is not required for an image, which bypasses this accessibility check for the image when you create accessible content. For more information, see “Excluding Images from Accessibility Report Checks in FrameMaker”.
ImageLongDescText marker type	Specifies the long description for an image. This text is added to the `longdesc` attribute of the `img` tag in the output. Screen readers read this description when you create accessible content. For more information, see “Assigning Long Descriptions to Images in FrameMaker”.
Keywords marker type	Specifies the keywords to include in the `meta` tag for the topic. The `meta` tag improves searchability on the Web. For more information, see “Creating Meta Tag Keywords in FrameMaker”.
PageStyle marker type	Specifies the name of a page style defined in the project to apply to a topic. This marker type is an internal marker type that is not displayed in Stationery Designer. You cannot create a marker type with a different name and assign it this functionality. For more information, see “Assigning Custom Page Styles in FrameMaker”.
PassThrough	Specifies that ePublisher place the contents of the marker directly into the generated output without processing the content in any way. For example, you could use a PassThrough marker if you wanted to embed HTML code within your generated output.
Popup marker type	Specifies the start of the content to include in a popup window. The content is displayed in a popup window when you hover over the link. When you click

Marker Type	Description
	the link in some output formats, the topic where the popup text is stored, such as the glossary, is displayed. For more information, see “Using Markers to Create Popup Windows in FrameMaker”.
PopupEnd marker type	Marks the end of the content to include in a popup window. For more information, see “Using Markers to Create Popup Windows in FrameMaker”.
PopupOnly marker type	Specifies the start of the content to include in only a popup window. Browsers display the content in a popup window when you hover over or click the link. For more information, see “Using Markers to Create Popup Windows in FrameMaker”.
RubiComposite marker type	No longer supported.
SeeAlsoKeyword marker type	Specifies an internal identifier for a topic. SeeAlsoLink markers in other topics can list this identifier to create a link to this topic. Used in conjunction with a See Also paragraph format or character format. For more information, see “Creating See Also Links in FrameMaker”.
SeeAlsoLink marker type	Identifies an internal identifier from another topic to include in the list of See Also links in this topic. Used in conjunction with a See Also paragraph format or character format. For more information, see “Creating See Also Links in FrameMaker”.
SeeAlsoLinkDisplayType marker type	Specifies whether to display the target topics on a popup menu or in a window. By default, the links are displayed in the Topics Found window. To display a popup menu, set the value to `menu`. This marker type is supported only in HTML Help. For more information, see “Creating See Also Links in FrameMaker”.
SeeAlsoLinkWindowType marker type	Specifies the name of the window defined in the `.hhp` file, such as TriPane or Main, that the topic opens in when the user clicks the link. This marker type is supported only in HTML Help. For more information, see “Creating See Also Links in FrameMaker”.
TableStyle marker type	Specifies the name of a table style defined in the project to apply to a table in versions of Microsoft Word that did not support table styles. This marker type is an internal marker type that is not displayed in Stationery Designer. This marker type is supported only for Microsoft Word documents. You cannot create a marker type with a different name and assign it this functionality. For more information, see “Applying Table Styles in Word”
TableSummary marker type	Specifies an alternate text summary for a table, which is used when you create accessible content. This text is added to the `summary` attribute of the `table` tag in the output. Screen readers read this description when you create accessible content. For more information, see “Assigning Alternate Text (Summaries) to Tables in FrameMaker”.
TableSummaryNotReq marker type	Specifies that a summary is not required for a table, which bypasses this accessibility check for that table. For more information, see “Excluding Tables from Accessibility Report Checks in FrameMaker”.

Marker Type	Description
TOCIconHTMLHelp marker type	Identifies the image to use as the table of contents icon for a topic in the HTML Help output format. For more information, see “Customizing Table of Contents Icons in FrameMaker”.
TOCIconJavaHelp marker type	Identifies the image to use as the table of contents icon for a topic in the Sun JavaHelp output format. For more information, see “Customizing Table of Contents Icons in FrameMaker”.
TOCIconOracleHelp marker type	Identifies the image to use as the table of contents icon for a topic in the Oracle Help output format. For more information, see “Customizing Table of Contents Icons in FrameMaker”.
TOCIconWWHelp marker type	Identifies the image to use as the table of contents icon for a topic in the WebWorks Help output format. For more information, see “Customizing Table of Contents Icons in FrameMaker”.
TopicAlias marker type	Specifies an internal identifier for a topic that can be used to create a context-sensitive link to that topic. For more information, see “Creating Context-Sensitive Help in FrameMaker”.
TopicDescription marker type	Specifies a topic description for a context-sensitive help topic in Eclipse help systems. For more information, see “Specifying Context-Sensitive Help Links in FrameMaker”.
WhatIsThisID marker type	Identifies a What’s This help internal identifier for creating context-sensitive What’s This field-level help for Microsoft HTML Help. For more information, see “Opening Topics in Custom Windows in FrameMaker”.
WindowType marker type	Specifies the name of the window defined in the Help project that the topic should be displayed in. In Microsoft HTML Help, the window names are defined in the `.hhp` file. This marker type is supported in Microsoft HTML Help and Oracle Help. For more information, see “Opening Topics in Custom Windows in FrameMaker”.

Paragraph and Character Formats in FrameMaker

ePublisher projects use the paragraph formats and character formats defined by the Stationery designer to implement online features when generating output. Before you begin using paragraph formats and character formats to implement online features, talk to the Stationery designer and verify which online features your Stationery supports. Your Stationery only recognizes the paragraph formats and character formats defined by the Stationery designer in your Stationery. If you try to implement online features using paragraph formats and character formats not supported in your Stationery, ePublisher does not recognize these items when generating output.

When the Stationery designer creates the Stationery, the Stationery designer specifies the names of paragraph format and character formats used to implement an online feature. Consult with the Stationery designer to obtain the names of the paragraph formats and character formats defined by the Stationery designer to support each online feature you want to implement.

The following table lists the default names of paragraph formats and character formats used to implement online features. Always verify with the Stationery designer the names of the paragraph formats and character formats you should use when implementing online features before you use these items in your source documents.

Format	Description
AbbreviationTitle character format	Specifies abbreviation alternate text for browsers to display for abbreviations such as SS# when a user hovers over the abbreviation in output. Screen readers also can read the abbreviation alternate text. Used in combination with the AbbreviationTitle marker type. For more information, see “Assigning Alternate Text to Abbreviations in FrameMaker”.
AcronymTitle character format	Specifies acronym alternate text for browsers to display for acronyms such as HTML when a user hovers over the acronym in output. Screen readers can also read the acronym alternate text. Used in combination with the AcronymTitle marker type. For more information, see “Assigning Alternate Text to Acronyms in FrameMaker”.
Citation character format	Specifies the source of a quote using a fully qualified Uniform Resource Identifier (URI) when a user hovers over the quote in output. Screen readers can also read the URI for the quote. Used in combination with the Citation marker type. For more information, see “Providing Citations for Quotes in FrameMaker”.
Expand/Collapse paragraph format	Specifies the content you want to include in an expand/collapse section. Used in conjunction with a DropDownEnd marker type. For more information, see “Creating Expand/Collapse Sections (Drop-Down Hotspots) in FrameMaker”.
Popup paragraph format	Specifies the popup content to display in both a popup window and in a standard help topic. Applied to the first paragraph of popup content. For more information, see “Using Paragraph Formats to Create Popup Windows in FrameMaker”.

Format	Description
Popup Append paragraph format	Specifies the popup content to display in a popup window and in a standard help topic. Applied to additional popup paragraphs when you have more than one paragraph of popup content. For more information, see “Using Paragraph Formats to Create Popup Windows in FrameMaker”.
Popup Only paragraph format	Specifies the popup content to display in only a popup window. Applied to the first paragraph of popup content. For more information, see “Using Paragraph Formats to Create Popup Windows in FrameMaker”.
Popup Only Append paragraph format	Specifies the popup content to display in only a popup window. Applied to additional popup paragraphs when you have more than one paragraph of popup content. For more information, see “Using Paragraph Formats to Create Popup Windows in FrameMaker”.
Related Topic paragraph format	Specifies related topics links. For more information, see “Creating Related Topics in FrameMaker”.
See Also character format	Specifies the text you want to include in a See Also button. For more information, see “Creating See Also Links in FrameMaker”.
See Also paragraph format	Specifies the text you want to include in a See Also inline text link. For more information, see “Creating See Also Links in FrameMaker”.

Obtaining and Applying the Latest Adobe FrameMaker Template

An efficient, effective, and consistent ePublisher online content generation process relies upon the use of templates. Templates define marker types and paragraph, character, and table formats. Templates may also contain standard conditions, variables, and cross-reference definitions that you can use when creating and working with source documents used to generate online content. Templates help control the look and feel of source documents and generated output across multiple writers, multiple projects, and multiple types of generated output.

The ePublisher content generation process assumes that you use marker types and paragraph, character and table formats defined in an Adobe FrameMaker template prepared by a Stationery designer as you create content and format your source documents. Using Adobe FrameMaker templates and the marker types and paragraph, character, and table formats and other layout formats and characteristics defined in templates ensures that you format content in your source documents consistently and also ensures ePublisher can use your source documents effectively to generate output.

If your source documents do not use templates or do not use the same marker types, formats, and standards defined in your Stationery by the Stationery designer, your generated output may not conforms to the styles and standards defined by the Stationery designer for output. You may also not be able to implement some online features if you do not use the correct templates or the correct marker types and formats defined in the templates.

As a part of preparing your Adobe FrameMaker source documents for output generation, ensure your source documents use the correct Adobe FrameMaker templates from the Stationery designer and you have applied all paragraph, character, and table formats specified in the template correctly.

Importing Custom Marker Types in FrameMaker

Typically the Stationery designer defines custom marker types supported in your ePublisher Stationery in an Adobe FrameMaker template. You then import the custom marker types defined in an Adobe FrameMaker template into your Adobe FrameMaker source documents by importing document properties from the Adobe FrameMaker template. This procedure explains how to import custom marker types from an Adobe FrameMaker template. For more information about creating custom marker types, see “Creating Custom Marker Types in FrameMaker”.

The following procedure provides an example of how to import custom marker types into Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for importing custom marker types in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To import custom marker types from an Adobe FrameMaker template into your source documents

1. In your Adobe FrameMaker source document, on the **File** menu, click I**mport > Formats**.
2. In the **Import from Document** field, select the Adobe FrameMaker template that contains the custom marker types you want to import from the list.
3. In the **Import and Update** field, select only the **Document Properties** check box.
4. Click **Import**.
5. Click **OK** to confirm the operation.

Creating Custom Marker Types in FrameMaker

Typically you should not need to create a custom marker type in an Adobe FrameMaker source document. If you want to use a custom marker type to implement an online feature, use the custom marker type provided in the Adobe FrameMaker template you use for your source documents. If you do not see a custom marker type you want to use to implement an online feature in the Adobe FrameMaker template, verify with the Stationery designer that your Stationery supports the custom marker type before you insert and use the custom marker in a source document.

Occasionally your Stationery may support a custom marker type that is not defined in the Adobe FrameMaker template you use with your source documents. In this situation, first confirm with the Stationery designer that your Stationery supports the custom marker type. After confirming your project supports the custom marker type, you can create the custom marker type in your Adobe FrameMaker source document.

The following procedure provides an example of how to create custom marker types in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for creating custom marker types in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create a custom marker type in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, on the **Special** menu, click **Marker**.
2. In the **Marker Type** field, select **Edit** from the drop-down list.
3. Type `CustomMarkerTypeName` to create a custom marker type, where `CustomMarkerTypeName` is the name of the custom marker type you want to create.

   Note: The custom marker type name you type must match the name of the custom marker type supported in your ePublisher Stationery. If you specify a name for the custom marker type that is different than the name of the custom marker type supported in your ePublisher Stationery, ePublisher will not be able to recognize and use the custom marker type when generating output.
4. Click **Add**.
5. Click **OK** to confirm the operation.
6. Click **Done**. Adobe FrameMaker displays the custom marker type you created in the Marker window in the **Marker Type** field.

Creating a Passthrough Marker in FrameMaker

A passthrough marker is a marker that allows you to insert content that you do not want ePublisher to process when you generate output. For example, if you have embedded multimedia files in your source documents, such as Audio Video Interleave files (`.avi`) or Adobe Software Flash files (`.swf`), you can insert a passthrough marker with a value that is set to the HTML code that you do not want ePublisher to process.

The following example shows `.avi` code to which you could insert using a passthrough marker.

```
<embed src="sample.avi" width="400"
 height="300" pluginspage="">
</embed>
```

To create a passthrough marker in an Adobe FrameMaker source document

1. In Adobe FrameMaker, on the **Special** menu, click **Marker**.
2. In the **Marker Type** field, select **Passthrough** from the drop-down list.
3. ***If the Passthrough marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.
4. In the **Marker Text** field, type the html code that you would like to not be processed by ePublisher such as the Flash embed code indicated in the previous topic.
5. Click **New Marker**.
6. Save your source document.
7. Generate output for your project. For more information, see “Generating Output”.
8. In Output Explorer, verify ePublisher created the appropriate result for your embedded html code. For more information, see “Viewing Output in Output Explorer”.

Creating Cross-References and Links in FrameMaker

When you generate output, ePublisher automatically converts all cross-references in your Adobe FrameMaker source documents to links. Typically Stationery designers specify in Stationery how cross-references should display in generated output. For example, Stationery designers typically specify that cross-references that contain page numbers in source documents display without page numbers in generated output, as page numbers are out of context in online output. If you have target setting permissions, you can also customize the cross-reference formats you want to use when you generate output. For more information about customizing cross-reference settings, see "Setting Cross-References in Projects".

Including cross-references in Adobe FrameMaker source documents is typically the easiest way to produce links in online content. However, in some cases you may not be able to achieve the effect you want by creating links using cross-references. In these cases, you can insert native Adobe FrameMaker Hypertext markers that use the `gotolink` and `message URL` hypertext commands in your Adobe FrameMaker source documents and use the Hypertext markers to create the links you want.

To create a link using a cross-reference or Hypertext markers in an Adobe FrameMaker source document

1. ***If you want to create a link using a cross-reference,*** complete the following steps:
    a. In your Adobe FrameMaker source document, select the text for which you want to create a link.
    b. On the **Special** menu, click **Cross-Reference**.
    c. In the **Document** field, select the document that contains the content to which you want to link.
    d. In the **Paragraph Tags** field, select the paragraph tag used for the content to which you want to link.
    e. In the **Paragraphs** field, select the paragraph to which you want to link.
    f. In the **Format** field, select the appropriate format for the link. For example, if you are creating a link to a glossary term, select a glossary term cross-reference format.
    g. Click **Replace**.
2. ***If you want to create a link using hypertext markers***, complete the following steps:
    a. In your Adobe FrameMaker source document, insert your cursor in front of the link target text.
    b. On the **Special** menu, click **Marker**.
    c. In the **Marker Type** field, select **Hypertext** from the list.
    d. In the **Marker Text** field, type `newlink` *`linkname`* or `newlink` *`filename:linkname`*, where *`linkname`* is the name of the named destination for the link, and *`filename`* is the name of the file that contains the link, if the link is in a different Adobe FrameMaker source

document. To make maintenance easy, create short link names that use alphanumeric, lowercase characters.

e. Click **New Marker**.

f. Insert your cursor in front of the word or phrase for which you want to create a link.

a. On the **Special** menu, click **Marker**.

b. In the **Marker Type** field, select **Hypertext** from the list.

c. In the **Marker Text** field, type `gotolink` *`linkname`* or `gotolink` *`filename:linkname`*, where *`linkname`* is the name of the named destination you created for the link, and *`filename`* is the name of the file that contains the link, if the link is in a different Adobe FrameMaker source document.

d. Click **New Marker**.

e. Select the word or phrase for which you want to create a link. The selected area must contain the both text and the hypertext marker you created.

f. Apply a link character format to the word or phrase. Applying a link character format to the word or phrase makes the link appear active, or clickable, in the generated output. If you do not know which character format to use for links, consult the Stationery designer.

3. ***If you want to create a link to a PDF file***, complete the following steps:

a. In the FrameMaker menu, go to **Special > Hypertext**

b. From the Command dropdown menu, chose **Message Client**

c. In the Syntax text box, type message `openfile relative_path` where `relative_path` is the relative directory where you have your PDF located and then you add `example.pdf` to this path

4. Save your Adobe FrameMaker source document.

Working with Tables in FrameMaker

This section explains how to prepare tables in source documents for output generation. Obtain your latest templates and apply the latest table formats from the template to tables in your source documents. If your tables do not have header rows, create a header row for each table. If your tables do not have footer rows, create a footer row for each table.

Applying Table Formats in FrameMaker

Table formats define the appearance of your tables, and ePublisher uses table formats to define the appearance of tables in generated output. When you work with tables in your Adobe FrameMaker source documents, ensure you apply the correct table formats to your tables. The Stationery designer defines the table formats you can use in your Adobe FrameMaker source documents in the Adobe FrameMaker templates you associate with your Adobe FrameMaker source documents. If you want to specify a different table format for sets of tables in your generated output, first ensure the different table format you want to apply is available in your Adobe FrameMaker source document. Then apply the different table format to tables in your Adobe FrameMaker source documents as appropriate.

For example, you may have a small set of tables that contain information about a specific component in a product. If you decide you want to modify the appearance of these tables in your generated output by specifying that the tables associated with this component display with a yellow background in your generated output, apply a table format available in your Adobe FrameMaker source document that the Stationery designer created to meet this requirement. When you generate output, the Stationery designed by the Stationery designer specifies that any tables created with a table format configured to display tables with a yellow background display in your output with a yellow background.

Creating Table Header Rows in FrameMaker

Most tables in Adobe FrameMaker source documents include header rows, because by default Adobe FrameMaker allows you to quickly and easily specify the number of header rows in a table when you create a table. However, if your tables do not have header rows, consider adding table header rows to tables in your Adobe FrameMaker source documents. Using table header rows allows you to more tightly control the appearance of tables when you generate output. For example, if you use header rows, you can specify one appearance for header rows in your generated output, and a difference appearance for body rows in your generated output.

The following procedure provides an example of how to create table header rows in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for creating table header rows in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create a table header row in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the table for which you want to create a table header row.
2. Insert your cursor in the top row of the table.
3. On the **Table** menu, click **Add Rows or Columns**.
4. Click **Add 1 Row**.
5. Select **To Heading** from the list.
6. Click **Add**. Adobe FrameMaker inserts a header row into the table.
7. Type the text for the header row into the table.
8. Delete any existing rows in the text that contain the text you typed into the new table header row as needed.

Creating Table Footer Rows in FrameMaker

Most tables in Adobe FrameMaker source documents include footer rows, because by default Adobe FrameMaker allows you to quickly and easily specify the number of footer rows in a table when you create a table. However, if your tables do not have footer rows, consider creating footer rows in your source documents in order to quickly and easily specify the appearance that you want for your table footer rows in your generated output. For example, if you use footer rows in conjunction with header rows, you can specify one appearance for footer rows in your generated output, and then different appearances for header rows and table body rows in your generated output.

The following procedure provides an example of how to create table footer rows in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for creating table footer rows in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create a table with a footer row in Adobe FrameMaker

1. In your Adobe FrameMaker source document, locate the table for which you want to create a table footer row.
2. Insert your cursor in the bottom row of the table.
3. On the **Table** menu, click **Add Rows or Columns**.
4. Click **Add 1 Row**.
5. Select **To Footing** from the list.
6. Click **Add**. Adobe FrameMaker inserts a footer row into the table.

Working with Images in FrameMaker

Many writers include images when producing documents using Adobe FrameMaker. Most writers typically insert images into Adobe FrameMaker source documents in one of the following ways:

- Copying images directly into in Adobe FrameMaker source documents, also known as embedding images
- Importing images by reference, which creates a link to the source image in the Adobe FrameMaker source documents

If you copy an image into an Adobe FrameMaker source document, Adobe FrameMaker copies, or embeds, the image in the Adobe FrameMaker source document, and the image becomes a part of the document.

If you import an image by reference in Adobe FrameMaker source documents, Adobe FrameMaker creates a link to the image and displays the image in the Adobe FrameMaker source document. The link becomes a part of the document, but the actual image file is not inserted into the document, although the actual image files is displayed in the document. If you update the image file referenced by the link, Adobe FrameMaker displays the updated image referenced by the link automatically.

There are benefits and drawbacks to copying images directly into Adobe FrameMaker source documents and importing images by reference.

For example, if you copy images into Adobe FrameMaker source documents, you do not have worry about breaking the reference, or link, between the Adobe FrameMaker source documents and the image files. If you import the image by reference into Adobe FrameMaker source documents, you must ensure that you keep the same file structure for the image files in order to not break the references, or links, between the Adobe FrameMaker source document and the image file.

However, importing images by reference in Adobe FrameMaker source documents, rather than copying images into the source documents, provides the following benefits:

- You can update image files without recopying the image into your Adobe FrameMaker source documents.
- If you have one image used in multiple places, you can update the image in one place, rather than recopying the image into multiple places.
- You can manage your documentation files and image files separately, which makes organizing images easier.
- Source documents with images imported by reference in Adobe FrameMaker are smaller than source documents with copied images.

When you work with Adobe FrameMaker source documents that you will use to generate output, ensure you follow the guidelines specified by the Stationery designer for the following items:

- Method used to insert images

- Correct DPI to use for inserted images
- Correct image file format to use for inserted images

Inserting Images in FrameMaker

Before you insert images into Adobe FrameMaker source documents you plan to use to generate output, review image considerations. For more information, see “Working with Images in FrameMaker”.

When you insert images into Adobe FrameMaker source documents, insert the image into an anchored frame. The anchored frame allows you to specify the image alignment and position. For more information about anchored frame options, see the Adobe FrameMaker documentation.

The following procedure provides an example of how to use an anchored frame to insert an image by reference in an unstructured Adobe FrameMaker source document using Adobe FrameMaker 7.2. Steps for inserting an image by reference in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To insert an image in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, insert your cursor on a blank line below the paragraph where you want to insert your image.

   Note: Inserting an image on a blank line allows you to customize the paragraph tag applied to the line. Many Adobe FrameMaker templates have a special paragraph tag for you to use when you insert graphics. This paragraph tag specifies the space above and below the paragraph and the alignment of the inserted image.

2. Apply the appropriate paragraph format for images to the blank anchored frame line. For more information about the correct paragraph format to use for image anchored frame lines, consult with the Stationery designer.

3. On the **Special** menu, click **Anchored Frame**.

4. Specify the position, alignment, and size of the anchored frame, and then click **New Frame**. Adobe FrameMaker inserts an empty anchored frame into the source document.

   For more information about anchored frame options, see the Adobe FrameMaker documentation.

5. On the **File** menu, click **Import > File**.

6. Click **OK** to continue.

7. Browse to the location of the file you want to import and select the file.

8. Click **Import by Reference**, and then click **Import**.

9. Specify the size of the graphic.

   Note: Most writers do not select the **Fit in Selected Rectangle** option. This option resizes the image to fit inside the selected anchor or graphic frame. When you select this option, Adobe FrameMaker sets the DPI to unknown, and the imported image is usually distorted.

   - ***If you want to use the DPI from the graphic,*** do not change the setting in the **Custom dpi** field. The number in the **Custom dpi** field is the DPI of the imported graphic.

- ***If you want to change the size of the graphic***, click the button for the dpi setting you want to specify.

  Note: If you do not use the same DPI setting as the source image, the image in your output may be distorted.

10. Click **Set**. Adobe FrameMaker imports the image into the source document.
11. ***If you have white space between the graphic and the anchored frame***, you can shrink-wrap the frame by completing the following steps:

    Shrinking-wrapping an anchored frame removes the white space between the graphic and the anchored frame and changes the anchoring position of the frame to **At Insertion Point** and displays the frame 0 points above the baseline of the text. If the anchored frame is on the same line as the text, the 0 point baseline can cause the image to cover the text of the preceding lines. For this reason, many writers prefer to insert images on a separate line below the text. The image may also be distorted if you don't shrink wrap the image.

    a. Click the anchored frame or the image in the anchored frame.

    b. Press `ESC+M+P`. Adobe FrameMaker shrinks or expands the anchored frame to fit the contents of the anchored frame and positions the anchored frame according to the paragraph pagination settings.

After you insert an image, you can assign alternate text or a long description to the image. For more information, see "Assigning Alternate Text to Images and Image Maps in FrameMaker" and "Assigning Long Descriptions to Images in FrameMaker".

Creating Image Links in FrameMaker

You can create image links that allow users who click the image to link to content in another location. For example, if you include your company logo in a source document, you can define a link for the logo so that when users click the logo, they link to your company home page.

The following procedure provides an example of how to create an image link in Adobe FrameMaker source documents using Adobe FrameMaker 7.2. Steps for creating an image link in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create an image link in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, insert the image for which you want to create an image link. For more information, see "Inserting Images in FrameMaker".
2. ***If you want to link to content in a different location in your source document***, create a named destination for the link by completing the following steps:

   Note: You do not need to perform these steps if you want to link to content on a Web site.

   a. Locate the link target in your source document.

   b. On the **Special** menu, click **Marker**.

   c. In the **Marker Type** field, click **Hypertext**.

   d. In the **Marker Text** field, `newlink` *`linkname`* or `newlink` *`filename:linkname`*, where *`linkname`* is the name of the named destination for the link, and *`filename`* is the name of the file that contains the link, if the link is in a different Adobe FrameMaker source document.

      Note: To make maintenance easy, create short link names that use alphanumeric, lowercase characters.

   e. Click **New Marker**.
3. In the anchored frame that contains the image for which you want to create a link, draw a text frame that covers the entire clickable region by completing the following steps:

   a. On the **Graphics** menu, click **Tools** to display the graphic tools palette.

   b. Click the **Text Frame** icon.

   c. Drag the cursor across the image to draw a text frame over the image.

   d. In the Create New Text Frame window, in the **Number** field, type `1`, and then click **Set**.

   e. Click outside of the image, and then insert your cursor in the text frame.
4. In the text frame, insert a Hypertext marker that specifies the destination of the link by completing the following steps:

   a. On the **Special** menu, click **Marker**.

b. In the **Marker Type** field, select **Hypertext** from the list.

c. ***If you want link to content in a different location in your source document***, use the named destination link you created in step 2. In the **Marker Text** field, type `gotolink` *`linkname`* or `gotolink` *`filename:linkname`*, where *`linkname`* is the name of a link target you created previously, and *`filename`* is the name of the file that contains the link, if the link is in a different Adobe FrameMaker source document.

d. ***If you want to link to a page on a Web site,*** in the **Marker Field**, type `message URL` *`web address`*, where *`web address`* is the URL of the web page you want to open when users click the image.

Note: For more information about using the `gotolink` and `message URL` commands, see the Adobe FrameMaker documentation.

e. Click **New Marker**.

5. Save your Adobe FrameMaker source document.

6. Generate output for your project. For more information, see “Generating Output”.

7. In Output Explorer, verify ePublisher created the image link using the link information you specified on the page by clicking on the image. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Creating Clickable Regions for Image Maps in FrameMaker

An image map can be a single image separated with clickable regions or a composite image made up of multiple images grouped together, yet still separated with clickable regions. For example, you could create an image of the countries of Europe and then define an image map for the image that allows users to link to a topic about each country when they click on an area of the image. User can click France to see information about France, Italy to see information about Italy, and so on.

When you define an image map, you can also define alternate text for each clickable region. For example, you might define alternate text for the Italy region as “Click here for more information about Italy.” For more information about assigning alternate text to image maps, see “Assigning Alternate Text to Images and Image Maps in FrameMaker”.

Creating Image Maps for Single Images in FrameMaker

You create image maps for single images in Adobe FrameMaker source documents using text frames and hyperlinks. In Adobe FrameMaker, a hyperlink consists of a link and a link target, or named destination. A **named destination** is a unique identifier for a location in the document.

You can also create an image map for a composite image in an Adobe FrameMaker source document. For more information about creating composite images, see "Creating Image Maps for Composite Images in FrameMaker".

The following procedure provides an example of how to create an image map for a single image in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for creating an image map for a single image in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create an image map for a single image in an Adobe FrameMaker source document:

1. In your Adobe FrameMaker source document, insert the image you want to use for you image map into an anchored frame. For more information, see "Inserting Images in FrameMaker".
2. ***If you want to link to content in a different location in your source document***, create a named destination for the link for each area of the image map by completing the following steps:

   Note: You do not need to perform these steps if you want to link to content on a Web site.

   a. Locate the link target in your source document.

   b. On the **Special** menu, click **Marker**.

   c. In the **Marker Type** field, click **Hypertext**.

   d. In the **Marker Text** field, `newlink` *`linkname`* or `newlink` *`filename:linkname`*, where *`linkname`* is the name of the named destination for the link, and *`filename`* is the name of the file that contains the link, if the link is in a different Adobe FrameMaker source document.

      Note: To make maintenance easy, create short link names that use alphanumeric, lowercase characters.

   e. Click **New Marker**.
3. In the anchored frame that contains the image for which you want to create an image map, draw a text frame that covers each region of the image where you want users to be able to click by completing the following steps:

   a. On the **Graphics** menu, click **Tools** to display the graphic tools palette.

   b. Click the **Text Frame** icon.

   c. Drag the cursor over the portion of the image for which you want to create a clickable area.

   d. In the Create New Text Frame window, in the **Number** field, type `1`, and then click **Set**.

4. In the text frame, insert a Hypertext marker that specifies the destination for the clickable region by completing the following steps:

    a. Insert your cursor into the text frame.

    b. On the **Special** menu, click **New Marker**.

    c. ***If you want to link to content in a different location in your source document***, use a named destination link you created in step 2. In the **Marker Text** field, type `gotolink` *`linkname`* or `gotolink` *`filename:linkname`*, where *`linkname`* is the name of a link target you created previously, and *`filename`* is the name of the file that contains the link, if the link is in a different Adobe FrameMaker source document.

    d. ***If you want to link to a page on a Web site***, in the **Marker Text** field, type `message URL` *`web address`*, where *web address* is the URL of the web page you want to open when users click the image.

        Note: For more information about using the these commands, see the Adobe FrameMaker documentation.

    e. Click **New Marker**.

5. Save your Adobe FrameMaker source document.

6. Generate output for your project. For more information, see “Generating Output”.

7. In Output Explorer, verify ePublisher created the image map using the link information you specified by clicking on the page that contains the image map and then clicking on each area of the image where you created a link. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Creating Image Maps for Composite Images in FrameMaker

You can create composite images by inserting the composite images into an anchored frame and then inserting text frames that contain the link you want users to go to when they click an area of a composite image.

The following procedure provides an example of how to use Hotspots in Adobe FrameMaker to create image maps for composite images in source documents.

To create an image map for a composite image using Hotspots in an Adobe FrameMaker source document:

1. In your Adobe FrameMaker source document, insert each image you want to use for your image map into an anchored frame. For more information, see "Inserting Images in FrameMaker".
2. ***If you want to link to content in a different location in your source document***, create a named destination for the link for each area of the image map by completing the following steps:

   Note: You do not need to perform these steps if you want to link to content on a Web site.

   a. Locate the link target in your source document.

   b. On the **Special** menu, click **Marker**.

   c. In the **Marker Type** field, click **Hypertext**.

   d. In the **Marker Text** field, `newlink` *`linkname`* or `newlink` *`filename:linkname`*, where *`linkname`* is the name of the named destination for the link, and *`filename`* is the name of the file that contains the link, if the link is in a different Adobe FrameMaker source document.

      Note: To make maintenance easy, create short link names that use alphanumeric, lowercase characters.

   e. Click **New Marker**.
3. In the anchored frame that contains the image for which you want to create an image map, draw a text frame that covers each region of the image where you want users to be able to click by completing the following steps:

   a. On the **Graphics** menu, click **Tools** to display the graphic tools palette.

   b. Click the **Text Frame** icon.

   c. Drag the cursor over the portion of the image for which you want to create a clickable area.

   d. In the Create New Text Frame window, in the **Number** field, type `1`, and then click **Set**.
4. For each text frame, set the Hotspot properties that specifies the destination for the clickable region and optionally set its tooltip text by completing the following steps:

   a. On the **Graphics** menu, click **Tools** to display the graphic tools palette.

b. Click the **Select Object** icon.

c. Click on the text frame so that you can set its hotspot properties.

d. With the text frame selected, right-click and select **Hotspot Properties...** from the menu.

e. ***If you want to link to content in a different location in your source document***, under **Destination** select the **Document** radio button. Then select either **Current** or the name of your target source document. Then click the marker with the named destination you created previously in step 2.

f. ***If you want to link to a page on a Web site***, under Destination select the URL radio button. Then type the complete web address (including protocol such as http://) in the adjacent text box of the web page you want to open when users click the image.

g. ***If you want tooltip text to be available in the output***, under the **Tooltip Text** label, enter the text you want to appear as the tooltip.

h. Click the **Save** button on the **Hotspot** dialog.

5. Save your Adobe FrameMaker source document.

6. Generate output for your project. For more information, see "Generating Output".

7. In Output Explorer, verify ePublisher created the image map using the link information you specified by clicking on the page that contains the image map and then clicking on each area of the image where you created a link. For more information about viewing output files in Output Explorer, see "Viewing Output in Output Explorer".

If your version of Adobe FrameMaker does not provide for Hotspot properties, you can still create image maps for composite images by following this alternate example of how to create image maps for composite images in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for creating image maps for composite images in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create an image map for a composite image in an Adobe FrameMaker source document using hypertext markers:

1. In your Adobe FrameMaker source document, insert each image you want to use for your image map into an anchored frame. For more information, see "Inserting Images in FrameMaker".

2. ***If you want to link to content in a different location in your source document***, create a named destination for the link for each area of the image map by completing the following steps:

Note: You do not need to perform these steps if you want to link to content on a Web site.

a. Locate the link target in your source document.

b. On the **Special** menu, click **Marker**.

c. In the **Marker Type** field, click **Hypertext**.

d. In the **Marker Text** field, `newlink` *`linkname`* or `newlink` *`filename:linkname`*, where *`linkname`* is the name of the named destination for the link, and *`filename`* is the name

of the file that contains the link, if the link is in a different Adobe FrameMaker source document.

Note: To make maintenance easy, create short link names that use alphanumeric, lowercase characters.

e. Click **New Marker**.

3. In the anchored frame that contains the image for which you want to create an image map, draw a text frame that covers each region of the image where you want users to be able to click by completing the following steps:

a. On the **Graphics** menu, click **Tools** to display the graphic tools palette.

b. Click the **Text Frame** icon.

c. Drag the cursor over the portion of the image for which you want to create a clickable area.

d. In the Create New Text Frame window, in the **Number** field, type `1`, and then click **Set**.

4. In each text frame, insert a Hypertext marker that specifies the destination for the clickable region by completing the following steps:

a. Insert your cursor into the text frame.

b. On the **Special** menu, click **Marker**.

c. ***If you want to link to content in a different location in your source document***, use the named destination link you created in step 2. In the **Marker Text** field, type `gotolink` *`linkname`* or `gotolink` *`filename:linkname`*, where *`linkname`* is the name of a link target you created previously, and *`filename`* is the name of the file that contains the link, if the link is in a different Adobe FrameMaker source document.

d. ***If you want to link to a page on a Web site***, in the **Marker Text** field, type `message URL` *`web address`*, where *web address* is the URL of the web page you want to open when users click the image.

Note: For more information about using the these commands, see the Adobe FrameMaker documentation.

e. Click **New Marker**.

5. Save your Adobe FrameMaker source document.

6. Generate output for your project. For more information, see “Generating Output”.

7. In Output Explorer, verify ePublisher created the image map using the link information you specified by clicking on the page that contains the image map and then clicking on each area of the image where you created a link. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Assigning Image Scales in FrameMaker

When ePublisher converts images inserted into your source documents, it can scale images to make them display larger or smaller in your generated output. By default, ePublisher uses the scaling factor applied to images as specified by the image format you apply to each image. For example, if you apply an image format to images and the Stationery designer defined the image format to scale images to 80% of their original size, all images that have this image format applied to them will be scaled to 80% in the generated output.

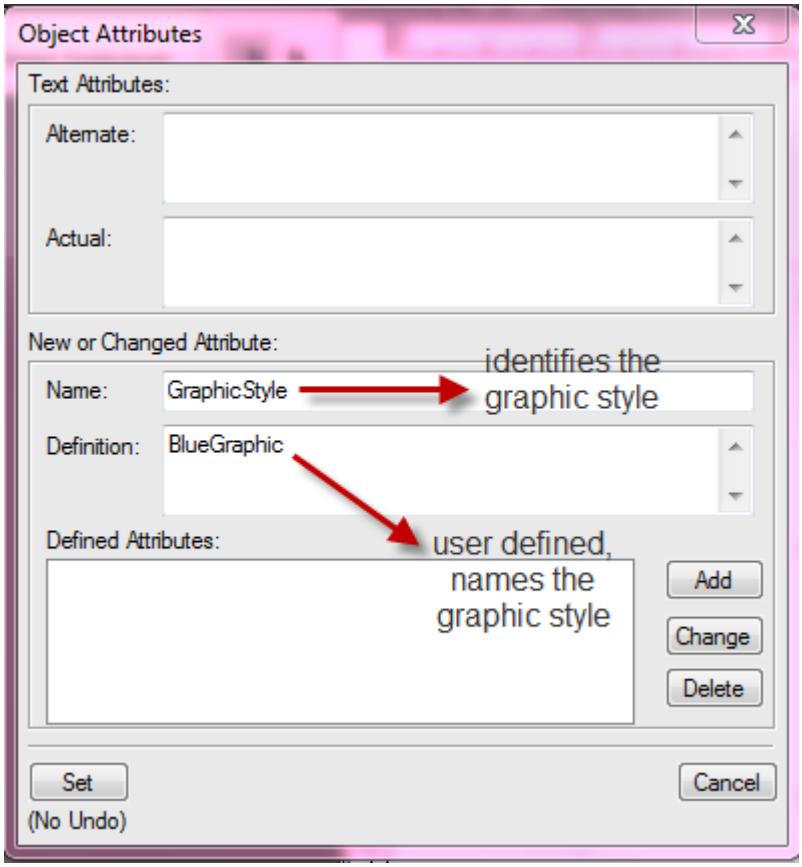
Typically, using the standard scaling factor specified in the image format is sufficient. Occasionally, however you may want to override the scaling factor for an individual image. For example, while most `.gif` images scale to 80%, you may have one large image that you want scaled to 60% in your generated output. You can manually override the standard scaling factor specified in your Stationery for a specific image by using the GraphicScale marker.

To assign a scale to a specific image, your Stationery and template must have the GraphicScale marker type configured. Your output format must also support scaling by image.

The following procedure provides an example of how to specify image scaling for an image in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for specifying image scaling for an image in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To specify an image scale for an image in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the anchored frame for the image for which you want to specify image scaling.
2. Right-click on the image anchor. Make sure this is the entire anchor, not just the graphic itself
3. Click on **Object Properties** and then click **Object Attributes**. Identify the **GraphicScale** according to the box below. Assign the desired style name in the attribute value box.



4. Click **Add**, then **Set**, and then **Set**. Adobe FrameMaker may prompt you to approve the change as the operation cannot be undone

5. Save your Adobe FrameMaker source document.

6. Scan this document in ePublisher so that the **GraphicScale** marker will show up under "Marker Styles". This will configure the correct marker behavior for processing.

7. Generate output for your project. For more information, see "Generating Output".

8. In Output Explorer, verify ePublisher created the image using the image scale you specified in the **GraphicScale** marker by clicking on the page that contains the image for which you specified image scaling. For more information about viewing output files in Output Explorer, see "Viewing Output in Output Explorer".

Assigning Image Styles in FrameMaker

Typically you do not need to specify an image style for images when you generate output. By default, each image generated by ePublisher is associated with the default image style defined in the Stationery. However, if you want to change the image style of one image or a small set of images, you can specify the image style you want to use for an image in your source document using the GraphicStyle marker type.

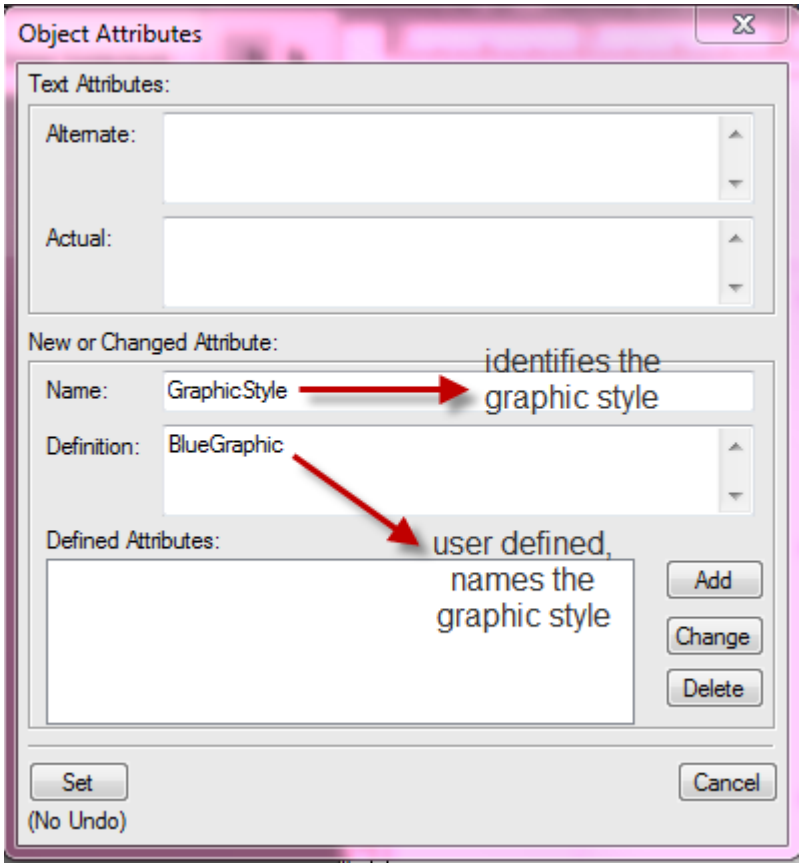
For example, if you want to specify a yellow border around a set of screen shot images that illustrate a particular piece of product functionality, you can specify that each of the screen shots images in the set have a yellow border around them through the use of the GraphicStyle marker type.

To assign a style to a specific image, your Stationery and FrameMaker template must have the GraphicStyle marker type configured. Your output format must also support specifying image styles.

The following procedure provides an example of how to specify image styles for images in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for specifying image styles for images in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To specify an image style for an image in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the anchored frame for the image for which you want to specify an image style.
2. Right-click on the image anchor. Make sure this is the entire anchor, not just the graphic itself
3. Click on **Object Properties** and then click **Object Attributes**. Identify the **GraphicStyle** according to the box below. Assign the desired style name in the attribute value box.



4. Click **Add**, then **Set**, and then **Set**. Adobe FrameMaker may prompt you to approve the change as the operation cannot be undone
5. Save your Adobe FrameMaker source document.
6. Scan this document in ePublisher so that the **GraphicStyle** marker will show up under "Graphic Styles"
7. Generate output for your project. For more information, see "Generating Output".
8. In Output Explorer, verify ePublisher created the image using the image style you specified by clicking on the page that contains the image for which you specified an image style and verifying ePublisher applied the image style you specified in the generated output. For more information about viewing output files in Output Explorer, see "Viewing Output in Output Explorer".

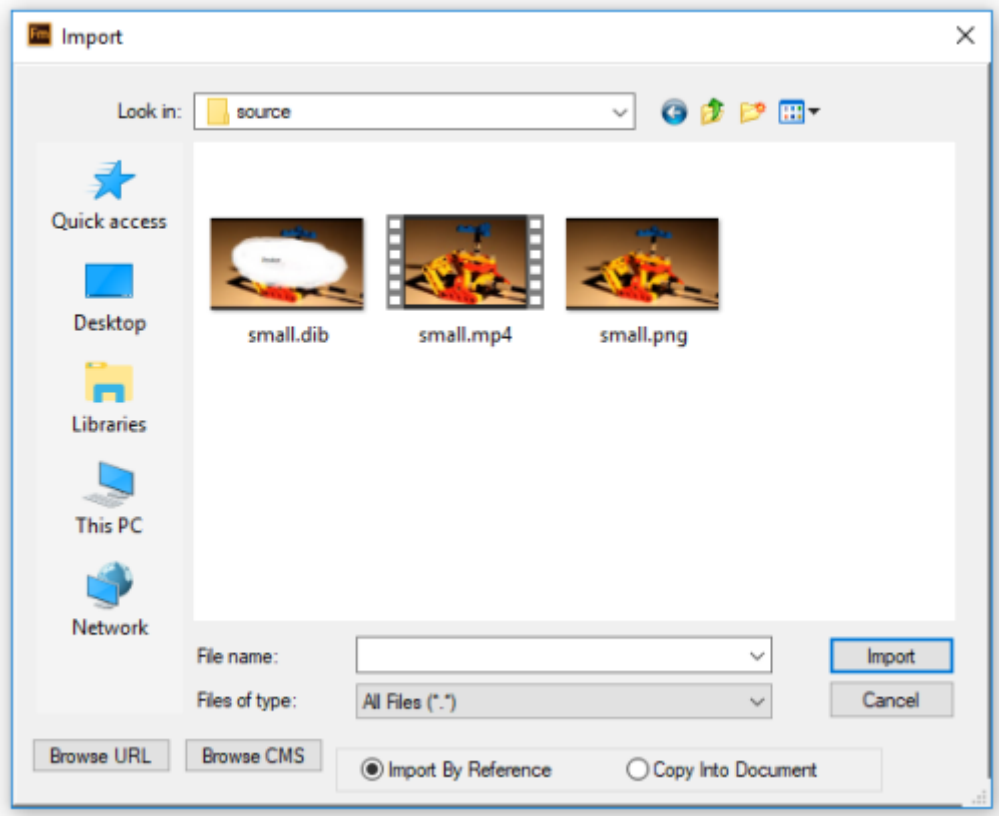
Working with Videos in FrameMaker

Occasionally, writers include videos when producing documents using Adobe FrameMaker.

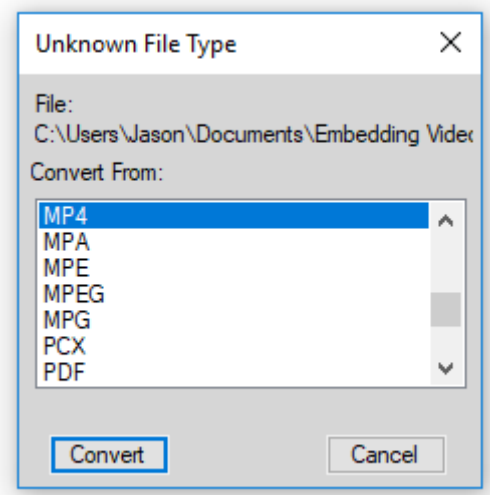
Note: If the video is not found, ensure the correct path to the video is specified in the FrameMaker document.

To include a video in an Adobe FrameMaker source document

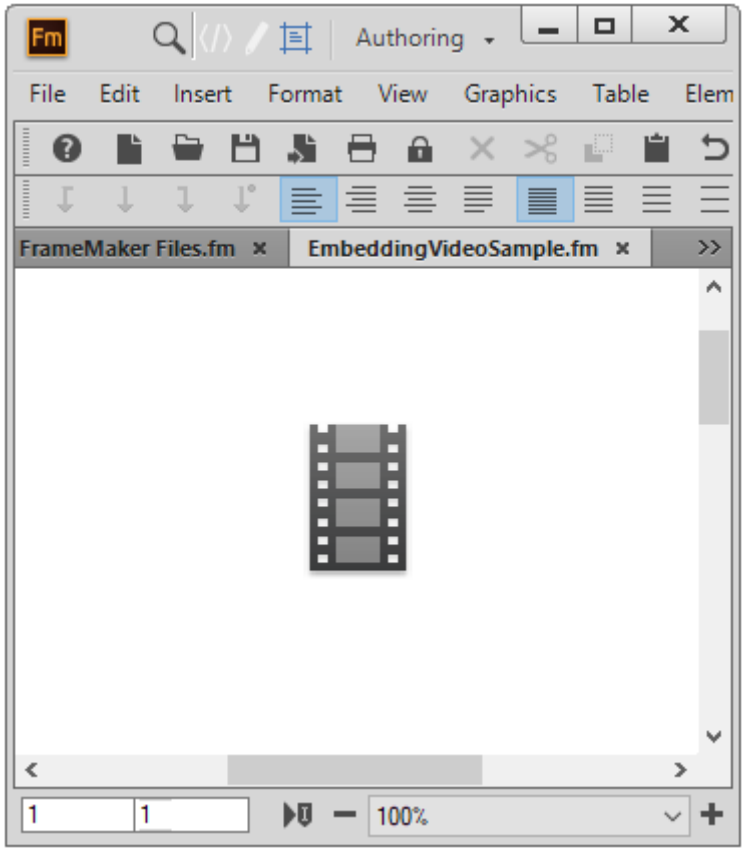
1. In your Adobe FrameMaker source document, click on File from the tool bar, select Import File, then a dialog box will appear.



2. Locate the video file in the dialog box.
3. Select the video file, click the import button.
4. Select which file type to convert from. This example uses an MP4 formatted video selected, so MP4 format should be selected in the dialog.



5. Select the Imported Graphic Scaling. For this example 72 dpi was used. After making the selection, the video will appear in the FrameMaker document.

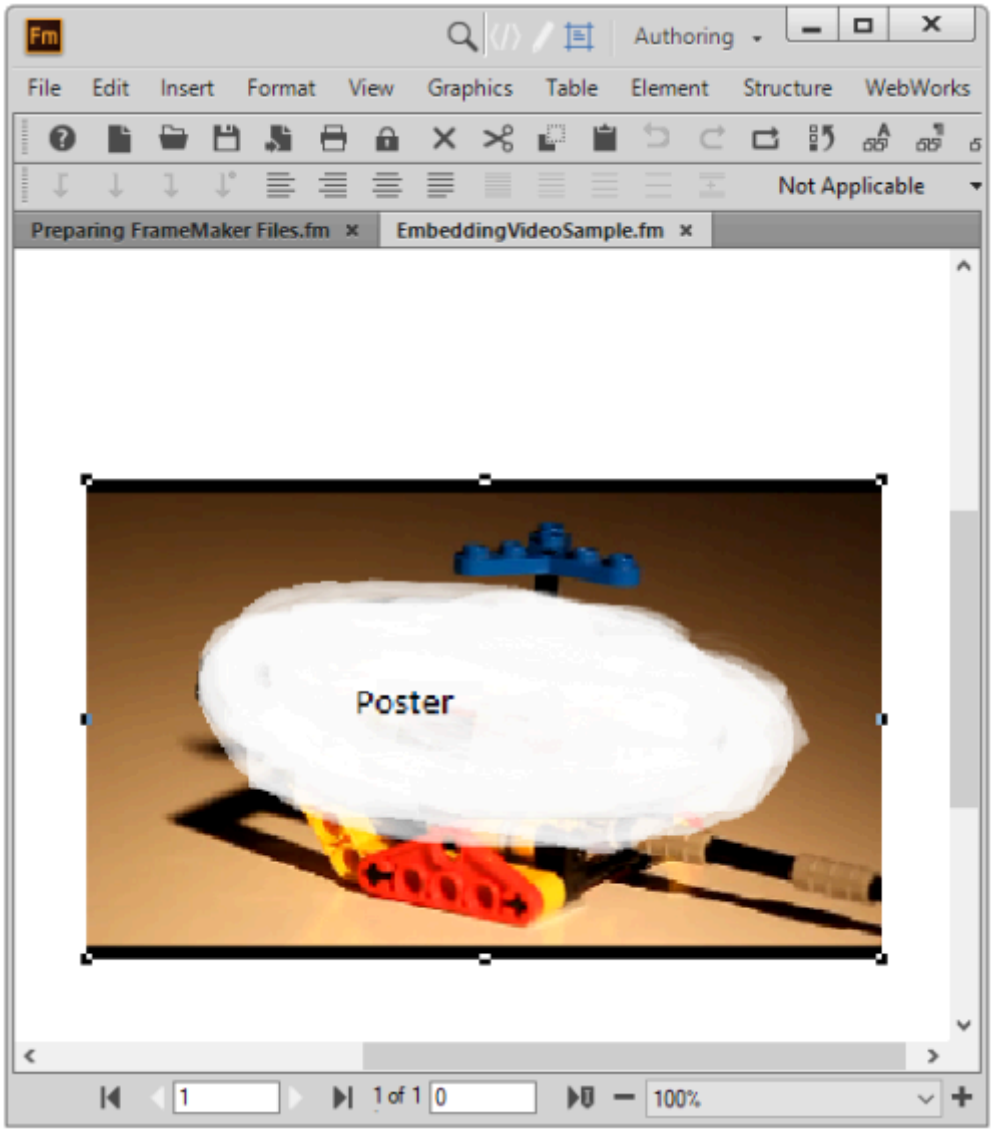


6. Add a poster image to the video file. This step is optional.

   a. Click on File, then select Import File. Select the image to use as the poster file image.

b. Click the Replace button.

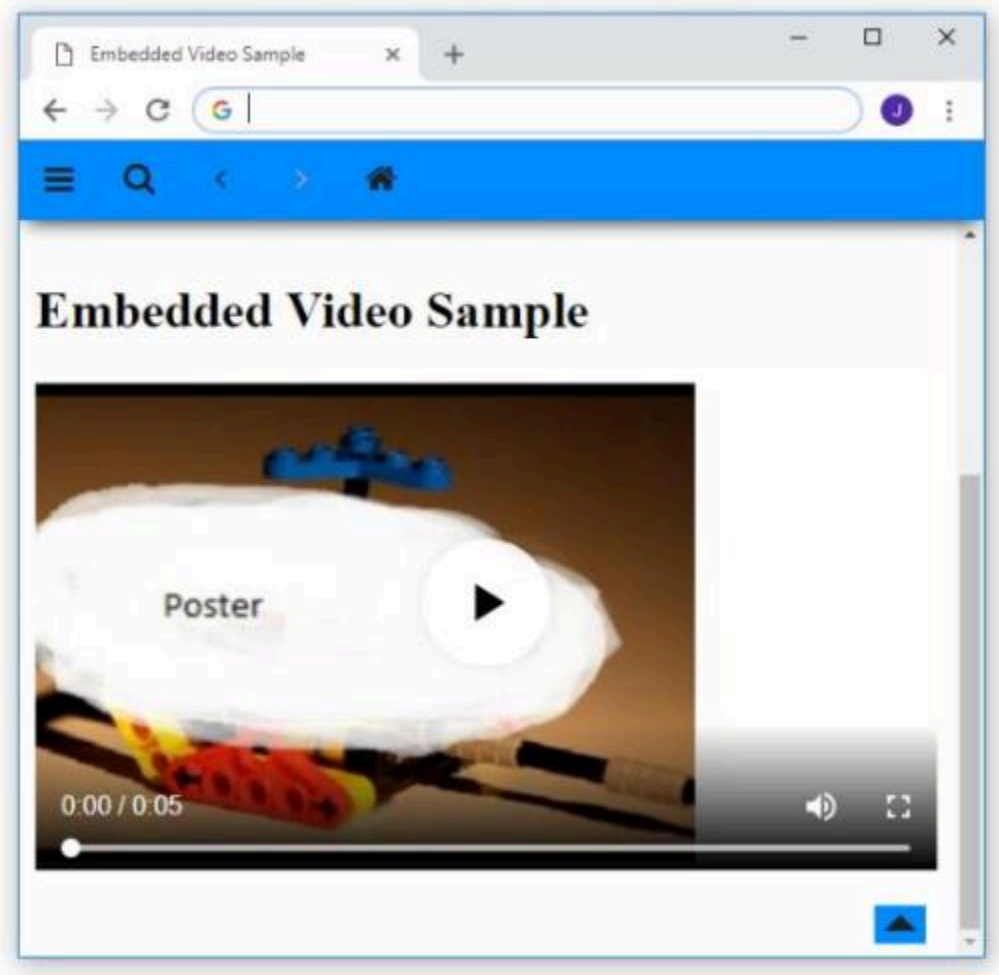
c. Select the Imported Graphic Scaling. 72 dpi was used in the example.



d. Position the video with your mouse.

e. Save the File

7. Add the document to your Reverb 2.0 project and generate.


Embedded Video Sample
Embedded Video Sample
Poster
0:00 / 0:05

Creating Index Entries in FrameMaker

An index lists the terms and topics discussed in a document and the page or pages on which they appear. An online index provides the user with a point-and-click resource for quickly navigating online content.

ePublisher uses the same native index entry features used in source documents to create a printed index to create an online index. If you include index entries in your source documents, ePublisher detects the index entries and uses the index entries to create an online index in your generated output.

Adobe FrameMaker inserts index entries as Index markers. To create index entries in an Adobe FrameMaker source document, insert Index markers into your Adobe FrameMaker source document. ePublisher then uses the Index markers to create an online index when you generate output.

Before you insert index entries, verify with the Stationery designer that your Stationery is configured to support online index generation. By default, ePublisher enables online index generate for output, but this functionality can be disabled in your Stationery by the Stationery designer. Also confirm that your output format supports online index creation.

Talk with the Stationery designer and other writers about the standard location and method you should use when you insert Index markers into your Adobe FrameMaker source documents. For example, some writers prefer to insert index entries into topic headings, while other writers prefer to insert index entries on the first line of the paragraph that contains the indexed term or terms. Some writers prefer to create one Index marker for each term, while other writers prefer to create one Index marker and then type all index terms associated with a paragraph into one Index marker.

The following procedure provides an example of how to inset index entries in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for inserting index entries in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To insert an index entry in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, insert your cursor in the location where you want to create an index entry.
2. On the **Special** menu, click **Marker**.
3. In the **Marker Type** field, select **Index** from the drop-down list.
4. In the **Marker Text** field, type the text you want to specify for the index entry.

Note: Following are some common ways writers can create index entries in Adobe FrameMaker. For more information about creating index entries, see the Adobe FrameMaker Help.

- ***If you want to specify multiple index entries in the marker***, separate each index entry with a semicolon (;) character.

  For example, type `index; table of contents; headings; footers`

- ***If you want to create a subentry,*** separate the primary and secondary entry with a colon (:).

For example, type `index:creating; index:generating;`

- ***If you want to create See references***, insert the entry but use the `<$nopage>` command to suppress the page number.

  For example, type `document, See source document <$nopage>`

- ***If you want to create see references with the word See italicized***, use a character tag inside the Index marker and the `<Default Para Font>` tag to turn off the character tagging.

  For example, type `document, <Emphasis>See <Default Para Font>source documents<$nopage>`

- ***If you want to create See also references***, use alternate text that specifies how Adobe FrameMaker sorts the see also reference.

  For example, type `document, <Emphasis>See also<Default Para Font>source documents<$nopage>[document:aa]`

  The text in brackets at the end of the entry controls where Adobe FrameMaker displays the text in the entry. In this example, the `aa` text ensures Adobe FrameMaker displays the entry as the first subentry under document.

5. Click **New Marker**.
6. After you insert you index entries, save your Adobe FrameMaker source document.
7. Generate output for your project. For more information, see “Generating Output”.
8. In Output Explorer, verify ePublisher created the index correctly by clicking on the page or tab that displays the index and then clicking on the index entries. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Using Variables in FrameMaker

A variable serves as a placeholder for information that may change frequently. Using variables in source documents allows you to quickly and easily control the content in your generated output. When you change the value of a variable in an ePublisher project, it changes the value in only your generated output. The variable value does not change in your source document.

Once you insert variables into your source documents, whenever the value of a item defined by a variable needs to change, you can make the change in a single location, rather than searching and replacing for all instances of the item. For example, you can use variables in the following ways:

- If you have publication dates or release dates in your source documents that you need to update periodically, you can set up the date as a variable.
- If you work with products that have names or versions that frequently change, you can set up variables for product names and versions.
- If you need to produce documentation sets for a product with multiple brands, you can use variables to help you produce documentation for each different brand using the same set of source files.

Importing or Creating Variables in FrameMaker

When you work with Adobe FrameMaker source documents, typically you import variables into your Adobe FrameMaker source documents from an Adobe FrameMaker template. The Adobe FrameMaker template contains variables defined by the Stationery designer.

Typically you should not need to create variables in your Adobe FrameMaker source files if you use an Adobe FrameMaker template created by a Stationery designer. However, in some cases you may need to create a variable in an Adobe FrameMaker source document if you do not have an Adobe FrameMaker template that includes a variable you need for your project.

The following procedure provides an example of how to import or create variables in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for importing or creating variables in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To import variables or create a variable in an Adobe FrameMaker source document

1. Open your Adobe FrameMaker source file.
2. ***If you want to import variables into your Adobe FrameMaker source file from an Adobe FrameMaker template***, complete the following steps:
    a. Open the Adobe FrameMaker template that contains the variables you want to import.
    b. On the **File** menu, click **Import** > **Formats**.
    c. In the **Import from Document** field, select the Adobe FrameMaker template that contains the variables you want to import from the list.
    d. In the **Import and Update** field, select only the **Variable Definitions** check box.
    e. Click **Import**.
    f. Click **OK** to confirm the operation.
3. ***If you want to create a variable in your Adobe FrameMaker source file***, complete the following steps:
    a. On the **Special** menu, click **Variables**.
    b. Click **Create Variable**.
    c. In the **Name** field, type a name for the variable. Variable names are case sensitive. For example, VariableName and variablename are different variables.
    d. Insert your cursor in the **Definition** field.
    e. In the **Character Formats** field, select a character format for the variable and then type a value for the variable. For more information about specifying character formats for variables, see the Adobe FrameMaker Help.
    f. Click **Add**. Adobe FrameMaker adds the variable to the list of variables. The variable is the value that Adobe FrameMaker displays in your Adobe FrameMaker source document.

g. Click **Done**.

4. Save your Adobe FrameMaker source file.

Inserting Variables into FrameMaker

You can insert a variable into a source document after you import the variables into your source document. If you want to use a variable that is not defined in an Adobe FrameMaker template, you must create the variable in your source document before you can insert it. For more information about importing or creating variables, see “Importing or Creating Variables in FrameMaker”.

The following procedure provides an example of how to insert variables in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for inserting variables in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To insert a variable into an Adobe FrameMaker source document

1. Open the Adobe FrameMaker source document into which you want to insert a variable.
2. Place your cursor in the location where you want to insert the variable.
3. On the **Special** menu, click **Variable**.
4. In the **Variable** field, select the variable you want to insert from the list, and then click **Insert**. Adobe FrameMaker inserts the variable.

Changing Variable Values in FrameMaker

You can change the value assigned to a variable in an Adobe FrameMaker source document.

The following procedure provides an example of how to change variable values in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for changing variable values in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To change a variable value in an Adobe FrameMaker source document

1. Open the Adobe FrameMaker source document that contains the variable with a value you want to change.
2. On the **Special** menu, click **Variable**.
3. In the **Variable** field, select the variable with the value you want to change.
4. Click **Edit Definition**.
5. In the **Definition** field, edit the variable value.
6. Click **Done**. Adobe FrameMaker updates the variable value in each place in your source document where you inserted the variable.
7. Click **Done** again to close the window.

Deleting Variables in FrameMaker

Delete a variable in an Adobe FrameMaker source document when you no longer want to use the variable. Before you delete a variable, ensure you search for the variable and delete or replace all references to the variable. If your source document still contains a reference to a variable after you delete it, Adobe FrameMaker prompts you to convert references to the variable in your source document to editable text.

The following procedure provides an example of how to delete variables in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for deleting variables in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To delete a variable in an Adobe FrameMaker source document

1. Open the Adobe FrameMaker source document that contains the variable you want to edit.
2. Search for and replace all references to the variable you want to delete in the source document by completing the following steps:
    a. On the **Edit** menu, click **Find/Change**.
    b. In the **Find** field, select **Variable of Name** from the list.
    c. In the field next to the **Find** field, type the name of the variable you want to delete.
    d. Click **Find**.
    e. Delete each variable you find or replace the variable with a different variable as appropriate.
3. On the **Special** menu, click **Variable**.
4. In the **Variable** field, select the variable you want to delete.
5. Click **Edit Definition**.
6. In the **User Variables** field, ensure the variable you want to delete is selected, and then click **Delete**.
7. Click **Done**.
8. Click **OK** to confirm the operation.

Using Conditions in FrameMaker

Conditions allow you to show or hide information in your source documents and in your online output. You apply conditions to the content in your source documents, and then you set the visibility for those conditions either in your source documents or in your ePublisher project.

For example, your source documents might contain some content that should be displayed in only the printed version and other content that should be displayed in only the online version. You can use the same set of source documents for both printed and online versions through the use of conditions. You can create one condition called PrintOnly specifically for printed content, and then you can create another condition called OnlineOnly specifically for online content. After you create the PrintOnly and OnlineOnly conditions, you can apply them to the appropriate content in your source documents.

After you apply conditions in your source documents, ePublisher can use the conditions defined in your source document to control the visibility of content when it generates output. You can also change the visibility specified for any condition in your ePublisher project. Changing the visibility specified for any condition in your ePublisher project does not change the visibility specified for the condition in your source documents.

Creating Conditions in FrameMaker

When you work with Adobe FrameMaker source documents, typically you import conditions into your Adobe FrameMaker source documents from an Adobe FrameMaker template. The Adobe FrameMaker template contains conditions defined by the Stationery designer.

Typically you should not need to create conditions in your Adobe FrameMaker source files if you use an Adobe FrameMaker template created by a Stationery designer. However, in some cases you may need to create a condition in your Adobe FrameMaker source documents if you do not have an Adobe FrameMaker template that includes a condition you need for a project. If you need to create a condition that is not available in your Adobe FrameMaker template, use native Adobe FrameMaker functionality to create the condition.

The following procedure provides an example of how to create conditions in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for creating conditions in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create a condition in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, on the **Special** menu, click **Conditional Text**.
2. Click **Edit Condition Tag**.
3. In the **Tag** field, type a name for the condition.

   For example, if you want to create a condition for content that you want to display in only online content, type `OnlineOnly`. If you want to create a condition for content that you want to display in only printed content, type `PrintOnly`.
4. ***If you want to specify a style for the condition***, in the **Style** field, select the style you want to use for the condition from the drop-down list. Specifying a style for the condition allows you to more easily see the content tagged with the condition in your Adobe FrameMaker source documents. If you do not want to use a style for the condition, select **As Is**.
5. ***If you want to specify a color for the condition***, in the **Color** field, select a color for the condition from the drop-down list. Specifying a color for the condition allows you to more easily see the content tagged with the condition in your Adobe FrameMaker source documents. If you do not want to use a color for the condition, select **As Is**.
6. Click the **Set** to create the condition.

Applying Conditions in FrameMaker

After you have imported conditions from your Adobe FrameMaker template or created conditions in your Adobe FrameMaker source document, you can apply conditions to content in your Adobe FrameMaker source documents. For more information about creating conditions in Adobe FrameMaker source documents, see “Creating Conditions in FrameMaker”.

The following procedure provides an example of how to apply conditions in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for applying conditions in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To apply a condition to content in an Adobe FrameMaker source document

1. In your Adobe Framemaker source document, select the content to which you want to apply the condition.
2. On the **Special** menu, click **Conditional Text.**
3. In the **Not In** list, select the condition you want to apply to the content.
4. Click the left arrow to move the condition from the **Not In** list to the **In** list.
5. Click **Apply** button to apply the condition.

Removing Conditions in FrameMaker

If you no longer want to apply a condition to content in an Adobe FrameMaker source document, you can remove the applied condition from the content.

The following procedure provides an example of how to remove conditions from content in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for removing conditions from content in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To remove a condition from content in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, select the content with the condition you want to remove.
2. On the **Special** menu, click **Conditional Text**.
3. Click **Unconditional**.
4. Click **Apply**.

Modifying Conditions in FrameMaker

You can edit the name of a condition and change the color or style assigned to a condition in an Adobe FrameMaker source document.

The following procedure provides an example of modify conditions in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for modifying conditions in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To modify a condition in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, on the **Special** menu, click **Conditional Text**.
2. Select the condition you want to modify.
3. Click **Edit Condition Tag**.
4. ***If you want to modify the name of the condition***, in the **Tag** field, type the new name for the condition.
5. ***If you want to modify the style specified for the condition***, in the **Style** field, select the style you want to apply to the condition from the drop-down list. If you do not want to use a style, select **As Is**.
6. ***If you want to modify the color specified for the condition***, in the **Color** field, select a color for the condition from the drop-down list. If you do not want to use a color, select **As Is**
7. Click **Set** to modify the condition.

Showing and Hiding Conditions in FrameMaker

You can show and hide conditions you applied in your Adobe FrameMaker source document. You can also show all of the conditions you applied in your Adobe FrameMaker source document. Showing all of the conditions applied allows you to see where all of the conditional content is in your Adobe FrameMaker source document.

The following procedure provides an example of how to show and hide conditions in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for showing and hiding conditions in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To show and hide conditions in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, on the **Special** menu, click **Conditional Text**.
2. Click **Show/Hide**.
3. ***If you want to show all conditions***, click **Show All**. Showing all conditions is helpful when you are working with a document and you want to be sure you can see all of the content in the document.
4. ***If you want to show a specific condition***, select it, and then click the left arrow to move it to the **Show** list on the left.
5. ***If you want to hide a specific condition***, select it, and then click the right arrow to move it to the **Hide** list on the right.
6. Click **Set**.

Using Passthrough Conditions in FrameMaker

A passthrough condition is a condition you apply to content that you do not want ePublisher to process when you generate output. For example, if you have embedded multimedia files in your source documents, such as Audio Video Interleave files (`.avi`) or Adobe Software Flash files (`.swf`), you can apply a passthrough condition to the code so that ePublisher does not process the code.

The following example shows `.avi` code to which you can apply a passthrough condition.

```
<embed src="sample.avi" width="400"
 height="300" pluginspage=";>
</embed>
```

The following example shows `.swf` code to which you can apply a passthrough condition.

```
<embed src="sample.swf" width="400"
 height="300" pluginspage="
http://www.macromedia.com/shockwave/download/index.cgi?
P1_Prod_Version=ShockwaveFlash";>
</embed>
```

If you have code in your Adobe FrameMaker source documents that you do not want ePublisher to process, create a passthrough condition and then apply the passthrough condition to the code.

Typically you use a passthrough condition defined in an Adobe FrameMaker template by a Stationery designer. You import this condition into your Adobe FrameMaker source document from an Adobe FrameMaker template. After you import the passthrough condition into your source document from the template, you apply the passthrough condition to the content as appropriate.

Typically you should not need to create a passthrough condition in your Adobe FrameMaker source file if you use an Adobe FrameMaker template created by a Stationery designer. However, in some cases you may need to create a passthrough condition in your Adobe FrameMaker source document if you do not have an Adobe FrameMaker template that includes a passthrough condition you need for a project. If you need to create a passthrough condition that is not available in your Adobe FrameMaker template, use native Adobe FrameMaker functionality to create the condition. For more information about creating a condition and applying a condition, see “Creating Conditions in FrameMaker” and “Removing Conditions in FrameMaker”.

You can also use Passthrough markers and the Passthrough paragraph styles and character styles options to insert content directly into your output without being transformed and coded for your output.

Deleting Conditions in FrameMaker

Delete a condition in an Adobe FrameMaker source document when you no longer want to apply the condition to content in the source document.

The following procedure provides an example of how to delete conditions in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for deleting conditions in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To delete a condition in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, on the **Special** menu, click **Conditional Text**.
2. Select the condition you want to delete.
3. Click **Edit Condition Tag**.
4. Click **Delete**.

Conditional Output Using Expressions in FrameMaker

Adobe FrameMaker 8.0 introduced ways to use Boolean Expressions (using the terms AND, OR, or Not) in condional text, for exmaple WebHelp AND PDF. To do this, you first create the conditions, and then use the Build Expression button to create this.

The following procedure provides an example of how to use Expressions in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 9. Steps for deleting conditions in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To build an Expression in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, on the **Special** menu, click **Conditional Text** -> **Show/Hide Conditional Text**
2. Click button **Build Expression**
3. Click desired condition and the arrow button to add it to the Expression
4. Separate the conditions by clicking the buttons for “AND” “OR” or “NOT”
5. When finished select Set so that this Expression will be able to be selected in the **Expression** dropdown located in the **Show/Hide Conditional Text** window
6. Go to **Special** -> **Conditional Text** -> **Show/Hide Conditional Text** and select the Show as per Expression radio button
7. Select the desired text you want to apply the expression and hit **Apply**
8. Save your source document.
9. Generate output for your project. For more information, see “Generating Output”.
10. In Output Explorer, verify ePublisher created an output file using the file name you specified. For more information, see “Viewing Output in Output Explorer”.

Specifying Output File Names in FrameMaker

By default, ePublisher automatically assigns file names to your generated output files for topics (pages) and for embedded image (graphic) output files.

Note: If you insert your images by reference in Adobe FrameMaker, ePublisher preserves the original file names. For more information, see "Working with Images in FrameMaker".

You can customize this naming convention using one of the following methods:

- Inserting Filename markers into your source documents
- Specifying the topic (page) and image (graphic) naming patterns for ePublisher to use in the target settings for your output

This section explains how you can specify output file names in your FrameMaker source documents using Filename markers. For more information about using target settings to specify output file names using page and image naming patterns, see "Specifying Page, Image, and Table File Naming Patterns".

To specify a file name for a page or image output file using Filename markers, your Stationery and FrameMaker template must have the Filename marker type configured.

The following procedure provides an example of how to specify page and embedded image output file names in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for specifying page and embedded image output file names in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To specify page and embedded image output file names in an Adobe FrameMaker source document

1. ***If you want to insert a Filename marker for a page output file***, complete the following steps:
    a. In your Adobe FrameMaker source document, locate the page for the topic to which you want to assign a specific filename. For more information about creating pages using page breaks, see "Specifying Page Breaks Settings".
    b. Insert your cursor at the beginning of the first paragraph on the page.
2. ***If you want to insert a Filename marker for an embedded image output file***, complete the following steps:
    a. In your Adobe FrameMaker source document, locate the embedded image for which you want to assign the output graphic file.
    b. On the **Graphics** menu, click **Tools** to display the graphic tools palette.
    c. Click the **Text Frame** icon.
    d. Drag the cursor across the image to draw a text frame over the image.

e. In the Create New Text Frame window, in the **Number** field, type `1`, and then click **Set**.

f. Click outside of the image, and then insert your cursor in the text frame.

3. In Adobe FrameMaker, on the **Special** menu, click **Marker**.
4. In the **Marker Type** field, select **Filename** from the drop-down list.
5. ***If the Filename marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to "Implementing Online Features in FrameMaker".
6. In the **Marker Text** field, type the file name you want to assign to the output file. Do not include the output file extension when you type the Filename marker text.
7. Click **New Marker**.
8. Save your source document.
9. Generate output for your project. For more information, see "Generating Output".
10. In Output Explorer, verify ePublisher created an output file using the file name you specified. For more information, see "Viewing Output in Output Explorer".

Creating Context-Sensitive Help in FrameMaker

This section explains how you can use ePublisher to create links to context-sensitive help content in Adobe FrameMaker source documents.

Context-Sensitive Help in FrameMaker

Context-sensitive help links provide content based on the context of what the user is doing. In many cases, this help content is based on the window that is open and active. For example, the Help button on a window in a software product can open a specific Help topic that provides important information about the window:

- What the window allows you to do
- Brief concepts needed to understand the window
- Guidance for how to use the window
- Descriptions about each field on the window, valid values, and related fields
- Links to related topics, such as concepts and tasks related to the window

The Help topic can also be embedded in the window itself, such as an HTML pane that displays the content of the Help topic. Providing this content when and where the user needs it, without requiring the user to search through the help, keeps the user productive and focused. This type of help also makes the product more intuitive by providing answers when and where needed.

There are several methods for creating context-sensitive Help. In addition, output formats use different mechanisms to support context-sensitive Help. You can reference a topic in the following ways:

File name

Use a Filename marker to assign a file name to a topic. Each topic can have no more than one Filename marker by default. However, you can create a custom mapping mechanism using file names. Then, you can open the specific topic with that file name. However, if your file naming changes, you need to change the link to the topic. This file naming approach delivers context-sensitive help capabilities in output formats that do not provide a mapping mechanism.

Internal identifier (topic alias)

Use a TopicAlias marker to define an internal identifier for each topic. The benefit of using an internal identifier is that it allows file names to change without impacting the links from the product. The writer inserts this marker in a topic and specifies a unique value for that topic. Then, the mapping mechanism of your output format determines how that internal identifier is supported. Some output formats, such as HTML Help, use a mapping file that defines these topic aliases.

To simplify the coding of your source documents, the Stationery designer can also configure your Stationery to define both the file name and the topic alias for each topic file.

Before you begin to insert Filename markers or TopicAlias markers into your source documents, consult with your Stationery designer. Confirm that your Stationery supports context-sensitive help links, and discuss with your Stationery designer the type of marker you should use to define context-sensitive help link in your source documents.

If you generate Eclipse Help output, you also can choose the topic description you want to display for each context-sensitive link. When you use a TopicAlias marker to create context-sensitive links, Eclipse creates a `contexts.xml` file that lists all of the context IDs for the Eclipse Help system you created using TopicAlias markers. In the `contexts.xml` file, Eclipse also provides a description of the context-

sensitive link. By default, the description Eclipse provides for the context-sensitive link is the text of the first paragraph of the topic. However, if you want to specify a different description for the context-sensitive link, you can do this by using the TopicDescription marker. For more information about using the TopicDescription marker, see “Specifying Context-Sensitive Help Links in FrameMaker”.

Planning for Context-Sensitive Help in FrameMaker

Creating context-sensitive help requires you to collaborate with application developers. Because topic IDs and map numbers must be embedded in both the software application and in your source documents, you and the application developers must agree in advance on the values to use.

Before you create context-sensitive help topics, first confirm with your application developers that the application supports context-sensitive help. Then work with your application developers to decide how to choose the topic ID for each context-sensitive help topic:

You choose the topic IDs

You can choose a set of topic IDs and embed them in your source documents using TopicAlias markers. When you generate output, ePublisher can generate a mapping file using those topic IDs and assign a unique number to each topic ID. You can provide the generated mapping file to your application developers, who can embed the topic IDs in the application code. You can then manually maintain this mapping file, or you can allow ePublisher to generate a new file each time you generate the help. Remember to give the updated help system and mapping file to your application developers each time.

Your developers choose the topic IDs

Your application developers can choose a set of topic IDs and embed them in the application code. Then, you can get a copy of the mapping file from your application developers, specify this mapping file in your project settings, and embed the topic IDs in your source documents using TopicAlias markers. In this case, ePublisher does not generate the mapping file.

Before you begin to implement context-sensitive help, meet with your application developers to select one of these methods for assigning the topic IDs to use for context-sensitive help links. Once you choose a set of topic IDs, embed them in your source documents using TopicAlias markers and do not change them.

Specifying Context-Sensitive Help Links in FrameMaker

You can use TopicAlias markers that contain topic IDs, or Filename markers that specify file names, to create context-sensitive help. If your output format supports the use of mapping files and topic IDs, typically you use TopicAlias markers to create context-sensitive help. If your output format does not support the use of mapping files and topic IDs, typically you use Filename markers to create context-sensitive help.

If you are generating Eclipse Help, you can also choose to specify a topic description for each context-sensitive help link your created using a TopicAlias marker by using a TopicDescription marker in conjunction with the TopicAlias marker. For more information about how TopicAlias markers and TopicDescription markers can work together when generating Eclipse Help, see "Context-Sensitive Help in FrameMaker".

To specify a context-sensitive help link, your Stationery and template must have a TopicAlias or Filename marker type configured. If you are generating Eclipse Help and you want to be able to specify topic descriptions for your context-sensitive help links, your Stationery and template must also have a TopicDescription marker type configured. Consult with the Stationery designer to determine which marker type you should use to create context-sensitive help links and topic descriptions in your source documents.

The following procedure provides an example of how to create context-sensitive help links and topic descriptions in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for creating context-sensitive help links in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create a context-sensitive help link in an Adobe FrameMaker source document

1. Open the Adobe FrameMaker source document that contains the context-sensitive topic you want to link to when users click a help button or help icon from within an application.
2. Insert your cursor at the beginning of the topic or paragraph in which you want to link.
3. On the **Special** menu, click **Marker**.
4. In the **Marker Type** field, select the marker type the Stationery designer configured your Stationery to support from the drop-down list. For example, select **TopicAlias** or **Filename**.
5. ***If the TopicAlias or Filename marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to "Implementing Online Features in FrameMaker".
6. In the **Marker Text** field, type the topic ID or file name you want to use for the context-sensitive help link. Specify topic IDs or file names that met the following guidelines:
    - Must be unique
    - Must begin with an alphabetical character
    - May contain alphanumeric characters

- May not contain special characters or spaces, with the exception of underscores (_)

7. Click **New Marker**.
8. ***If you are generating Eclipse Help and you want to specify topic descriptions for each context-sensitive help link you are creating***, complete the following steps:
    a. Insert your cursor in the topic after the TopicAlias marker you inserted for the Eclipse context-sensitive help topic.
    b. On the **Special** menu, click **Marker**.
    c. In the **Marker Type** field, select **TopicDescription** marker type from the drop-down list.
    a. ***If the TopicDescription marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.
    b. In the **Marker Text** field, type the topic description you want to use.
    c. Click **New Marker**.
9. Save your source Adobe FrameMaker source document.
10. Generate output for your project. For more information, see “Generating Output”.
11. In Output Explorer, complete the following steps:
    a. Verify that ePublisher inserted the topic ID into the map file when it generated output.
    b. ***If you generated Eclipse Help and specified topic descriptions for your context-sensitive help topics***, verify that the `contents.xml` file for your Eclipse Help system contains the topic descriptions you specified for context-sensitive help topics.
    c. Test the generated output using the application and verify that the application links to the appropriate context-sensitive help topic. This testing ensures the context-sensitive help link you created displays correctly within the application.

Creating Popup Windows in FrameMaker

A popup window is a window that is smaller than standard windows and typically does not contain some of the standard window features such as tool bars or status bars. Popup windows display when users hover over or click on a link. The popup window closes automatically as soon as the users click somewhere else.

A typical use of popup windows is to display glossary terms. For example, in printed documentation, terms and definitions are typically grouped in a separate glossary document. However, in online content, you can display glossary definitions in popup windows. With glossary popup windows, users can choose whether or not they want to view the definition of a term.

You create popup windows by creating a link between the word or phrase in a topic and the content you want to display in the popup window. After you create the link, you then insert Popup markers or apply Popup paragraph styles to define the content you want to display in the popup window.

If the Stationery designer configured the Stationery to support popup windows using markers, you use the following Popup markers to create popup windows:

Popup

Specifies the start of the content to include in a popup window. The content displays in a popup window when users hover over or click on the link. In some output formats users can also view the content in a standard help topic window in addition to viewing the content in a popup window. For example, if you insert a Popup marker in front of a glossary definition, the glossary definition displays in both a popup window and in a glossary topic that contains the definition.

PopupEnd

Specifies the end of the content to display in the popup window.

PopupOnly

Specifies that the popup content displays only through a popup window. For example, if you insert a PopupOnly marker in front of a glossary definition, the glossary definition displays only in a popup window.

If the Stationery designer configured the Stationery to support popup windows using paragraph formats, you use the following paragraph formats to create popup windows:

Popup and Popup Append paragraph behaviors

Specifies that content displays both in popup windows and in standard help topics. You apply the Popup paragraph format to the first paragraph of content you want displayed in the popup window. If you have more than one paragraph of content you want to display, you apply the Popup Append format to the additional paragraphs.

For example, if you apply a glossary term and glossary definitions format for a glossary using the Popup and Popup Append format, the terms and definitions in your output display in both a popup window and in a glossary topic that contains the definitions.

Popup Only and Popup Only Append paragraph behaviors

Specifies that content displays only in popup windows. You apply the Popup Only paragraph format to the first paragraph of content you want displayed in the popup window. If you have more than one paragraph of content you want to display, you apply the Popup Only Append format to the additional paragraphs.

For example, if you apply a glossary term and glossary definition format for a glossary using the Popup Only and Popup Only Append paragraph format, the terms and definitions in your output display in only popup windows. The content is not displayed in an additional glossary topic that contains the definitions.

Creating Popup Window Links in FrameMaker

Your first step in creating a popup window is to create a link between a word or phrase in a topic and the popup content you want to display when users hover over or click the link. Use native Adobe FrameMaker functionality to create a link between the word or phrase in a topic and the content you want to display in a popup window. You can create a link in Adobe FrameMaker by using a cross-reference or by using hypertext markers.

Before you create popup window links, verify that your output format supports this feature.

The following procedure provides an example of how to create a popup window link in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for creating links in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create a link between a word or phrase and popup content in an Adobe FrameMaker source document

1. In you Adobe FrameMaker source document, locate the text you want to create a link to and display in the popup window.
2. ***If you want to create a link that includes the link target text***, create the link using a cross-reference by completing the following steps:
    a. Select the text for which you want to create a link.
    b. On the **Special** menu, click **Cross-Reference**.
    c. In the **Document** field, select the document that contains the content to which you want to link.
    d. In the **Paragraph Tags** field, select the paragraph tag used for the content to which you want to link.
    e. In the **Paragraphs** field, select the paragraph to which you want to link.
    f. In the **Format** field, select the appropriate format for the link. For example, if you are creating a link to a glossary term, select a glossary term cross-reference format.
    g. Click **Replace**.
3. ***If you want to create a link that does not include the link target text***, create the link using a hypertext marker by completing the following steps:
    a. Insert your cursor in front of the link target text.
    b. On the **Special** menu, click **Marker**.
    c. In the **Marker Type** field, click **Hypertext**.
    d. In the **Marker Text** field, `newlink` *`linkname`* or `newlink` *`filename:linkname`*, where *`linkname`* is the name of the named destination for the link, and *`filename`* is the name of the file that contains the link, if the link is in a different Adobe FrameMaker source

document. To make maintenance easy, create short link names that use alphanumeric, lowercase characters.

e. Click **New Marker**.

f. Insert your cursor in front of the word or phrase for which you want to create a link.

a. On the **Special** menu, click **Marker**.

b. In the **Marker Type** field, select **Hypertext** from the list.

c. In the **Marker Text** field, type `gotolink` *`linkname`* or `gotolink` *`filename:linkname`*, where *`linkname`* is the name of the named destination you created for the link, and *`filename`* is the name of the file that contains the link, if the link is in a different Adobe FrameMaker source document.

d. Click **New Marker**.

e. Select the word or phrase for which you want to create a link. The selected area must contain the both text and the hypertext marker you created.

f. Apply a link character format to the word or phrase. If you do not know which character format to use for links, consult the Stationery designer.

4. Save your Adobe FrameMaker source document.

After you create a link between a word or phrase in a topic and the popup content you want to display in the popup window, define the content you want to display in the popup window using one of the following methods:

- Create popup windows using Popup markers. For more information, see “Using Markers to Create Popup Windows in FrameMaker”.
- Create popup windows using Popup paragraph formats. For more information, see “Using Paragraph Formats to Create Popup Windows in FrameMaker”.

Using Markers to Create Popup Windows in FrameMaker

You can insert Popup markers into your Adobe FrameMaker source documents to create popup windows. To use Popup markers to create popup windows, your Stationery and FrameMaker template must have the following items configured:

- Popup marker type
- PopupEnd marker type
- PopupOnly marker type

Your output format must also support this feature.

The following procedure provides an example of how to insert Popup markers in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for inserting Popup markers in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

Note: Popup content is created from whole paragraphs. You cannot include a subset of a paragraph in a popup.

To use popup markers to create popup windows in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, create a link between a word or phrase in the topic and the content you want to display in the popup window. For more information, see “Creating Popup Window Links in FrameMaker”.
2. Insert your cursor in front of the text you want to display in the popup window.
3. On the **Special** menu, click **Marker**.
4. ***If you want the popup content to display in both a popup window and in a standard help topic,*** in the **Marker Type** field select **Popup** from the drop-down list.
5. ***If you want the popup content to display only in a popup window***, in the **Marker Type** field select **PopupOnly** from the drop-down list.
6. ***If the Popup or PopupOnly marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.
7. In the **Marker Text** field, do not enter any text. You do not need to enter any text in this field when you insert a Popup or PopupOnly marker.
8. Click **New Marker**.
9. Specify where you want the popup content to end by completing the following steps:
    a. Insert your cursor at the end of the content you want to display in the popup window.

b. On the **Special** menu, click **Marker**.

c. In the **Marker Type** field, select **PopupEnd** from the drop-down list.

d. ***If the PopupEnd marker type is not on the list,*** check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to "Implementing Online Features in FrameMaker".

e. In the **Marker Text** field, do not enter any text. You do not need to enter any text in this field when you insert a PopupEnd marker.

f. Click **New Marker**.

10. Save your Adobe FrameMaker source document.

11. Generate output for your project. For more information, see "Generating Output".

12. In Output Explorer, go to the page where you created the popup window and verify that ePublisher created the popup window and that the popup window displays the content you specified. For more information, see "Viewing Output in Output Explorer".

Using Paragraph Formats to Create Popup Windows in FrameMaker

You can use Popup paragraph formats in your Adobe FrameMaker source documents to create popup windows. To use Popup paragraph formats to create popup windows, your Stationery and FrameMaker template must have the following items configured:

- Popup and Popup Append paragraph formats if you want your content to display both in popup windows and in standard help topics.
- Popup Only and Popup Only Append paragraph formats if you want your content to display only in popup windows.

Your output format must also support this feature.

The following procedure provides an example of how to use Popup paragraph formats to create popup windows in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for using Popup paragraph formats to create popup windows in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create popup windows using Popup paragraph formats in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, create a link between a word or phrase in the topic and the content you want to display in the popup window. For more information, see “Creating Popup Window Links in FrameMaker”.
2. Apply the appropriate Popup paragraph format to the popup content you want to display in the popup window.
3. Save your Adobe FrameMaker source document.
4. Generate output for your project. For more information, see “Generating Output”.
5. In Output Explorer, go to the page where you created the popup window and verify that ePublisher created the popup window and that the popup window displays the content you specified. For more information, see “Viewing Output in Output Explorer”.

Creating Expand/Collapse Sections (Drop-Down Hotspots) in FrameMaker

You can create sections of content that expand and collapse when you click a link or hot spot. This structure allows you to create items, such as tasks with numbered procedures, bulleted lists, or definitions, that are easy to scan. Users can then expand individual items to display additional information.

Hot spots for expand/collapse sections initially display in one of the following states:

- The content is initially collapsed and will expand beneath the hotspot when the user clicks the hotspot. Clicking the hotspot a second time causes the expanded content to return to its original collapsed state.
- The content is initially expanded and will collapse or disappear from beneath the hotspot when the user clicks the hotspot.

You use an Expand/Collapse paragraph or table format to start expand/collapse sections and a DropDownEnd marker to specify where the content in the expand/collapse section ends. The Stationery defines whether the sections should initially be expanded (shown) or collapsed (hidden) and the image used to show the state of the section.

To create expand/collapse sections, your Stationery and FrameMaker template must have the following items configured:

- An Expand/Collapse paragraph or table format
- A DropDownEnd marker

Your output format must also support this feature.

The following procedure provides an example of how to create expand/collapse sections in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for creating expand/collapse sections in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create an expand/collapse section in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, identify a topic that contains text for which you want to create an expand/collapse section.
2. Apply an Expand/Collapse paragraph format to the text you want users to click to expand or collapse content.

   For example, in the following sample procedure, you could apply the Expand/Collapse paragraph format to the *To open a project* text.

   To open a project

   a. On the **File** menu, click **Open**.

b. Browse to the location of the project on your local computer.

c. Select the project, and then click **Open**.

3. Insert your cursor at the end of the content you want to display in the expand/collapse section.

For example, in the following sample procedure, you would insert your cursor after the period in the last sentence of the procedure, *Select the project, and then click Open*.

To open a project

a. On the **File** menu, click **Open**.

b. Browse to the location of the project on your local computer.

c. Select the project, and then click **Open**.

4. On the **Special** menu, click **Marker**.

5. In the **Marker Type** field, select **DropDownEnd** from the drop-down list.

6. ***If the DropDownEnd marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to "Implementing Online Features in FrameMaker".

7. In the **Marker Text** field, do not enter any text. You do not need to enter any text in this field when you insert a DropDownEnd marker.

8. Click **New Marker**.

9. Save your Adobe FrameMaker source document.

10. Generate output for your project. For more information, see "Generating Output".

11. In Output Explorer, go to the page where you created the expand/collapse section and verify that ePublisher created the expand/collapse section and that the expand/collapse section displays the content you specified. For more information, see "Viewing Output in Output Explorer".

Creating Related Topics in FrameMaker

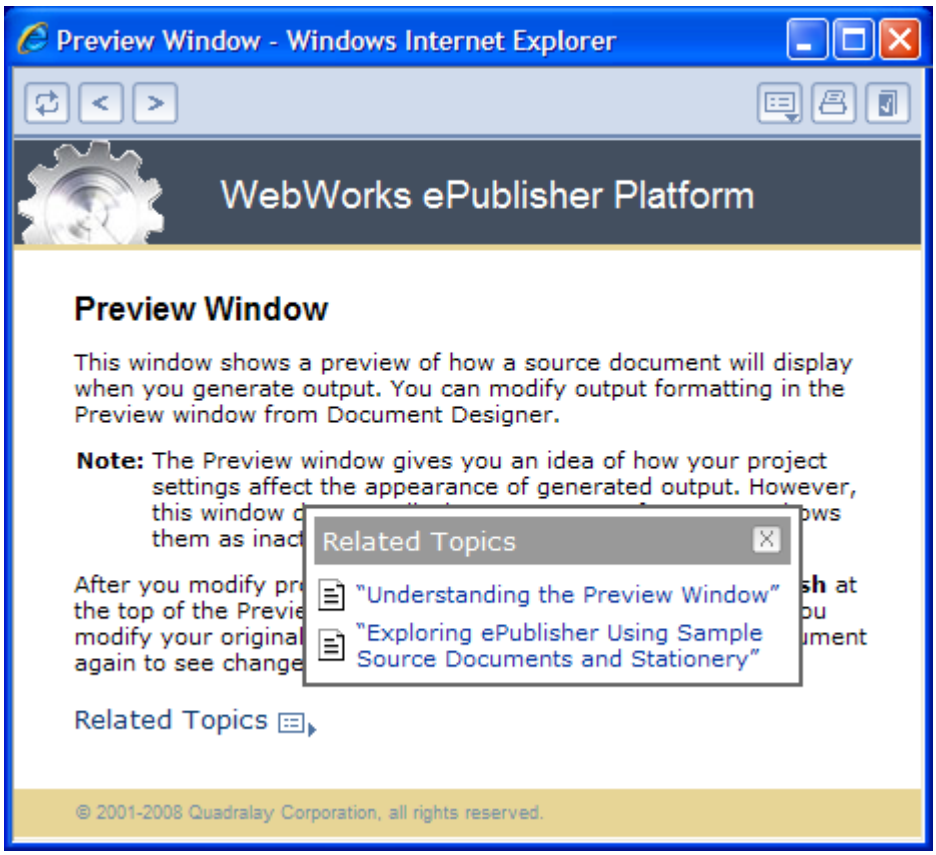
Related topics provide a list of other topics that may be of interest to the user viewing the current topic. For example, you could have a section called Creating Web Pages in your help. You may also have many other topics, such as HTML Tags and Cascading Style Sheets, that related to creating Web pages. Identifying these related topics for users can help them find the information they need and identify additional topics to consider. However, providing these types of links as cross-references within the content itself may not be the most efficient way to present the information. By utilizing related topics links, you combine the capabilities of cross-references with the efficiency of a related topics button.

Related topics and See Also links provide similar capabilities, but there are several important differences:

- Related topics can link to headings in a Help system that do not start a new page.
- Relate topics links are static and defined in the source documents as links. You must have all the source documents to create the link and generate the output.
- If a related topics list contains a broken link in the source document, that link is broken in the generated output. In a See Also link list, the broken link is not included in the output.

The Stationery designer can configure related topics to display in the following ways:

- Included as a list in the topic itself.
- Displayed in a popup window when the user clicks a button, as show in the following figure.



Note: If a related topic link is broken in the source document, in most cases that link is broken in the generated output. WebWorks Help and WebWorks Reverb provide an additional feature by removing broken links from related topics lists that are displayed in a popup window when a user clicks the Related Topics button.

To create related topics links, your Stationery and FrameMaker template must have a Related Topics paragraph style configured. Your output format must also support this feature.

The following procedure provides an example of how to create related topics links in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for creating related topics links in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create a related topics list in an Adobe FrameMaker source document

1. Identify the topic in which you would like to insert a related topics list.
2. Identify the different topics you want to link to from this topic.

   Note: Generally, you should only create one related topics list for each section of your source document that corresponds to a help topic. For example, if the Stationery designer specified in your Stationery that there will be a page break at each Heading 1 section, then you should only create one related topics list for each Heading 1 section within your source document.

3. Create a cross-reference to each topic you want to include in the related topics list by completing the following steps:

    a. Insert your cursor in the location in your Adobe FrameMaker source document where you want to insert the link to the related topic.

    b. On the **Special** menu, click **Cross-Reference**.

    c. In the **Document** field, select the source document that contains the topic to which you want to link.

    d. In the **Source Type** field, click **Paragraphs**.

    e. In the **Paragraph Tags** field click the paragraph tag used by the topic to which you want to link.

    f. In the **Paragraphs** field, select the topic to which you want to link.

    g. In the **Format** field, select the format you want to use for the cross-reference.

    h. Click **Insert**.

4. Apply the Related Topic paragraph format to the cross-references in your related topics list.

5. ***If you want to display the list of related topics in only your generated output***, apply an OnlineOnly condition to the list of related topics. For more information about applying conditions, refer to “Applying Conditions in FrameMaker”.

6. Save your Adobe FrameMaker source document.

7. Generate output for your project. For more information, see “Generating Output”.

8. In Output Explorer, go to the page where you created the related topics list and verify that ePublisher created the related topics and that the related topics list displays the topics you specified. For more information, see “Viewing Output in Output Explorer”.

Creating See Also Links in FrameMaker

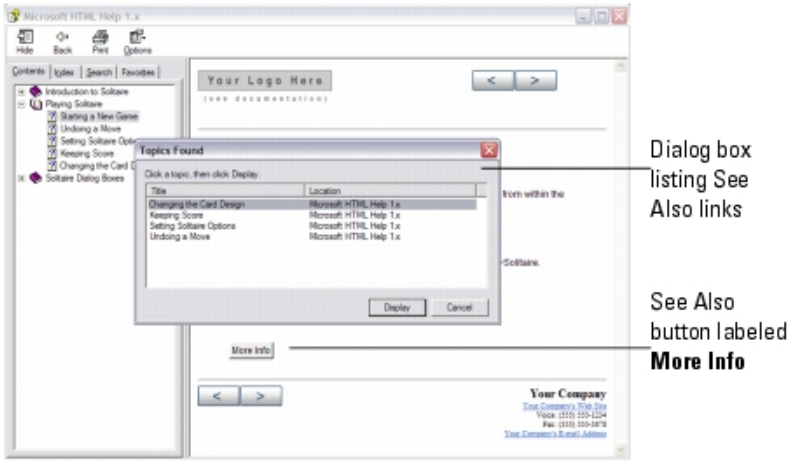
See Also links, also known as ALinks, or associative links, are links that may be of interest to the user viewing the current topic. These links use internal identifiers to specify the links and the link list is built dynamically based on the topics available when the user clicks to display the links. See Also links are important to use with larger help sets and merged help sets.

Related topics and See Also links provide similar capabilities, but there are several important differences:

- See Also links must link to formats that start a new topic, such as a heading.
- See Also links are dynamic and the lists of links are built at display time instead of during help generation.
- Since see Also link lists are dynamically built, they do not include links to topics that are not available when the user displays the links. If a related topics list contains a broken link in the source document, that link is broken in the generated output for most output formats.

See Also links are useful if you plan to merge help systems. For example, if you have a multiple help systems that you merge into one main help system at run time and if your topics in the merged help systems contain See Also keywords that are also used in the main help system, links to those topics are included in the See Also lists in the main project.

You can create See Also links as buttons or as inline text links in Microsoft HTML Help and WebWorks Help. The following example shows how the two different types of See Also links display in a Microsoft HTML Help system.



Create See Also links by applying the See Also paragraph format or character format to text in your Adobe FrameMaker source documents and inserting markers into your Adobe FrameMaker source documents. To create See Also links, your Stationery and template must have the following items configured:

- See Also paragraph format if you want to create See Also links with buttons

- See Also paragraph format if you want to create see Also links as inline text links
- SeeAlsoKeyword marker type
- SeeAlsoLink marker type
- SeeAlsoLinkDisplay marker type if you generate Microsoft HTML Help and you want to display the target topics in a popup menu
- SeeAlsoLinkWindowType marker type if you generate Microsoft HTML Help and you want to display the target topics in a custom window

The following procedure provides an example of how to create See Also links in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for creating See Also links in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create a See Also link in an Adobe FrameMaker source document

1. Identify each topic to which you want to link from a See Also link, and then complete the following steps for each topic:
   a. Insert your cursor into the topic to which you want to link.
   b. On the **Special** menu, click **Marker**.
   c. In the **Marker Type** field, select **SeeAlsoKeyword** from the drop-down list.
2. ***If the SeeAlsoKeyword marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to "Implementing Online Features in FrameMaker".
3. In the **Marker Text** field, type a text string that is a unique identifier for the topic.

   For example, if you have a unique topic called About WebWorks Help, type `AboutWebWorks` help in the **Marker Text** field.
4. Click **New Marker**.
5. Identify the topic where you want to insert a list of See Also links.
6. Enter the text you want to display for the See Also button or for the See Also inline text link on a separate line in the source document where you want the See Also button or inline text link to display.

   For example, if you want to create a button with the text See Also on the button, type `See Also`. If you want to create inline text with the text Additional Information for the link, type `Additional Information`.
7. ***If you want to create a See Also button for your See Also links***, apply the See Also paragraph format to the text you want to display in the See Also button.
8. ***If you want to create a See Also inline text link for your See Also links***, apply the See Also character format to the text you want to display for the See Also inline text link.

9. Apply an OnlineOnly condition to the See Also text. Applying an OnlineOnly condition to the See Also button or See Also inline text displays the See Also link in your generated output, but does not display the See Also button or link in your printed content.

10. Insert your cursor inside the text you specified for the See Also button or See Also inline text link.

11. On the **Special** menu, click **Marker**.

12. In the **Marker Type** field, select **SeeAlsoLink** from the drop-down list.

13. ***If the SeeAlsoLink marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to "Implementing Online Features in FrameMaker".

14. In the **Marker Text** field, type the text string that is a unique identifier for the topic to which you want to link. This text string is the text string you typed when you created the SeeAlsoKeyword marker for the topic.

    For example, if you have a unique topic called About WebWorks Help, type `AboutWebWorks` help in the **Marker Text** field.

15. Click **New Marker**.

16. Continue to insert SeeAlsoLink markers for each topic you want display when users click the See Also button or inline text link.

17. ***If you generate Microsoft HTML Help output and you want to display the target topics in a popup menu***, complete the following steps:

    a. Insert your cursor inside the text you specified for the See Also button or inline text link.

    b. In the **Marker Type** field, select **SeeAlsoLinkDisplayType** from the drop-down list.

       Note: This marker type is supported only in Microsoft HTML Help.

    c. ***If the SeeAlsoLinkDisplayType marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to "Implementing Online Features in FrameMaker".

    d. In the **Marker Text** field, type `menu`. By default, Microsoft HTML Help displays See Also links in the Topics Found window. To display See Also links in a popup menu, specify menu for the marker value.

    e. Click **New Marker**.

18. ***If you generate Microsoft HTML Help output and you want to display the target topics in a custom window***, complete the following steps:

    a. Insert your cursor inside the text you specified for the See Also button or inline text link.

    b. In the **Marker Type** field select **SeeAlsoLinkWindowType** from the drop-down list.

       Note: This marker type is supported only in Microsoft HTML Help.

c. ***If the SeeAlsoWindowType marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”

d. In the **Marker Text** field, type `menu`. By default, Microsoft HTML Help displays See Also links in the Topics Found window. To display See Also links in a popup menu, specify menu for the marker value.

e. Click **New Marker**.

19. Save your Adobe FrameMaker source document.

20. Generate output for your project. For more information, see “Generating Output”.

21. In Output Explorer, go to the page where you created the See Also links and verify that ePublisher created the See Also button or See Also inline text and that the See Also button or inline text displays the links you specified. For more information, see “Viewing Output in Output Explorer”.

Creating Meta Tag Keywords in FrameMaker

Meta tags are lines of code placed between the `<head>` and `</head>` tags in HTML pages. Meta tags give web search engines information about the content of the web page and how search engines should treat the web page. Users viewing web pages do not see the meta tags, but meta tags can be used to influence the way web pages on a web site appear in web search engine results. Users also see the text you specify for meta tags right following the title of your page when your page comes up in search results.

In help systems, search ranking works like ranking in an Internet search engine. If you generate help system output, you can use meta tag keywords to specify terms for pages for help topics where you want to improve searchability. For example, assume that in your help system you have a topic called See Also links. However, you know that See Also links are also sometimes referred to as ALinks, and you think that some users of your help system may search for information about See Also links by typing `ALinks` into the **Search** field for your help system. In this example, you can insert ALinks as a meta tag keyword for each page that discusses See Also links, so users who search your system for information about ALinks can find the information they are looking for in your See Also link topics.

To assign meta tag keywords, your Stationery and template must have the Keywords marker type configured. Your output format must also support this feature.

The following procedure provides an example of how to create meta tag keywords in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for creating meta tag keywords in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To create meta tag keywords for a page in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, find the first paragraph in the page for the page for which you want to create a meta tag keyword.
2. Insert your cursor into the paragraph.
3. On the **Special** menu, click **Marker**.
4. In the **Marker Type** field, select **Keywords** from the drop-down list.
5. ***If the Keywords marker type is not on the drop-down list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.
6. In the **Marker Text** field, type the comma-delimited list of keywords that you want search engines to use.

   For example, type `keyword1`, `keyword2`, `keyword3`, where *`keyword`* is the keyword you want search engines to use.
7. Click **New Marker**.

8. Save your Adobe FrameMaker source document.

9. Generate output for your project. For more information, see “Generating Output”.

10. In Output Explorer, verify that ePublisher inserted your meta tag keywords correctly by completing the following steps:

    a. On the **View** menu, click **Output Explorer**.

    b. In the *`TargetName`**`ProjectName`* folder, open the page to which you assigned meta tag keywords in Notepad, where *`TargetName`* is the name of your target and *`ProjectName`* is the name of your project.

    c. Verify that the text you specified for your meta tag displays in the `meta name` attribute between in the `<head>` and `</head>` tags section of your web page. For example, if you typed `keyword1`, `keyword2`, `keyword3`, for your meta tag keywords, your meta tags in for the page should be similar to the following entry:

    ```
    <meta name="keywords" content="keyword1, keyword2, keyword3" />
    ```

Assigning Custom Page Styles in FrameMaker

By default, each page generated by ePublisher is associated with the default page style defined in the Stationery used by your ePublisher project. This means that typically you do not need to specify a page style for pages when you generate output. However, if you want to change the page style of one page or a smaller set of pages, you can specify the page style you want to use for a page in your Adobe FrameMaker source document using the PageStyle marker.

For example, you may want to use one page style in your help system for all concept and procedure topic pages, and another page style for all context-sensitive window description topic pages in your help system. In this example, you can use the default page style for all of your concept and procedure topic pages, and then you can use a second custom page style defined in your Stationery for all context-sensitive window description topic pages in your help system.

To assign custom page styles, your Stationery and template must have the following items configured:

- Custom page styles defined for your Stationery by the Stationery designer
- PageStyle marker type

Your output format must also support this feature.

The following procedure provides an example of specifying custom page styles for pages in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for specifying custom page styles for pages in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To specify a custom page style for a page in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the page for the topic to which you want to assign a page style.
2. Insert your cursor in the location on the page where you want to insert the PageStyle marker.
3. On the **Special** menu, click **Marker**.
4. In the **Marker Type** field, select **PageStyle** from the drop-down list.
5. ***If the PageStyle marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.
6. In the **Marker Text** field, type the name of the custom page style the Stationery designer configured for your Stationery.

   For example, if the Stationery designer configured an page style called BluePage in your Stationery, type `BluePage`.

7. Click **New Marker**.
8. Save your Adobe FrameMaker source document.
9. Generate output for your project. For more information, see “Generating Output”.
10. In Output Explorer, verify ePublisher created the page using the page style you specified by clicking on the page and verifying ePublisher applied the page style you specified in the generated output. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Opening Topics in Custom Windows in FrameMaker

You can open topics in custom windows in Microsoft HTML Help and Oracle Help. By default, Microsoft HTML Help displays content in the standard Microsoft HTML Help tri-pane window. The Stationery designer can modify the size, position, and other characteristics of the tri-pane window in your Microsoft HTML Help project. The Stationery designer can also define custom windows for you to use in a Microsoft HTML Help project. If the Stationery designer defines custom windows in a Microsoft HTML Help project, you can specify which topics you want to display in the custom window using the WindowType marker.

By default, Oracle Help displays content in the standard Oracle Help viewer. The Stationery designer can modify the size, position, and other characteristics of Oracle Help windows. The Stationery designer can also define custom windows for you to use in an Oracle Help project. If the Stationery designer defines custom windows in an Oracle Help project, you can specify which topics you want to display in the custom window using the WindowType marker.

For example, if you want your context-sensitive help topics to display in a different type of window than other content, after you create a context-sensitive help topic you can use the WindowType marker to specify that you want the context-sensitive help topics to display in a custom window. After you assign a custom window to a topic using the WindowType marker, the help system displays the topic in your generated output in the custom window whenever users access the topic from the table of contents, index, a standard hyperlink, a related topics list, or a See Also link.

To open topics in custom windows, your Stationery and template must have the following items configured:

- Custom window styles defined for your Stationery by the Stationery designer
- PageStyle marker type

Your output format must also support this feature.

The following procedure provides an example of how to specify topics open in custom Microsoft HTML Help or Oracle Help windows in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for specifying topics open in custom Microsoft HTML Help or Oracle Help windows in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To specify topics open in a custom window in an Adobe FrameMaker source document

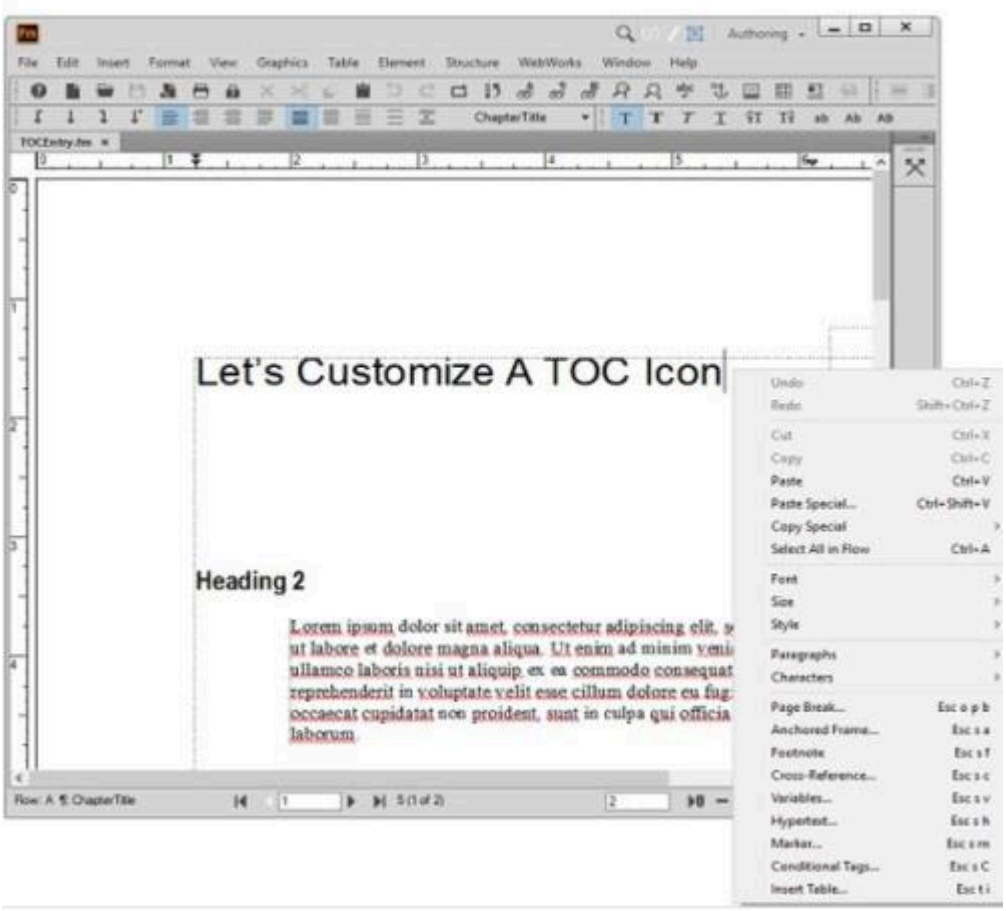
1. Obtain the names of custom windows configured in the Stationery you use for your ePublisher project from the Stationery designer.
2. In your Adobe FrameMaker source document, locate the topic that you want to open in a custom window.
3. Insert your cursor into the topic.
4. On the **Special** menu, click **Marker**.

5. In the **Marker Type** field, select **WindowType** from the drop-down list.
6. ***If the WindowType marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.
7. In the **Marker Text** field, type the name of the custom window configured by the Stationery designer that you want to specify for the topic.
8. Click **New Marker**.
9. Save your Adobe FrameMaker source document.
10. Generate output for your project. For more information, see “Generating Output”.
11. In Output Explorer, verify the topic displays in the custom window you specified for the topic. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

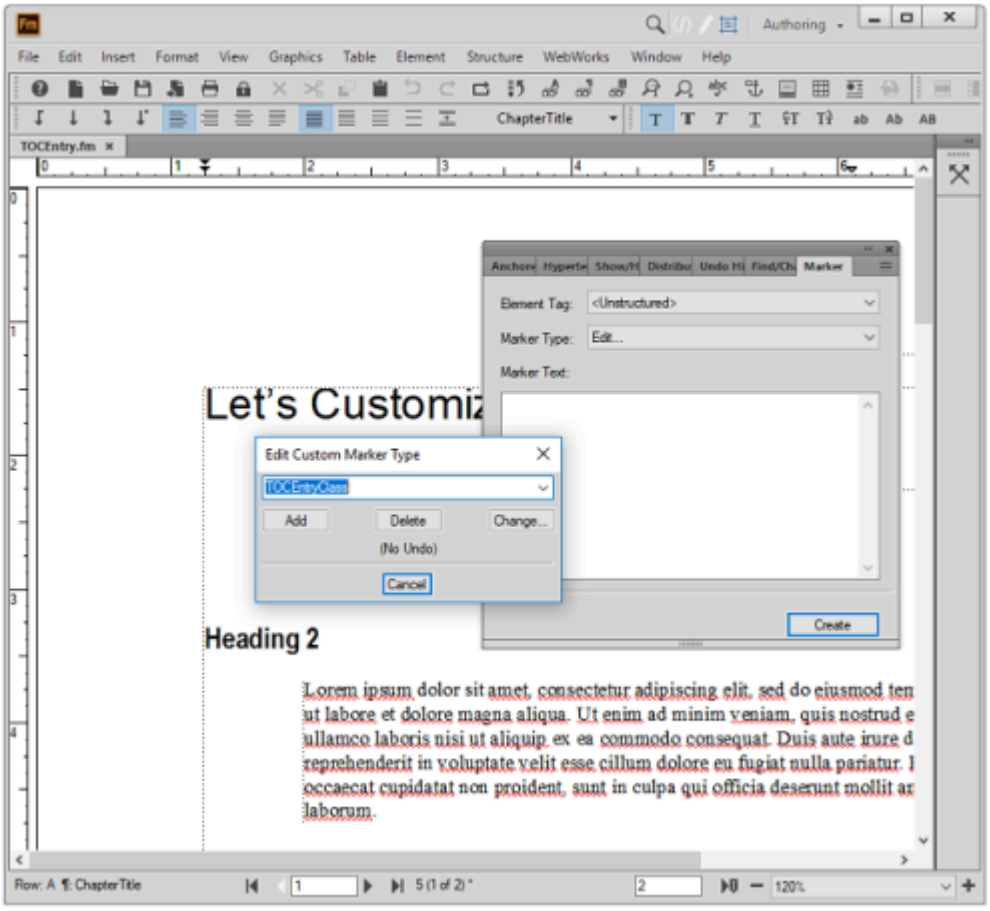
Customizing TOC Entry in FrameMaker

Use these steps to customize a TOC entry in your **Reverb 2.0** output. Your FrameMaker file must have a nested heading structure for TOC Icons to appear.

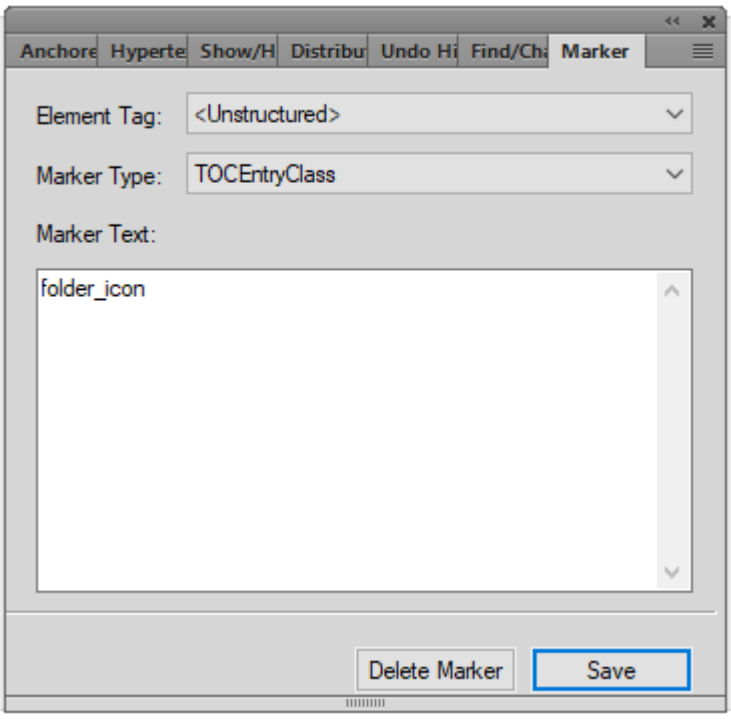
1. Right-click at the end the heading. Select Marker.



2. Select Edit from the Marker Type dropdown.
3. Type in TOCEntryClass. Click Add, then click Done.



4. In the Marker Text window, type in the name of your custom class. In the example, folder_icon is the class name.

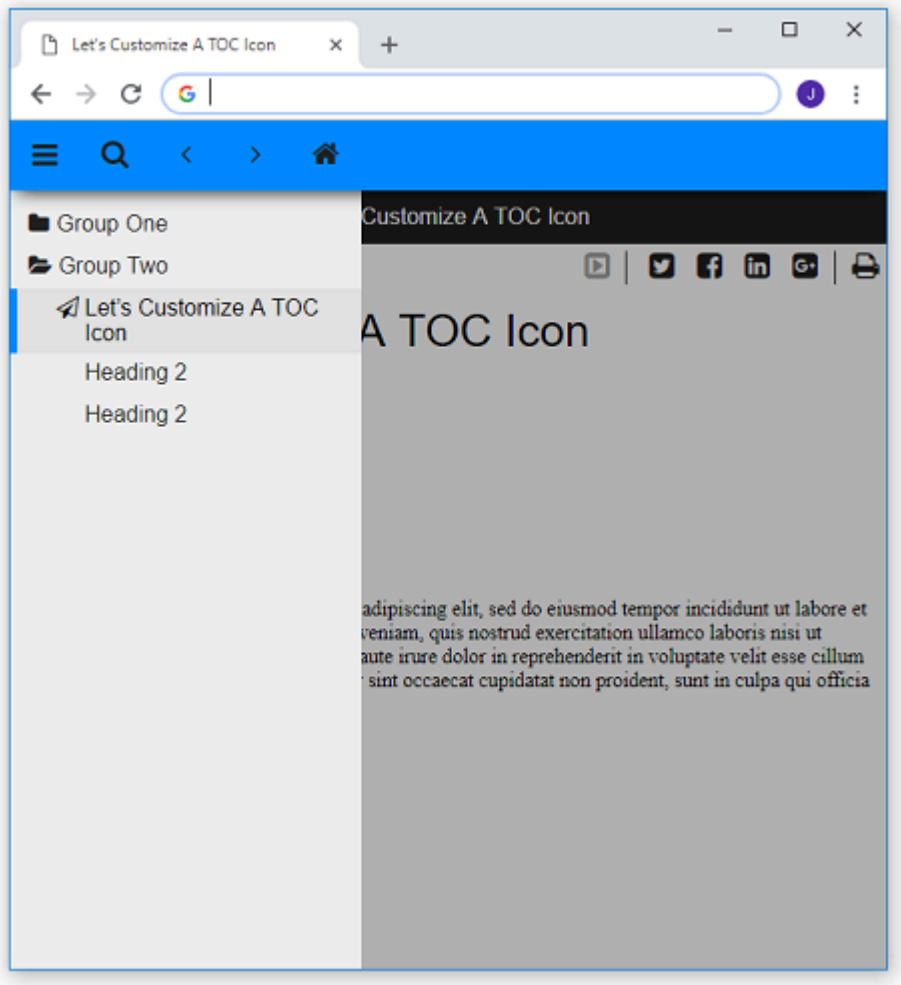


5. Save the FrameMaker document.
6. Scan the document in **ePublisher Designer**.
7. Open the **Style Designer**.
8. Open **Marker Styles**.
9. Locate the **Marker Type Option** from the **Options** tab and set its value to `TOC Entry Class`.
10. In this example, the assigned class for the Menu TOC entry will be the value of the marker: `folder_icon`.
11. Add the following to a target override of `_icons.scss`. Notice how the CSS class is the name of the value given in the Marker Text Window. In this example we change the icon color and the icon of the TOC entry. You are able to make other customizations such as adding a border, or changing the background color.

```
.folder_icon {

 > div > span > i {

color: black;

&:before {

      content: $folder_icon;

   }

}
```

```
}
```

12. Save your project and generate the output.



Customizing Table of Contents Icons in FrameMaker

By default, the **Contents** tab in a Microsoft HTML Help, Oracle Help, and WebWorks Help uses book and page icons to identify entries. By default, the **Contents** tab in Sun JavaHelp uses folder and page icons to identify entries. You can also customize the table of contents icons.

For example, if you want to make new topics stand out by using a unique icon specific to new books, pages, or folders, you can insert a marker into a topic and specify the icon you want to display for the book, page, or folder in your help system table of contents.

To customize a table of contents icon, your Stationery and template must have the following items configured:

- TOCIconHTMLHelp for Microsoft HTML Help
- TOCIconOracleHelp for Oracle Help
- TOCIconJavaHelp for Sun JavaHelp
- TOCIconWWHelp for WebWorks Help

You can customize the appearance of table of contents icons for topics in Microsoft HTML Help, Sun JavaHelp, Oracle Help, and WebWorks help.

The following procedure provides an example of how to customize table of contents icons for topics in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for customizing table of contents icons for topics in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To specify a custom table of contents icon in an Adobe FrameMaker source document

1. ***If you want to specify a custom table of contents icon for Microsoft HTML Help***, identify the number of the image you want to use for the table of contents image for the topic in the `.hhp` file for your Microsoft HTML Help project by completing the following steps:
    a. On the **View** menu, click **Output Directory**.
    b. Open the `ProjectName` folder, where *`ProjectName`* is the name of your project.
    c. Open the `ProjectName.hhp` file where *`ProjectName`* is the name of your project.
    d. On the **Contents** tab, select a table of contents entry, and then click the **Pencil** icon.
    e. On the **Advanced** tab, in the **Image index** field, use the up and down arrows to identify the table of contents image you want to use for the topic.
    f. Note the number of the image you want to use for the table of contents image for the topic.

    For example, if you want to use a question mark icon with a red star for the table of contents icon for new topics, note that the number for this icon is 10.

g. Close HTML Help Workshop.

2. ***If you want to specify a custom table of contents icon for Oracle Help or Sun JavaHelp***, create the graphic file for the custom table of contents icon in `.gif` format. The default graphics used as Sun JavaHelp or Oracle Help table of contents icons are 17 x 17 pixels. The custom graphics you create for Sun JavaHelp or Oracle Help table of contents icons should also be 17 x 17 pixels. You can assign any name to the graphic files.

3. ***If you want to specify a custom table of content icon for WebWorks help***, create graphics files containing the collapsed and expanded versions of the icons you want to use, then save the graphic files in `.gif` format. The default graphics used as WebWorks Help table of contents icons are 17 x 17 pixels. The custom graphics you create for WebWorks Help table of contents icons should also be 17 x 17 pixels. You can assign any name to the graphic files.

4. Copy the graphic files you want to use as icons in the table of contents into the following folder:

   Note: If the folder does not exist, first create the folder using the specified folder structure and then copy the graphic files you want to use as icons into the folder. You do not need to perform this step when specifying custom table of contents icons for Microsoft HTML Help.

   - ***If you are generating Oracle Help***, copy the graphic files you want to use into the following folder:

     ProjectName`\Formats\Oracle Help\Files\images` folder, where *ProjectName* is the name of your project.

   - ***If you are generating Sun JavaHelp 1.1.3***, copy the graphic files you want to use into the following folder:

     ProjectName`\Formats\Sun Java Help 1.1.3\Files\images` folder, where *ProjectName* is the name of your project.

   - ***If you are generating Sun JavaHelp 2.0***, copy the graphic files you want to use into the following folder:

     ProjectName`\Formats\Sun Java Help 2.0\Files\images` folder, where *ProjectName* is the name of your project.

   - ***If you are generating WebWorks Help***, in your `ProjectName\Files` folder, where *ProjectName* is the name of your project, create a `wwhelp\images` subfolder and copy the graphic files you want to use into this folder. Your project file structure should be similar to the following structure:

     ProjectName`\Files\wwhelp\images,` where *ProjectName* is the name of your project.

5. In your Adobe FrameMaker source document, locate the topic where you want to use the custom table of contents icon.

6. Insert your cursor into the heading for the topic.

7. On the **Special** menu, click **Marker**.

8. In the **Marker Type** field, select the appropriate TOCIcon marker type from the drop-down list.

9. ***If the appropriate TOCIcon marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to "Implementing Online Features in FrameMaker".

10. In the **Marker Text** field, type the following text:

    - ***If you are generating Microsoft HTML Help***, type the number of the icon that you want to use for the table of contents image.

      For example, if you want to use a question mark icon with a red star for the table of contents icon for new topics, type `10`.

    - ***If you are generating Oracle Help or Sun JavaHelp***, type the following text:

      `images/TOCIcon.gif`

      where `TOCIcon.gif` is the name of the table of contents icon you want to display for the topic.

    - ***If you are generating WebWorks Help***, type the following text:

      `c="collapsed.gif" e="expanded.gif"`

      where `collapsed.gif` is the name of the icon you want to use when the table of contents entry is collapsed, and `expanded.gif` is the name of the icon you want to use when the table of contents entry is expanded. If the table of contents entry is for a page instead of a book, the entry will never be expanded, so you can omit the `e="expanded.gif"` portion of the entry for pages.

      For example, you might create a special icon to highlight books that are new for a particular release of your WebWorks Help system. If you named these icons `newbookopen.gif` and `newbookclosed.gif`, you would type the following text into the **Value** field:

      ```
      c="newbookclosed.gif" e="newbookopen.gif"
      ```

11. Click **New Marker**.

12. Save your Adobe FrameMaker source document.

13. Generate output for your project. For more information, see "Generating Output".

14. In Output Explorer, verify ePublisher created the table of contents using the table of contents icon you specified for the topic. For more information about viewing output files in Output Explorer, see "Viewing Output in Output Explorer".

Specifying Context Plug-ins in FrameMaker

You can specify Eclipse Help context plug-ins by using Context Plugin markers in your source documents. ePublisher places the context plug-ins you specify in your source documents in the `plugin.xml` file generated for each source document group you have in Document Manager. You can then have developers use the context plug-ins defined in `plugin.xml` files to call your Eclipse Help system as appropriate from Eclipse plug-ins.

For example, assume you have the following three top-level groups in Document Manager for your Eclipse Help system target:

- Component A group - contains the source documents for ComponentA Feature1 and ComponentA Feature2
- Component B group - contains the source documents for ComponentB Feature1 and ComponentB Feature 2
- Component C group - contains the source documents for ComponentC Feature1 and ComponentC Feature 2

You insert the following Context Plugin markers into the source documents for each group:

- ComponentAFeature1 and ComponentAFeature2 Context Plugin markers in source documents contained in the ComponentA group
- ComponentBFeature1 and ComponentBFeature2 Context Plugin markers in source documents contained in the ComponentB group
- ComponentCFeature1 and ComponentCFeature2 Context Plugin markers in source documents contained in the ComponentC group

When you generate your Eclipse Help system, ePublisher creates the following folder structure in the *`ProjectName`*`\Output\`*`TargetName`* folder, where *ProjectName* is the name of your ePublisher project, and *TargetName* is the name of your target:

- `ComponentA` folder, which contains a `plugin.xml` file with the following entries:

  ```
  plugin="ComponentAFeature1ContextPlugin"
  plugin="ComponentAFeature2ContextPlugin"
  ```

- `ComponentB` folder, which contains a `plugin.xml` file with the following entries:

  ```
  plugin="ComponentBFeature1ContextPlugin"
  plugin="ComponentBFeature2ContextPlugin"
  ```

- `ComponentC` folder, which contains a `plugin.xml` file with the following entries:

  ```
  plugin="ComponentCFeature1ContextPlugin"
  plugin="ComponentCFeature2ContextPlugin"
  ```

You can then provide the context plug-in IDs in your `plugin.xml` files to the appropriate Eclipse developers to use. The Eclipse developers use the context plug-ins defined in `plugin.xml` files to call your Eclipse Help system as appropriate from Eclipse plug-ins.

To specify a context plug-in in an Adobe FrameMaker source document

1. Identify a topic in a source document where you want to insert the context plug-in.
2. On the **Special** menu, click **Marker**.
3. In the **Marker Type** field, select **Context Plugin** from the drop-down list.
4. ***If the Context Plugin marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to "Implementing Online Features in FrameMaker".
5. In the **Marker Text** field, type the appropriate ID for the context plug-in.

   Note: If you are responsible for defining the ID, ensure you supply the context plug-in ID to your developers to use as appropriate for their Eclipse plug-ins. If your developers define the ID, use the context plug-in ID you obtained from your developers.
6. Click **New Marker**.
7. Save your Adobe FrameMaker source document.
8. Generate output for your project. For more information, see "Generating Output".
9. In Output Explorer, verify ePublisher generated a `plugin.xml` file that contains the context plug-in IDs you specified by completing the following steps:
   a. On the **View** menu, click **Output Directory**.
   b. Open the *`ProjectName`* folder, where *`ProjectName`* is the name of your project.
   c. Open the group folder for a group that contains the source documents where you specified your context plug-in ID.
   d. Open the `plugin.xml` file in Notepad and verify that the context plug-in IDs you specified in your source documents are listed in the `plugin.xml` file. Your context plug-in IDs should be listed in the Contexts area of the file. Following is an example of the how the context plug-in IDs you specified in your source documents should be displayed in the `plugin.xml` file:

```
<!-- Contexts -->
<!-- -->
<extension point="org.eclipse.help.contexts">
<contexts file="contexts.xml" plugin="ComponentAFeature1ContextPlugin" />
</extension>
<extension point="org.eclipse.help.contexts">
<contexts file="contexts.xml" plugin="ComponentAFeature2ContextPlugin" />
```

```
    </extension>
```

Creating Accessible Online Content in FrameMaker

Accessible content is content that can be easily accessed by users with certain disabilities. This section explains how you can prepare your Adobe FrameMaker source documents to ensure your content is accessible to users using assistive technologies.

Accessible Content in FrameMaker

Images and tables are helpful ways to convey information to end users. However, users with disabilities often cannot access the important information provided by images and table layouts in online content. You should document images and other non-text items such as table layouts so that users using assistive technologies to access online content can access the information these items provide.

Content that must easily be accessed by people with disabilities must conform to certain guidelines published by both the W3C and the United States government in order to produce accessible online output, also known as Section 508 compliant output. These guidelines are intended to help writers produce accessible content.

You can use ePublisher to help you produce online content that conforms to the W3C Web Content Accessibility Guidelines 1.0 (WCAG), Section 508 of the U.S. Rehabilitation Act of 1998, and the Americans with Disabilities Act (ADA). If you are required to generate accessible content, typically you provide the following items in your online content:

- Alternate text and descriptions for all images and image maps. For more information, see "Assigning Alternate Text to Images and Image Maps in FrameMaker".
- Long descriptions for all images. For more information, see "Assigning Long Descriptions to Images in FrameMaker".
- Summaries for all tables. For more information, see "Assigning Alternate Text (Summaries) to Tables in FrameMaker".

You may also choose to provide the following items in your online content:

- Alternate text for abbreviations. For more information, see "Assigning Alternate Text to Abbreviations in FrameMaker".
- Alternate text for acronyms. For more information, see "Assigning Alternate Text to Acronyms in FrameMaker".
- Citations for quotes. For more information, see "Providing Citations for Quotes in FrameMaker"

You must prepare source documents and configure your ePublisher project in order to create accessible content. You prepare your source documents by inserting markers into your source documents and by applying character formats and paragraph formats. You configure accessibility settings in your ePublisher project. ePublisher uses the information in your source documents and your ePublisher project to generate accessible online output.

For more information about producing accessible content and to check your content further for compliance, see the following Web sites:

- For the complete W3C note on the WCAG, visit http://www.w3c.org/TR/WCAG10-CORE-TECHS.
- For information about the related Web Accessibility Initiative, visit http://www.w3.org/WAI.
- For information about Section 508 of the U.S. Rehabilitation Act of 1998, visit http://www.w3.org/WAI/Policy/#508.

Accessible Content Navigation in FrameMaker

Users can navigate through the accessible content using keys on the keyboard. The following output formats support navigation keys:

- Dynamic HTML
- Microsoft HTML Help
- Oracle Help
- WebWorks Help

Note: For the Dynamic HTML, navigation key behavior may vary based on the browser the user uses. For example, in Netscape and Mozilla, users must hold down the `Alt` key while pressing the navigation keys. In Internet Explorer, users must first hold down the `Alt` key while pressing the navigation key, and then press `Enter.`

The following table lists the how each output format supports navigation keys.

Navigation Key	Function	Format
1	Display the TOC	• Dynamic HTML • WebWorks Help 5.0
2	Display the Index	• Dynamic HTML • WebWorks Help 5.0
3	Display the Search tab	WebWorks Help 5.0
4	Go to the previous page	• Dynamic HTML • Microsoft HTML Help • Oracle Help • WebWorks Help 5.0 ***If you are using Microsoft HTML Help***, `Alt+4` works only if the topic pane has the focus. If the topic pane does not have the focus, you must press `Alt+0` and then `Alt+4`. ***If you are using Oracle Help***, you must press Enter after pressing `Alt +4`.
5	Go to the next page	• Dynamic HTML • Microsoft HTML Help 1.x • Oracle Help • WebWorks Help 5.0 ***If you are using Microsoft HTML Help***, the `Alt+5` key works only if

Navigation Key	Function	Format
		the topic pane has the focus. If the topic pane does not have the focus, you must press `Al+ 0` and then `Alt +5`. ***If you are using Oracle Help***, you must press `Enter` after pressing `Alt+5`.
6	Shift the focus to the related topics list displayed at the bottom of the current page	WebWorks Help 5.0 After you press the `6` key, you can press `Tab` to cycle through the entries in the related topics list.
7	Display a blank feedback e-mail (equivalent to clicking the e-mail button in the toolbar frame)	WebWorks Help 5.0
8	Print the current page (equivalent to clicking the Print button in the toolbar frame)	WebWorks Help 5.0
9	Bookmark the current page (equivalent to clicking the Bookmark button in the toolbar frame)	WebWorks Help 5.0
10	Shift the focus to the topic frame (equivalent to clicking within the topic frame)	WebWorks Help 5.0

Validating Accessible Content in FrameMaker

After you configure your source documents and configure the appropriate settings, ePublisher uses Accessibility conformance reports to perform the following checks to verify that the generated output conforms to accessibility standards:

- Alternate text for all images
- Alternate text for all clickable regions in all image maps
- Long descriptions for all images
- Summaries for all tables

ePublisher does not verify that you have provided alternate text for abbreviations or acronyms or verify that you have included citations for quotes. For more information about understanding and using the Accessibility conformance reports ePublisher provides, see “Configuring Reports”, and “Generating Reports”.

Assigning Alternate Text to Images and Image Maps in FrameMaker

This section provides information about how to create accessible images and image maps in your generated output by assigning alternate text to images.

Image and Image Map Alternate Text in FrameMaker

One of the largest accessibility challenges with online content today is the lack of alternative text for images and image maps. Sight-impaired users often use screen readers or refreshable Braille devices to read online content. However, when these assistive technologies come across images or image maps without alternative text, also known as alternate text, they are unable to provide users with information about the image or image map and its meaning.

The Web Content Accessibility Guidelines require that alternate text be provided for all images and image maps in online content. The alternate text is an image label that describes the image or each area of the image map. Online content should display alternate text for images and image maps when users perform the following actions:

- The user hovers the mouse pointer over an image or section of an image map.
- The user browser has been configured to disable display of images and image maps.
- The user browser is a text-only browser such as Lynx.
- The user uses assistive technology such as a screen reader.

The alternate text you assign to an image or sections of an image map should be as accurate and as succinct as possible and provide users with a brief description of the image and how the image relates to the page they are viewing. Make sure that your alternate text conveys all of the important information related to the image or image map section, but do not burden users with excessively long alternative text. Screen readers or refreshable Braille devices always read the alternative text, so if your page has several images or complex image maps with long descriptions, it can take a long time for the assistive devices to read image-heavy pages with long descriptions. If you need to provide a description of the image or image map section that is more than a few words or a few short sentences, you should provide a brief alternate text description of the image or image map section and then assign a longer description the image using either the `longdesc` attribute or a description. Once you specify a long description using the `longdesc` attribute, you can also optionally display a D link next to the image. For more information about assigning long descriptions to images, see “Assigning Long Descriptions to Images in FrameMaker”.

Assigning Alternate Text to Images in FrameMaker

To assign alternate text to an image, your Stationery and template must have the ImageAltText marker type configured. Your output format must also support this feature.

The following procedure provides an example of how to assign alternate text to images in the most current version of FrameMaker. The Object Attribute method can be used with newer versions of FrameMaker.

To assign alternate text to an image in an Adobe FrameMaker source document using object attributes

1. In your Adobe FrameMaker source document, locate the anchored frame for the image to which you want to assign alternate text.
2. In the anchored frame that contains the image to which you want to assign alternate text, and complete the following:
    a. Select the anchored frame that contains the image to which you want to assign an alternate name.
    b. On the **Graphics** menu, click **Object Properties**.
    c. Click **Object Attributes**.
    d. In the **New or Changed Attribute** area, in the **Name** field, type **ImageAltText**.
    e. In the **Definition** field, type the alternate text you want to assign to the image. Your text cannot exceed 255 characters.
    f. Click **Add**.
    g. Click **Set**.
    h. Click **Set** again to close the window.

To assign alternate text to an image in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the anchored frame for the image to which you want to assign alternate text.
2. In the anchored frame that contains the image to which you want to assign alternate text, insert a text frame by completing the following steps:
    a. On the **Graphics** menu, click **Tools** to display the graphic tools palette.
    b. Click the **Text Frame** icon.
    c. Drag the cursor over the portion of the image where you want to insert the text frame that will contain the ImageAltText marker.
    d. In the Create New Text Frame window, in the **Number** field, type `1`, and then click **Set**.

e. Click outside the image, and then insert your cursor in the text frame.

3. On the **Special** menu, click **Marker**.
4. In the **Marker Type** field, select **ImageAltText** from the drop-down list.
5. ***If the ImageAltText marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to "Implementing Online Features in FrameMaker".
6. In the **Marker Text** field, type the alternate text you want to assign to the image.
7. Click **New Marker**.
8. Save your Adobe FrameMaker source document.
9. Generate output for your project. For more information, see "Generating Output".
10. Verify ePublisher assigned the alternate text you specified to the image when it generated output by completing the following steps:
    a. On the **View** menu, click **Output Directory**.
    b. In the *`TargetName`* folder, open the page that has the image to which you assigned alternate text in Notepad, where *`TargetName`* is the name of your target.
    c. Verify that the alternate text you specified is included in the `alt` tag for the image.

Assigning Alternate Text to Image Maps in FrameMaker

To assign alternate text to an image, your Stationery and template must have the ImageAreaAltText marker type configured. Your output format must also support this feature.

The following procedure provides an example of how to assign alternate text to an image map in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for assigning alternate text to an image map in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To assign alternate text for an image map in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the anchored frame for the image map to which you want to assign alternate text.
2. In the anchored frame that contains the image map to which you want to assign alternate text, complete the following steps for *each* area of an image map:
    a. Insert your cursor into the text frame that defines a clickable region on the image map.
    b. On the **Special** menu, click **Marker**.
    c. In the **Marker Type** field, select **ImageAreaAltText** from the drop-down list.
    d. ***If the ImageAreaAltText marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.
    e. In the **Marker Text** field, type the alternate text you want to assign to the image map area.
    f. Click **New Marker**.
3. Save your Adobe FrameMaker source document.
4. Generate output for your project. For more information, see “Generating Output”.
5. Verify ePublisher assigned the alternate text you specified to each area of the image map when it generated output by completing the following steps:
    a. On the **View** menu, click **Output Directory**.
    b. In the *`TargetName`* folder, open the page that has the image map to which you assigned alternate text in Notepad, where *`TargetName`* is the name of your target.
    c. Verify that the alternate text you specified is included in the `alt` tag for each area of the image map.

Assigning Long Descriptions to Images in FrameMaker

This section explains how to create accessible images in your generated output by assigning long descriptions to images.

Image Long Descriptions in FrameMaker

The Web Content Accessibility Guidelines and Section 508 guidelines require you to include long descriptions for each image in an HTML document. You can use the `longdesc` attribute and a long descriptions stored in an external `.txt` file to assign a long description to an image. When you use this approach, the long descriptions are referenced in the HTML `<img>` tag in the `longdesc` attribute as shown in the following example:

```
<img src="mission.gif" height="240" width="386" alt="The Mission"
longdesc="mission.txt" />
```

The `longdesc` attribute in the `<img>` tag provides a link to a separate page where a long description is available. The link is invisible to sighted users, but when a conformant screen reader application reads the `longdesc` attribute, it loads the file referenced in the `longdesc` attribute and reads it. In the previous example, the screen reader would load and read the `mission.txt` file.

ePublisher provides the following options for assigning long descriptions to images:

- You can use the ImageLongDescText marker to assign a long description to an image. With this method, you assign a long description to an image using a description you include in a marker you insert into your source document. For more information, see "Assigning Long Descriptions to Images in FrameMaker".
- You can use the ImageLongDescByRef marker to assign a long description to an image by referencing a long description saved in an external text (`.txt`) file. With this method, you specify the path to the external text file in a marker. For more information, see "Using Text in External Files to Assign Long Descriptions to Images in FrameMaker".

If you assign long descriptions to some, but not all of you images, you can use the ImageLongDescNotReq marker. Use this marker when you use accessibility reports to verify that all images have long description but you have certain images in your source document that do not require a long description. For more information, see "Excluding Images from Accessibility Report Checks in FrameMaker".

Although using the `longdesc` attribute is recommended in the Web Content Accessibility Guidelines and in 508 guidelines, older screen readers and many current browsers do not support this attribute and few online content developers use this attribute. As a result, the `longdesc` attributed benefits a only a small number of users. Only users who use modern screen readers can access the `longdesc` attribute easily. Older screen readers did not support this attribute. In addition, even users who use the latest version of screen reader may be unfamiliar with the `longdesc` attribute and may not know how to access long descriptions using their screen reader because the `longdesc` attribute is used so infrequently in online content.

If you use the ImageLongDescText marker to assign long descriptions to images, as an interim solution ePublisher allows you to display a D link immediately after the image. The D link is an upper case letter D link that directs users to another page that contains the text you specified in the ImageLongDescText marker. Although a D link is not required for accessible Web pages, it can be used in addition to the `longdesc` attribute. The D link technique works in all browsers, but it is less elegant than using the `londesc` attribute. Some users may be confused when they see a D link on the page, while other users will ignore the D link.

If you want to use D links in addition to the `longdesc` attribute when you generate output, your Stationery must have the D link option enabled. If you have permissions to modify target settings in ePublisher Express, you can enable the D link option setting in an ePublisher Express project. For more information about enabling the D link option in an ePublisher project, see “Specifying Accessibility Settings”. For more information about permissions required to modify target settings using ePublisher Express, see “Working with Target Settings”.

Specifying Long Descriptions for Images in FrameMaker

To assign a long description to an image, your Stationery and template must have the ImageLongDescText marker type configured. Your output format must also support this feature.

When you use the ImageLongDescText marker to assign long descriptions to images, ePublisher generates an external text file that contains the long description you specify. When a conformant screen reader application reads the generated page, it loads the `.txt` file referenced in the `longdesc` attribute on the page and reads the file.

You can use the ImageLongDescText marker to assign long descriptions to images if your image descriptions do not exceed 255 characters. If your image long descriptions are longer than 255 characters, you must use an external `.txt` file and the ImageLongDescByRef marker to assign long descriptions to images, because Adobe FrameMaker limits the length of marker text to 255 characters. For more information, see "Using Text in External Files to Assign Long Descriptions to Images in FrameMaker".

In addition, Adobe FrameMaker ignores carriage returns in marker text when generating MIF files. As result, if you use ImageLongDescText markers, each long description will be generated as a single paragraph.

The steps you use to assign long descriptions to images varies based on the version of Adobe FrameMaker you use. If you use Adobe FrameMaker 6.0, you use the ImageLongDescText marker. If you use Adobe FrameMaker 7.0 or later, you can use either the ImageLongDescText marker or an object attribute you create for the anchored frame to assign a long description to an image. For more information about long descriptions and D links, see "Specifying Long Descriptions for Images in FrameMaker".

To assign a long description to an image using marker text in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the anchored frame for the image to which you want to assign a long description.
2. ***If you are using Adobe FrameMaker 6.0***, complete the following steps:
    a. On the **Graphics** menu, click **Tools** to display the graphic tools palette.
    b. Click the **Text Frame** icon.
    c. Drag the cursor over the portion of the image where you want to insert the text frame that will contain the ImageLongDescText marker.
    d. In the Create New Text Frame window, in the **Number** field, type `1`, and then click **Set**.
    e. Click outside the image, and then insert your cursor in the text frame.
    f. On the **Special** menu, click **Marker**.
    g. In the **Marker Type** field, select **ImageLongDescText** from the drop-down list.
    h. ***If the ImageLongDescText marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this

functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.

i. In the **Marker Text** field, type the long description you want to assign to the image. Your description cannot exceed 255 characters.

j. Click **New Marker**.

3. ***If you are using Adobe FrameMaker 7.0***, complete the following steps:

   a. Select the anchored frame that contains the image to which you want to assign a long description using an external file.

   b. On the **Graphics** menu, click **Object Properties**.

   c. Click **Object Attributes**.

   d. In the **New or Changed Attribute** area, in the **Name** field, type **ImageLongDescText**.

   e. In the **Definition** field, type the long description you want to assign to the image. Your description cannot exceed 255 characters.

   f. Click **Add**.

   g. Click **Set**.

   h. Click **Set** again to close the window.

4. Save your Adobe FrameMaker source document.

5. Generate output for your project. For more information, see “Generating Output”.

6. Verify ePublisher assigned the long description to the image by completing the following steps:

   a. On the **View** menu, click **Output Directory**.

   b. In the *TargetName*`\images` folder, verify that ePublisher created a `.txt` file that contains the long description you specified in the ImageLongDescText marker, where *TargetName* is the name of your target.

      For example, if you specified a long description for *ImageName*`.png`, verify that ePublisher created an *ImageName*`.txt` file in the `images` folder, where *ImageName* is the name of the image to which you assigned a long description.

   c. In the *TargetName*`\`*ProjectName* folder, open the page that contains the image to which you assigned the long description in Notepad and verify that the `longdesc` attribute references the *ImageName*`.txt` file ePublisher created for the image, where *TargetName* is the name of your target, *ProjectName* is the name of your project, and *ImageName* is the name of the image to which you assigned a long description.

   d. ***If you used the ImageLongDescText marker and the Stationery designer configured your Stationery to support D links***, open the page in a browser, verify that the D link displays in the browser, and then click the D link and verify that a page opens that displays the long description that you specified in the **ImageLongDescText** marker.

Using Text in External Files to Assign Long Descriptions to Images in FrameMaker

Assign long descriptions to images using external files when you have image descriptions that exceed 255 characters or if you want to use image descriptions in external text files to assign long descriptions to images.

To assign a long description to an image, your Stationery and template must have the ImageLongDescText marker type configured. Your output format must also support this feature.

The steps you use to assign long descriptions to image varies based on the version of Adobe FrameMaker you use. If you use Adobe FrameMaker 6.0, you use the ImageLongDescByRef marker. If you use Adobe FrameMaker 7.0 or later, you use an object attribute you create for the anchored frame to assign a long description to an image.

To assign a long description to an image using an external file in an Adobe FrameMaker source document

1. Create a `.txt` file that contains each image long description.
2. Place each image long description text file in a folder in the *`ProjectName`*`\Formats\`*`TargetName`*`\Files` folder for your project, where *`ProjectName`* is the name of your ePublisher project and *`TargetName`* is the name of your target.

   For example, place the each image long description in the following location:

   `ProjectName``\Formats\`*`TargetName`*`\Files\`*`longdescriptions`*`\`*`imagelongdescription`*`.txt`

   where *`ProjectName`* is the name of your ePublisher project, *`TargetName`* is the name of your target, *`longdescriptions`* is the name of the folder where you placed the image long description, and *`imagelongdescription`* is the name of the `.txt` file that contains the image long description.
3. In your Adobe FrameMaker source document, locate the anchored frame for the image to which you want to assign a long description.
4. Complete the following steps:
   a. Select the anchored frame that contains the image to which you want to assign a long description using an external file.
   b. On the **Graphics** menu, click **Object Properties**.
   c. Click **Object Attributes**.
   d. In the **New or Changed Attribute** area, in the **Name** field, type **ImageLongDescByRef**.
   e. In the **Definition** field, type the path to the `.txt` file that contains the long description you want to assign to the image.

      For example, type:

      `./`*`longdescriptions`*`/`*`imagelongdescription`*`.txt`

where *longdescriptions* is the name of the folder where you placed the image long description, and *imagelongdescription* is the name of the `.txt` file that contains the image long description.

f. Click **Add**.

g. Click **Set**.

h. Click **Set** again to close the window.

5. Save your Adobe FrameMaker source document.

6. Generate output for your project. For more information, see “Generating Output”.

7. In Output Explorer, verify ePublisher assigned the long description to the image using the long description in the external file when it generated output by completing the following steps:

a. On the **View** menu, click **Output Directory**.

b. In the *TargetName*`\`*ProjectName* folder, open the page that contains the image to which you assigned the long description using an external file in Notepad and verify that the `longdesc` attribute references the external text file that contains the long description for the image, where *TargetName* is the name of your target, and *ProjectName* is the name of your project.

Excluding Images from Accessibility Report Checks in FrameMaker

In some instances, alternate text is sufficient for an image, and assigning a long description to an image in addition to alternate text would be redundant. However, you may have configured Accessibility reports to check for images without long descriptions and notify you when an image does not have a long description.

In this scenario, while you want an Accessibility report to notify you when you have an image without a long description, you do not want to be notified when you deliberately did not assign a long description to an image because assigning a both a long description and alternative text would be redundant. To address this issue, you can use the ImageLongDescNotReq marker to exclude an image that deliberately does not have a long description from validation when you generate Accessibility reports. For more information about Accessibility reports and configuring and generating Accessibility reports, see “Accessibility Reports”, “Configuring Reports”, and “Generating Reports”.

To exclude images without long descriptions from Accessibility reports, your Stationery and template must have the ImageLongDescNotReq marker type configured. Your output format must also support this feature.

The following procedure provides an example of how to exclude images without long descriptions from Accessibility report checks in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for excluding images without long descriptions from Accessibility report checks in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To exclude an image without a long description from Accessibility report checks in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the anchored frame for the image without a long description that you want to exclude from an Accessibility report check.
2. On the **Graphics** menu, click **Tools** to display the graphic tools palette.
3. Click the **Text Frame** icon.
4. Drag the cursor over the portion of the image where you want to insert the text frame that will contain the ImageLongDescNotReq marker.
5. In the Create New Text Frame window, in the **Number** field, type `1`, and then click **Set**.
6. Click outside the image, and then insert your cursor in the text frame.
7. On the **Special** menu, click **Marker**.
8. In the **Marker Type** field, select **ImageLongDescNotReq** from the drop-down list.
9. ***If the ImageLongDescNotReq marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.

10. In the **Marker Text** field, do not enter any text. You do not need to enter any text in this field when you insert a ImageLongDescNotReq marker.
11. Save your Adobe FrameMaker source document.
12. Generate output for your project. For more information, see “Generating Output”.
13. Generate an Accessibility report and confirm that ePublisher did not generate an `Image is missing a long description` message for the image. For more information about generating Accessibility reports and Accessibility report messages, see “Generating Reports” and “Accessibility Report Messages”.

Assigning Alternate Text (Summaries) to Tables in FrameMaker

Tables, just like images, are a way to visually display information. Although tables typically contain text, the purpose of the table is often not evident from text alone. The organization and display of the table may contain information that is not evident to assistive technologies. However, through the use of table summaries, assistive technologies can convey useful information to users about tables. The Web Content Accessibility Guidelines recommend that you provide summary text for each table in an HTML document. Table alternate text, or table summaries, provide users with information about what type of information the table contains.

You can create accessible tables by typing the table summary into a TableSummary marker. When ePublisher generates content, ePublisher puts the table summary you specify into the table in the `summary` attribute.

To assign alternate text to tables, your Stationery and template must have the TableSummary marker type configured. Your output format must also support this feature.

The following procedure provides an example of how to assign alternate text to tables in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for assigning alternate text to tables in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To assign table summaries in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the table to which you want to assign a table summary.
2. Insert your cursor in front of the table.
3. On the **Special** menu, click **Marker**.
4. In the **Marker Type** field, select **TableSummary** from the drop-down list.
5. ***If the TableSummary marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to "Implementing Online Features in FrameMaker".
6. In the **Marker Text** field, type the table summary you want to assign to the table.
7. Click **New Marker**.
8. Save your Adobe FrameMaker source document.
9. Generate output for your project. For more information, see "Generating Output".
10. Verify ePublisher assigned the table summary you specified to the table when it generated output by completing the following steps:
    a. On the **View** menu, click **Output Directory**.
    b. In the *`TargetName`* folder, open the page that has the table to which you assigned a table summary in Notepad, where *`TargetName`* is the name of your target.

c. Verify that the table summary you specified is included in the `summary` attribute for the table.

Excluding Tables from Accessibility Report Checks in FrameMaker

Tables used specifically for layout may not need a table summary. For example, if you use a table for layout, you probably would not assign a table summary to the table. However, you may have configured Accessibility reports to check for tables without table summaries and notify you when a table does not have a table summary.

In this scenario, while you want an Accessibility report to notify you when you have a table without a table summary, you do not want to be notified when you deliberately did not assign a table summary to a table because a table summary is not required. To address this issue, you can use the TableSummaryNotReq marker to exclude a table that deliberately does not have a table summary from validation when you generate Accessibility reports. For more information about Accessibility reports and configuring and generating Accessibility reports, see “Accessibility Reports”, “Configuring Reports”, and “Generating Reports”.

To exclude tables from Accessibility report checks, your Stationery and FrameMaker template must have the TableSummaryNotReq marker type configured. Your output format must also support this feature.

The following procedure provides an example of how to exclude tables without table summaries from Accessibility report checks in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for excluding tables without table summaries from Accessibility report checks in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To exclude a table without a table summary from Accessibility report checks in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the table without a table summary that you want to exclude from an Accessibility report check.
2. Insert your cursor in front of the table.
3. On the **Special** menu, click **Marker**.
4. In the **Marker Type** field, select **TableSummaryNotReq** from the drop-down list.
5. ***If the TableSummaryNotReq marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.
6. In the **Marker Text** field, do not enter any text. You do not need to enter any text in this field when you insert a TableSummaryNotReq marker.
7. Click **New Marker**.
8. Save your Adobe FrameMaker source document.
9. Generate output for your project. For more information, see “Generating Output”.
10. Generate the Accessibility report and confirm that ePublisher did not generate an `Table is missing a table summary` message for the table. For more information about generating

Accessibility reports and Accessibility report messages, see “Generating Reports” and “Accessibility Report Messages”.

Assigning Alternate Text to Abbreviations in FrameMaker

Abbreviations are often used in written communication. Using an Abbreviation character format and an AbbreviationTitle marker, you can specify alternate text for abbreviations. For example, if your source document includes an abbreviation such as SS#, you can specify Social Security Number as alternate text for the abbreviation. When you use an AbbreviationTitle marker and Abbreviation character format to specify alternate text for an abbreviation, ePublisher adds the abbreviation alternate text you specify to the `title` attribute of the `abbr` tag in the output.

Following is an example of the HTML code produced when you specify Social Security Number as alternate text for SS#.

```
<th>First name</th>
<th><abbr title="Social Security Number">SS#</abbr></th>
```

To assign alternate text to abbreviations, your Stationery and template must have the following items configured:

- Abbreviation character format
- AbbreviationTitle marker type

Your output format must also support this feature.

The following procedure provides an example of how to specify alternate text for abbreviations in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for specifying alternate text for abbreviations in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To specify alternate text for an abbreviation in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the abbreviation for which you want to specify alternate text.
2. Apply the Abbreviation character format to the abbreviation text.
3. Insert your cursor anywhere inside the abbreviation.
4. On the **Special** menu, click **Marker**.
5. In the **Marker Type** field, select **AbbreviationTitle** from the drop-down list.
6. ***If the AbbreviationTitle marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.
7. In the **Marker Text** field, type the abbreviation alternate text.
8. Click **New Marker**.
9. Save your Adobe FrameMaker source document.

10. Generate output for your project. For more information, see “Generating Output”.
11. Verify ePublisher assigned the abbreviation alternate text you specified when it generated output by completing the following steps:
    a. On the **View** menu, click **Output Directory**.
    b. In the *`TargetName`* folder, open the page that has the abbreviation to which you assigned alternate text in Notepad, where *`TargetName`* is the name of your target.
    c. Verify that the alternate text you specified for the abbreviation is included in the `abbr` tag in the `title` attribute.

Assigning Alternate Text to Acronyms in FrameMaker

Acronyms are often used in written communication. Using an Acronym character format and an AcronymTitle marker, you can specify alternate text for acronyms. For example, if your document includes an acronym like NATO you can specify North Atlantic Treaty Organization as alternate text for the acronym. When you use an AcronymTitle marker and an Acronym character format to specify alternate text for an acronym, ePublisher adds the acronym alternate text you specify to the `title` attribute of the `acronym` tag in the output.

Following is an example of the HTML code produced when you specify North Atlantic Treaty Organization as alternate text for NATO.

```
<p><acronym title=”North Atlantic Treaty Organization”>NATO</acronym> is a military
 alliance.<p>
```

To assign alternate text to acronyms, your Stationery and template must have the following items configured:

- Acronym character format
- AcronymTitle marker type

Your output format must also support this feature.

The following procedure provides an example of how to specify alternate text for acronyms in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for specifying alternate text for acronyms in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To specify alternate text for an acronym in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the acronym for which you want to specify alternate text.
2. Apply the Acronym character format to the acronym text.
3. Insert your cursor anywhere inside the acronym.
4. On the **Special** menu, click **Marker**.
5. In the **Marker Type** field, select **Acronym** from the drop-down list.
6. ***If the Acronym marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.
7. In the **Marker Text** field, type the acronym alternate text.
8. Click **New Marker**.
9. Save your Adobe FrameMaker source document.
10. Generate output for your project. For more information, see “Generating Output”.

11. Verify ePublisher assigned the acronym alternate text you specified when it generated output by completing the following steps:

    a. On the **View** menu, click **Output Directory**.

    b. In the *`TargetName`* folder, open the page that has the acronym to which you assigned alternate text in Notepad, where *`TargetName`* is the name of your target.

    c. Verify that the alternate text you specified for the acronym is included in the `acronym` tag in the `title` attribute.

Providing Citations for Quotes in FrameMaker

A **citation** is a reference or footnote to a book, article, or other material that specifies the source from which a quotation was borrowed. A citation contains all the information necessary to identify and locate the work. Using a Citation character format and the Citation marker, you can specify citations for quotes that enable users to go to a Web site that contains additional information about the quote.

Following is an example of the HTML code produced when you specify a citation for a quote.

```
<blockquote cite="http://shakespeare.mit.edu/lll/full.html"> <p>Remuneration! O!
 that's the Latin word for three farthings.
--- William Shakespeare (Love's Labor Lost).</p> </blockquote>
```

To provide citations for quotes, your Stationery and template must have the following items configured:

- Citation character format
- Citation marker type

Your output format must also support this feature.

The following procedure provides an example of how to specify citations for quotes in Adobe FrameMaker source documents using unstructured Adobe FrameMaker 7.2. Steps for specifying citations for quotes in Adobe FrameMaker may be different in other versions of Adobe FrameMaker.

To specify citations for quotes in an Adobe FrameMaker source document

1. In your Adobe FrameMaker source document, locate the quotation for which you want to specify a citation.
2. ***If the quotation is a phrase within a paragraph***, complete the following steps:
    a. Apply the Citation character format to the quotation phrase.
    b. Insert your cursor anywhere inside the quotation phrase.
    c. On the **Special** menu, click **Marker**.
3. ***If the quotation is a full paragraph***, complete the following steps:
    a. Insert your cursor into the paragraph.
    b. On the **Special** menu, click **Marker**.
4. In the **Marker Type** field, select **Citation** from the drop-down list.
5. ***If the Citation marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality and then use the marker type specified by the Stationery designer. For more information, refer to “Implementing Online Features in FrameMaker”.
6. In the **Marker Text** field, type the URL for the citation.

7. Click **New Marker**.
8. Save your Adobe FrameMaker source document.
9. Generate output for your project. For more information, see “Generating Output”.
10. Verify ePublisher created the citation you specified when it generated output by completing the following steps:
    a. On the **View** menu, click **Output Directory**.
    b. In the *TargetName* folder, open the page that has the quotation for which you specified a quotation in Notepad, where *TargetName* is the name of your target.
    c. Verify that the citation you specified for the quotation is included in the `cite` attribute.

Troubleshooting FrameMaker issues

Occasionally there might be issues with the source documents you are using. Below is a list linking to the wiki solutions website that will help you troubleshoot each one:

Issue	For more information, see...
If you are trying to use smart quotes	Character Mapping in mapentrysets.xml
If you are trying to turn off auto-hyphenation	Turning off Auto-hyphenation
If you are using autonumbering in anchored frames	Autonumbered Paragraphs
If you receive a “Cannot Duplicate Document” upon generation	Cannot Duplicate Document
If you receive a “Cannot Initialize API” message opening FrameMaker	Cannot Initialize API
If you are working with Change Bars	Change Bars
If you receive the “Error Communicating with FrameMaker” message upon generating	Error Communicating with FrameMaker
If you are not seeing your filename Markers in ePublisher	Filename markers are not working
If you are seeing unwanted files in the temp directory upon generating	Files being created in Temp directory
If you are trying to use an image as a bullet	Using an image and text together as a paragraph bullet
If you are trying to import an XLS file into FrameMaker	Problems with Imported Excel files
If you are trying to link to an external PDF	Creating links outside of project structure
If you are using multiple TOC files in FrameMaker	FrameMaker books with multiple TOC, Index, or Front-matter files

Issue	For more information, see...
If your title page material is not ordered properly	Title page material not ordered properly
If you are having unexpected paragraph ordering with multiple text flows	FrameMaker paragraphs in different text flows have unexpected order in HTML
If you are wanting to turn unbulleted text to bulleted text in the output	How do I turn regular FrameMaker text into bulleted text in my output?
If you are working with structured content in text-boxes	Structured Elements in Text Boxes
If you notice that some paragraphs in your output are formatting differently than others bearing the same style name	Unexpected changes in font size or other formatting for certain paragraphs
If you notice a few lines of code that are very small at the end of the HTML page from the FM document,	Unexpected Code from FrameMaker plugin

Microsoft Word

If you want to implement online content features in your generated output, you need to prepare your Microsoft Word source documents for output generation. This section explains how to prepare your Microsoft Word source documents.

Microsoft Word Templates and Standards

Microsoft Word is a powerful authoring tool that allows you to quickly create professional content. You can develop templates with the styles and other standard elements you need to deliver polished technical documentation. Microsoft Word provides many automation features to streamline the content development process. The new XML-based formats allow Microsoft Word to integrate content with other Microsoft Office products.

This section describes the design considerations for a Microsoft Word template. By effectively designing your Microsoft Word template, and by consistently applying styles throughout your source documents, you can streamline single-sourcing processes and reduce your production and maintenance costs. This section does not describe all Microsoft Word processes, but it focuses on the design considerations related to ePublisher.

Word Standards to Support Single-Sourcing

To define your Microsoft Word standards, create a template `.dot` file with all the elements you need in it, including styles, variables, autotext, toolbars, macros, and page layouts. You can use this file to create all new source documents and to update the styles, autotext, toolbars, and macros in existing source documents. You can also use this file to create a source document to include in your Stationery design project.

The following sections describe various template areas and considerations, and how to effectively design your Microsoft Word template to support single-sourcing with ePublisher.

Microsoft Word Template File

The template `.dot` file defines the default styles, autotext, toolbars, and macros for documents attached to that template. You can also define the starting variables, properties, content, sections, and page layouts for new documents created from the template. The existing bulleted and numbered list items in the template file preload the bullet and number gallery in Microsoft Word, which can cause inconsistencies on multiple computers when these items are not properly defined in the template.

When you create a new document based on a template, the contents of the template file, including text, paragraphs, sections, headers, and footers are copied to the new document. In addition, the new document is attached to the template and uses the styles, autotext, toolbars, and macros defined in the template.

You can also attach an existing document to a template. The document can then use the autotext, toolbars, and macros defined in the template. If you select the **Automatically update document styles** option, the style definitions in the template overwrite the style definitions in the document. The existing document content is not changed, including the headers, footers, and page layout. Style modifications on individual paragraphs, such as the Keep with Next setting, are also not changed or reset. You can create a macro to find each style and reset the paragraph to the default style, which removes the modifications.

Creating a Clean Base Template File

When you create custom styles and macros in Microsoft Word, these customizations are often stored in the default `Normal.dot` file on your computer. In addition, when you use Microsoft Word as your email editor in Microsoft Outlook, some other customizations can be stored in the default `Normal.dot` file. When you create a new Microsoft Word template, you want to use a clean `Normal.dot` file as the starting point for your template. Save the template file with a new name, and then customize the template as needed. To simplify your template use and maintenance, delete the unneeded styles, variables, autotext, and macros.

To create a clean base template file

1. On the **Tools** menu in Microsoft Word, click **Options**.
2. On the **File Locations** tab, find where the User Templates are stored.
3. Close Microsoft Word and all other Microsoft Office products.
4. Open the folder where the User Templates are stored and rename the `Normal.dot` file in that location in case you later need any customizations stored in that file.
5. Open Microsoft Word.
6. On the **File** menu, click **Save As**.
7. In **Save as type**, select **Document Template (*.dot)**.
8. Specify the name and location for your new template, and then click **Save**.

You can customize the new template to meet your specific needs. To create a new tab in the Template Selection window in Microsoft Word, you can create a folder within the User Templates location. Then, you can store your new template in that folder.

Paragraph Styles in Word

Create paragraph styles for items based on function, not based on formatting. This approach allows you to modify formatting over time and the style names continue to apply. It also prepares you for structured writing in the future.

Name your paragraph styles starting with naming conventions that group styles by function. For example, group procedure-related styles together by starting the style names with Procedure, such as ProcedureIntro, ProcedureStep, and ProcedureSubstep.

Note: Style names should not include a period in their name. The period can cause display issues when ePublisher creates the cascading style sheet entry that defines the appearance of the style.

To simplify formatting and save time for future maintenance and customization, set the default paragraph font and spacing for a base style, such as Normal. Then, base other styles on this base style to inherit the default formatting settings. This process allows you to quickly modify fonts and spacing across styles by modifying only the base style. You can customize settings for each style as needed. The customized settings are not affected when you modify those settings in the base style. To simplify maintenance for heading styles, which often use a different font than your content styles, you may want to base all heading styles on the Heading 1 style to define the font for all headings.

In ePublisher, you can scan the source documents to list all the paragraph styles. Then, you can organize them in ePublisher to allow property inheritance and to streamline the customization process for your generated output.

You may need multiple paragraph styles to define functions that support pagination settings, such as a BodyListIntro format that has **Keep with next** set. To reduce the number of paragraph styles, you can customize paragraphs to add the **Keep with next** setting as needed. Customizing this setting on a paragraph does not affect the ability for the paragraph to receive the other formatting settings from the style definition.

To automate and simplify template use, define the paragraph style that follows each paragraph style. This process allows the writer to press Enter after writing a paragraph and the template creates the next paragraph with the style most commonly used next. For example, after a Heading style, the writer most often writes a body paragraph of content.

Common paragraph styles include:

- Figure paragraphs. You may need multiple indents, such as Figure, FigureInList, and FigureInList2.
- Body paragraphs. You may need multiple indents, such as Body, BodyInList, and BodyInList2. To reduce training needs, you can use the default style names, such as Body Text, Body Text Indent, and Body Text Indent 2.
- Headings, such as Heading 1, Heading 2, Heading 3, and Heading 4. You may also need specialized headings, such as Title, Subtitle, FrontMatterHeading1, FrontMatterHeading2, and FrontMatterHeading3. The cross-reference feature in Microsoft Word allows you to create cross references to headings that use the default Heading styles named Heading *X*, where *X* is a number. To create cross references to other styles, use bookmarks. Do not paste content at the beginning of

a heading. Existing cross references to that heading may include the pasted content when the cross references are updated.

- Bulleted lists. You may need multiple bullet levels, such as Bullet, Bullet2, Bullet3. You may also need a bullet item within a procedure, such as a ProcedureBullet and a bullet item within a table, such as a CellBullet. For more information, see “Bulleted and Numbered Lists in Word”.
- Numbered lists. You may need multiple levels, such as ProcedureStep that uses numbers and ProcedureSubstep that uses lowercase letters. You may also need numbered list items in tables, such as CellStep. Be sure to consider related supporting formats, such as ProcedureIntro. For more information, see “Bulleted and Numbered Lists in Word”.
- Examples, such as code or command syntax statements, usually in a fixed font. To keep the lines of a code example together, you can set the **Keep with next** setting for the Example style and use an ExampleLast style to identify the end of the example. You may also need multiple example levels, such as ExampleInList and ExampleInListLast.
- Paragraphs in tables, such as CellHeading, CellBody, CellBody2, CellStep, and CellBullet.
- Legal notice and copyright or trademark styles for inside the cover page.
- Table of contents and Index styles.
- Definition lists, such as term and definition or description. You can use a two-column table for this purpose, but a definition list allows long terms, such as field labels in a user interface, to run across the page without wrapping. Then, the definition or description are indented below the term.
- Header and footer styles to control formatting.
- Notes, cautions, tips, and warnings.

ePublisher projects use custom field code markers, paragraph styles, and character styles to define online features. You need to give the list of markers and styles to the writers so they know how to implement each online feature. The writers use the markers and styles you create to define online features.

The Stationery defines the custom markers and styles. To reduce complexity, you can use the style names defined in the documentation, or you can define the online feature to a different style. The following list identifies additional paragraph styles you may need to support ePublisher online content features:

- Paragraph or character styles to support multiple languages, such as bidirectional languages and text.
- Dropdown paragraph style that identifies the start of an expand/collapse section. You can end the section with a paragraph style defined to end the section, or with a DropDownEnd marker.
- Popup paragraph styles that define several aspects of popup window content:
  - Popup paragraph style identifies the content to display in a popup window and in a standard help topic. This style is applied to the first paragraph of popup content.
  - Popup Append paragraph style identifies the content to display in a popup window and in a standard help topic. This style is applied to additional popup paragraphs when you have more than one paragraph of content to include in a popup window.

  - Popup Only paragraph style identifies the content to display only in a popup window. This style is applied to the first paragraph of popup content.
  - Popup Only Append paragraph style identifies the content to display only in a popup window. This style is applied to additional popup paragraphs when you have more than one paragraph of content to include in a popup window.
- Related topics paragraph style that identifies a link to a related topic, such as a concept topic related to a task or a task related to a concept.
- See Also paragraph style that identifies the text you want to include in an inline See Also link.

For more information about enabling a specific online feature, see “Designing, Deploying, and Managing Stationery”.

Character Styles in Word

Create character styles for items based on function, not based on formatting or appearance. This approach allows you to modify formatting over time and the style names continue to apply. It also prepares you for structured writing in the future.

You can apply only one character style to a set of text. If you apply a second character style to that text, it replaces the initial character style. Therefore, you may need more character styles to address all the possible combinations, such as variables in a paragraph and variables in a code sample.

Common character styles include:

- Book titles in cross references
- Emphasized text
- Command names
- File and folder names
- User interface items
- Optional steps or if clauses used to introduce optional steps
- Links
- New terms
- Step numbers, which allows you to apply formatting to the step number defined with a sequence field code
- Text the user must type
- Variables

ePublisher projects use custom field code markers and styles to define online features. You need to give the list of markers and styles to the writers so they know how to implement each online feature. The writers use the markers and styles you create to define online features.

The Stationery defines the custom markers and styles. To reduce complexity, you can use the style names defined in the documentation, or you can define the online feature to a different style. The following list identifies additional character styles you may need to support ePublisher online content features:

- Multiple language support, such as bidirectional languages and text, can require a paragraph or character style with Bidi support enabled.
- Abbreviation character style identifies abbreviation alternate text for browsers to display for abbreviations, such as SS#, when a user hovers over the abbreviation in output. Screen readers also can read the abbreviation alternate text. This character style is used in combination with the AbbreviationTitle marker type.
- Acronym character style identifies acronym alternate text for browsers to display for acronyms, such as HTML, when a user hovers over the acronym in output. Screen readers can also read the

acronym alternate text. This character style is used in combination with the AcronymTitle marker type.

- Citation character style identifies the source of a quote using a fully-qualified Uniform Resource Identifier (URI) when a user hovers over the quote in output. Screen readers can also read the URI for the quote. This character style is used in combination with the Citation marker type.
- See Also character style identifies the text you want to include in a See Also button. This style controls the appearance of the text on the button.

For more information about enabling a specific online feature, see “Designing, Deploying, and Managing Stationery”.

Bulleted and Numbered Lists in Word

ePublisher uses a table-like structure with two columns to display any paragraph style with a hanging indent, such as bulleted and numbered list items, in generated output. ePublisher uses the numbers, characters, and fonts from the source documents for the bullets or numbers. Since some fonts are not available on all computers, you should use character styles in ePublisher to override the formatting of the bullets or numbers. You can also use an image in ePublisher for bullets.

Bulleted and numbered list items can have inconsistent formatting from one computer to another. The formatting is defined by the Bullets and Numbering style gallery. To correctly populate the style gallery, the source document must have an example of each type of bulleted and numbered item. Leave the correct example of each item in the document until that style type is used. If a correctly formatted item does not exist in the document, copy the correct style from the template and paste it in the document to create the first item with that style.

Note: Be aware of paragraphs that have a hanging indent. The hanging indent can cause incorrect alignment of text on the first line of your generated output. For more information see "Defining the Appearance of Numbered Lists".

Bulleted Lists in Word

For bulleted lists, you may need multiple bullet levels, such as Bullet, Bullet2, and Bullet3. You may also need a format for a bullet within a procedure, such as a ProcedureBullet, and a bullet within a table, such as a CellBullet. Make sure you consider all supporting formats you may need, such as a ListIntro format for the paragraph that introduces the bulleted list, which should be set to stay with the list (Keep with Next).

Do not base two bulleted items with different bullets on the same base style. Otherwise, when you modify the bullet for one bulleted item style, the bullet on the other item is also affected.

Numbered Lists in Word

For numbered lists, you may need multiple levels, such as a ProcedureStep that uses numbers and a ProcedureSubstep that uses lowercase letters. You may also need a numbered list item in tables, such as CellStep. Make sure you consider all supporting formats you may need, such as a ProcedureIntro format.

Microsoft Word can have issues with the built-in autonumbering when restarting the numbering in lists. To avoid these issues, you can use sequence field codes to completely control numbering in your source documents. You can define a paragraph style with a hanging indent for each numbered step item. Define a character style and apply it to the sequence field code to control how the numbered list appears both in print and in your generated output. Autotext can help you automate list creation. You can also create a macro to quickly number a set of paragraphs. To restart a field code, add the `\r1` option.

Image Styles and Considerations in Word

If ePublisher cannot use an original image in the output, or if ePublisher determines it needs to modify the image based on how it is included in the source document, ePublisher rasterizes the image using the options you define for your graphic styles in Style Designer. For example, you can define the dots per inch (DPI) and format for the final images. Rasterization of an image can cause the image to be less clear in the output.

To avoid reduced image quality in your output, and to avoid an extended transformation time during the Image stage and pipeline, review the following considerations:

- When ePublisher encounters an image in your Microsoft Word source documents, ePublisher checks for the following conditions:
  - Is the frame a different size than the original image?
  - Is there white space in the frame with the image?
  - Is the image copied into the document, rather than imported by reference?
  - Is the original image a file format other than `.jpg`, `.gif`, `.png`, or `.svg`?
  - Are there additional elements in the frame, such as text boxes, multiple images, or callouts?

  If ePublisher determines that any of these conditions apply, ePublisher rasterizes the entire frame and applies the options you defined in Style Designer.
- To display images at full size in online output and avoid resizing, which can cause the image to be rasterized, set the **By reference graphics use document dimensions** option for your graphic styles to **Disabled**.
- If you want ePublisher to rasterize all images according to your Style Designer options, set the **By reference graphics** option to **Disabled** for all graphic styles.
- When ePublisher finds an image included by-reference that is the original size and contains no callouts, ePublisher copies the image directly into the output folder in most cases, bypassing the graphic style options.
- To improve the image quality in your output, resize your images as needed using an image editing application before importing them, rather than adjusting the size in Microsoft Word. Otherwise, an image included by reference retains its original file size, and it is either scaled by the browser or rasterized according to the size in the source document, which can result in a distorted image.
- For the best compatibility with most computer monitors, save and import your images at 96 DPI using a format that ePublisher does not rasterize.
- Image callouts are useful in many publications. However, text boxes and line drawings cause images to be rasterized, which can make images less clear in your output. Add and edit callouts in your image editing application and then import the single, final image to avoid the rasterization process.

- You can add text boxes with GraphicStyle markers to your images without causing the image to be rasterized, since markers do not affect the appearance of the image.
- Store image files and source documents on the local computer when generating output.

To achieve the best results when inserting images in Microsoft Word

1. Create a unique paragraph style for images. Use the paragraph alignment properties to control the position of your images.
2. Import your image file by reference onto the unique paragraph rather than copying it into the document. ePublisher supports only `.jpg`, `.gif`, `.png`, and `.svg` files. ePublisher rasterizes all other formats.
3. Do not resize the image in Microsoft Word.

Table Styles in Word

Table styles, which are available in recent versions of Microsoft Word, allow you to define standard tables and quickly create tables with those standards in your source documents. If your version of Microsoft Word does not support table styles, use the TableStyle marker to specify the style to apply in ePublisher that defines the appearance of the table.

Table styles are often overlooked in Microsoft Word. The default template provides many default table styles, such as Table Grid and Table Normal. You can create custom table styles for your specific requirements. Define table header rows for each table that repeat when the table splits across pages, and do not allow rows to break across pages, which can create awkward breaks within tables in your printed content. You can use autotext to quickly create standard tables in your source documents.

When you define your table styles, be sure to consider the various types of tables you may need, such as with lines, without lines, checklists, and action/result tables. You can use a table without lines to layout content within an area on a page, such as a definition list with short terms. You can also create a table style for each indent position needed. For example, you can create a table style to use for tables within a bulleted list that is indented to align with the text of each bulleted list item.

ePublisher allows you to define how the header, footer, and main rows of a table appear in your generated output. To support these formatting properties, your tables must have each of these parts defined in your source documents. If a table does not have a header defined, ePublisher cannot apply the formatting defined for the header row. Microsoft Word does not support table footers, so the footer formatting settings in ePublisher do not apply to Microsoft Word source documents.

ePublisher applies the paragraph and character styles you define for content within each cell. You can also configure ePublisher to ignore character styles in a table. You may need additional paragraph styles to use in tables, such as CellBody and CellBullet, so you can define the proper margins and appearance for your generated output.

Field Codes

Microsoft Word uses field codes to implement standard features, such as index entries and variables. You can use sequence field codes for numbering, such as numbering chapters, steps, figure captions, and table captions. ePublisher recognizes many of the standard field codes and uses them to implement these standard features in your generated output.

ePublisher projects also use custom field codes (markers) and styles to define online features. The toolbar provided by ePublisher in Microsoft Word allows you to quickly insert the custom markers. You need to give the list of markers and styles to the writers so they know how to implement each online feature. The writers use the markers and styles you create to define online features.

The Stationery defines the custom markers and styles. Markers with reserved names have their functions defined by default. You can use these default names, or you can create your own markers. To reduce complexity, use the default marker names, which are also used throughout the documentation. You can also use the style names defined in the documentation to reduce complexity. The following table lists the default custom marker types used to implement online features.

Marker Type	Description
AbbreviationTitle	Specifies abbreviation alternate text for browsers to display for abbreviations such as SS# when a user hovers over the abbreviation in output. Screen readers also can read the abbreviation alternate text. Used in combination with the Abbreviation character format.
AcronymTitle	Specifies acronym alternate text for browsers to display for acronyms such as HTML when a user hovers over the acronym in output. Screen readers can also read the acronym alternate text. Used in combination with the Acronym character format.
Citation	Specifies the source of a quote using a fully qualified Uniform Resource Identifier (URI) when a user hovers over the quote in output. Screen readers can also read the URI for the quote. Used in combination with the Citation character format.
Context Plugin	Specifies context plug-ins for Eclipse help systems. Other Eclipse plug-ins can use the context plug-in IDs to call the Eclipse help system. For more information, see “Using Markers to Specify Context Plug-ins in Eclipse Help”.
DropDownEnd	Marks the end of an expand/collapse section. Used in conjunction with an Expand/Collapse paragraph format.
Filename	Specifies the name of an output file for a page or an image.
GraphicScale	Specifies a percentage to use to resize an image, such as 50 or 75 percent, in generated output.
GraphicStyle	Specifies the name of a graphic style defined in a project to apply to an image. This marker type is an internal marker type that is not displayed in

Marker Type	Description
	Stationery Designer. You cannot create a marker type with a different name and assign it this functionality.
Hypertext	Specifies a link using the `newlink` and `gotolink` commands in Adobe FrameMaker. This marker type is a default Adobe FrameMaker marker type ePublisher automatically maps.
ImageAltText	Specifies alternate text for an image.This text is added to the `alt` attribute of the `img` tag in the output. Screen readers use this text when you create accessible content.
ImageAreaAltText	Specifies alternate text for clickable regions in an image map. This text is added to the `alt` attribute of the `img` tag in the output. Screen readers use this text when you create accessible content.
ImageLongDescByRef	Specifies the path to the file that contains the long description for an image. This text is added to the `longdesc` attribute of the `img` tag in the output. Screen readers read this description when you create accessible content.
ImageLongDescNotReq	Specifies that a long description is not required for an image, which bypasses this accessibility check for the image when you create accessible content.
ImageLongDescText	Specifies the long description for an image. This text is added to the `longdesc` attribute of the `img` tag in the output. Screen readers read this description when you create accessible content.
Keywords	Specifies the keywords to include in the `meta` tag for the topic. The `meta` tag improves searchability on the Web.
PageStyle	Specifies the name of a page style defined in the project to apply to a topic. This marker type is an internal marker type that is not displayed in Stationery Designer. You cannot create a marker type with a different name and assign it this functionality.
PassThrough	Specifies that ePublisher place the contents of the marker directly into the generated output without processing the content in any way. For example, you could use a PassThrough marker if you wanted to embed HTML code within your generated output.
Popup	Specifies the start of the content to include in a popup window. The content is displayed in a popup window when you hover over the link. When you click the link in some output formats, the topic where the popup text is stored, such as the glossary, is displayed.
PopupEnd	Marks the end of the content to include in a popup window.
PopupOnly	Specifies the start of the content to include in only a popup window. Browsers display the content in a popup window when you hover over or click the link.
RubiComposite	No longer supported.

Marker Type	Description
SeeAlsoKeyword	Specifies an internal identifier for a topic. SeeAlsoLink markers in other topics can list this identifier to create a link to this topic. Used in conjunction with a See Also paragraph format or character format.
SeeAlsoLink	Identifies an internal identifier from another topic to include in the list of See Also links in this topic. Used in conjunction with a See Also paragraph format or character format.
SeeAlsoLinkDisplayType	Specifies whether to display the target topics on a popup menu or in a window. By default, the links are displayed in the Topics Found window. To display a popup menu, set the value to `menu`. This marker type is supported only in HTML Help.
SeeAlsoLinkWindowType	Specifies the name of the window defined in the `.hhp` file, such as TriPane or Main, that the topic opens in when the user clicks the link. This marker type is supported only in HTML Help.
TableStyle	Specifies the name of a table style defined in the project to apply to a table in versions of Microsoft Word that did not support table styles. This marker type is an internal marker type that is not displayed in Stationery Designer. This marker type is supported only for Microsoft Word documents. You cannot create a marker type with a different name and assign it this functionality.
TableSummary	Specifies an alternate text summary for a table, which is used when you create accessible content. This text is added to the `summary` attribute of the `table` tag in the output. Screen readers read this description when you create accessible content.
TableSummaryNotReq	Specifies that a summary is not required for a table, which bypasses this accessibility check for that table.
TOCIconHTMLHelp	Identifies the image to use as the table of contents icon for a topic in the HTML Help output format.
TOCIconJavaHelp	Identifies the image to use as the table of contents icon for a topic in the Sun JavaHelp output format.
TOCIconOracleHelp	Identifies the image to use as the table of contents icon for a topic in the Oracle Help output format.
TOCIconWWHelp	Identifies the image to use as the table of contents icon for a topic in the WebWorks Help output format.
TopicAlias	Specifies an internal identifier for a topic that can be used to create a context-sensitive link to that topic.
TopicDescription	Specifies a topic description for a context-sensitive help topic in Eclipse help systems. For more information, see “Using Markers to Specify Topic Descriptions for Context-Sensitive Help Topics in Eclipse Help”.

Marker Type	Description
WhatIsThisID	Identifies a What Is This help internal identifier for creating context-sensitive What Is This field-level help for Microsoft HTML Help.
WindowType	Specifies the name of the window defined in the help project that the topic should be displayed in. In Microsoft HTML Help, the window names are defined in the `.hhp` file. This marker type is supported in Microsoft HTML Help and Oracle Help.

AutoText, AutoCorrect, and User-Defined Hotkeys

Autotext improves consistency and efficiency for your writing team. Writers can quickly create items, such as tables and lists, with the correct styles applied by default. You can create autotext for many common items:

- 2-, 3-, 4-, and 5-column tables
- Checklists
- Notes, tips, cautions, and warnings
- Cross reference introductory phrases, such as `For more information, see`
- New chapters and standard topics, such as tasks or command reference topics

AutoCorrect allows you to automatically fix common typing mistakes, including two capital letters in a row and misspelled words. You can also use autocorrect to define some shorthand character sequences that automatically insert longer common phrases. For example, you can define `acl` to be replaced with `access control list`. When you type `acl` and a space, Microsoft Word changes it to `access control list`.

Microsoft Word also allows you to define hot keys for common tasks, such as applying a style, inserting autotext, or running a macro. With hotkeys, autotext, and autocorrect, you can create content more efficiently.

Toolbars and Menus in Word

Create custom toolbars with the options you use most often. You can copy toolbar buttons from the Transit menu and other toolbars to create one combined toolbar with drop-down menus and selections based on your specific requirements. Helpful commands to include on your toolbars are Default Paragraph Font, Keep with Next, and Show/Hide field codes.

Variables and Conditions in Word

On the Properties tab, you can define custom variables to use throughout your source documents. To insert a variable in your content, create field code that references the custom property you created. You can also use field codes to display the contents of a specific style type, such as the previous Heading 1.

For conditions in your generated output, you can use the field codes (markers) supported by ePublisher. You can also use paragraph and character styles to identify conditional content. With this style approach, you need to create extra styles for each condition you need. Then, you can create multiple templates that show or hide specific styles. By attaching the appropriate template, you can include or exclude the appropriate content in your printed output. You can also include or exclude content from your generated output based on styles.

Page Layouts and Sections in Word

You should define all the sections you need in your template. Each section is separated by a section break and has its own page setup. The table of contents and index often have multiple section breaks to customize the page layout in those sections and correctly generate the lists based on field codes. You need to be careful when working around section breaks. If you delete a section break, the page layout for the section, including the headers and footers, may be changed.

In the headers and footers, use field codes to display the contents of a specific style from the associated section, such as the Title style from the title page to include the book title in the footer. This type of field code is automatically maintained and updated. If you use a variable field code in the headers or footers, you need to manually update those field codes. Variable field codes in headers and footers are not updated automatically with the rest of the document content.

Table of Contents and Index in Word

Since the appearance of online table of contents and indexes often differ from printed versions, you need to be able to deliver customized table of contents and indexes in your online content. Therefore, ePublisher does not need the table of contents and index formatting defined in your source documents. ePublisher allows you to define the table of contents levels and appearance, as well as the appearance of the index in your generated output. ePublisher uses the index field codes throughout your source documents to build the online index. This support allows you to deliver the online content you require.

Automation with Macros in Word

Macros allow you to automate, customize, and extend the default Microsoft Word capabilities, including some common tasks and maintenance operations:

- Deleting hidden table of contents bookmarks that build up over time
- Production book tasks, such as updating fields and paginating multiple files
- Attaching different templates for conditional text implementations

This automation can streamline the content authoring process and save you valuable time and effort. Microsoft Word provides a VBA (Visual Basic for Applications) environment for macros to give you the automation capabilities you need. To enable macros, set the security level to medium. You can record steps to perform a specific task, and then copy and edit the generated code to do exactly what you need.

Implementing Online Features in Word

Implement online features in your output by preparing your Microsoft Word source documents with custom marker types, paragraph styles, and character styles defined by the Stationery designer for your Stationery. These markers and styles define the presentation and behavior or your online content. For example, markers can define the name of the file generated for a topic. Formats can define how content displays online.

Custom Marker Types in Word

ePublisher projects use the custom marker types to implement online features when generating output. Custom markers are created using custom marker types available in the WebWorks Transit menu for Microsoft Word. ePublisher inserts markers based on marker types as field codes in your Microsoft Word source documents.

Before you begin using custom marker types to implement online features, talk to the Stationery designer and verify which online features your Stationery supports. Your Stationery only recognizes the custom marker types defined by the Stationery designer in your Stationery. If you try to implement online features using custom marker types not supported in your Stationery, ePublisher does not recognize these items when generating output.

When the Stationery designer creates the Stationery, the Stationery designer can use the default name for a custom marker type or the Stationery designer can use a different name for the customer marker type. The following table lists the default names of custom marker types used to implement online features. Always verify with the Stationery designer the names of the custom marker types you should use when implementing online features before you use these items in your source documents. If you need to create a custom marker type to implement an online feature, verify with the Stationery designer that your Stationery supports the custom marker type before you create the custom marker type and insert and use the custom marker in a source document. For more information about creating custom marker types, see "Creating Custom Marker Types Using the WebWorks Transit Menu in Word".

Marker Type	Description
AbbreviationTitle marker type	Specifies abbreviation alternate text for browsers to display for abbreviations such as SS# when a user hovers over the abbreviation in output. Screen readers also can read the abbreviation alternate text. Used in combination with the Abbreviation character style. For more information, see "Assigning Alternate Text to Abbreviations in Word"
AcronymTitle marker type	Specifies acronym alternate text for browsers to display for acronyms such as HTML when a user hovers over the acronym in output. Screen readers can also read the acronym alternate text. Used in combination with the Acronym character style. For more information, see "Assigning Alternate Text to Acronyms in Word".
Citation marker type	Specifies the source of a quote using a fully qualified Uniform Resource Identifier (URI) when a user hovers over the quote in output. Screen readers can also read the URI for the quote. Used in combination with the Citation character style. For more information, see "Providing Citations for Quotes in Word".
Context Plugin marker type	Specifies context plug-ins for Eclipse help systems. Other Eclipse plug-ins can use the context plug-in IDs to call the Eclipse help system. For more information, see "Specifying Context Plug-ins in Word".

Marker Type	Description
DropDownEnd marker type	Marks the end of an expand/collapse section. Used in conjunction with an Expand/Collapse paragraph style. For more information, see “Creating Expand/Collapse Sections (Drop-Down Hotspots) in Word”.
Filename marker type	Specifies the name of an output file for a page or an image. For more information, see “Specifying Output File Names in Word”.
GraphicScale marker type	Specifies a percentage to use to resize an image, such as 50 or 75 percent, in generated output. For more information, see “Assigning Image Scales in Word”.
GraphicStyle marker type	Specifies the name of a image style defined in a project to apply to an image. This marker type is an internal marker type that is not displayed in Stationery Designer. You cannot create a marker type with a different name and assign it this functionality. For more information, see “Assigning Image Styles in Word”.
ImageAltText marker type	Specifies alternate text for an image.This text is added to the `alt` attribute of the `img` tag in the output. Screen readers use this text when you create accessible content. For more information, see “Assigning Alternate Text to Images in Word”.
ImageAreaAltText marker type	Specifies alternate text for clickable regions in an image map. This text is added to the `alt` attribute of the `img` tag in the output. Screen readers use this text when you create accessible content. For more information, see “Assigning Alternate Text to Image Maps in Word”.
ImageLongDescByRef marker type	Specifies the path to the file that contains the long description for an image. This text is added to the `longdesc` attribute of the `img` tag in the output. Screen readers read this description when you create accessible content. For more information, see “Assigning Long Descriptions to Images in Word”.
ImageLongDescNotReq marker type	Specifies that a long description is not required for an image, which bypasses this accessibility check for the image when you create accessible content. For more information, see “Excluding Images from Accessibility Report Checks in Word”.
ImageLongDescText marker type	Specifies the long description for an image. This text is added to the `longdesc` attribute of the `img` tag in the output. Screen readers read this description when you create accessible content. For more information, see “Specifying Long Descriptions for Images in Word”.
Keywords marker type	Specifies the keywords to include in the `meta` tag for the topic. The `meta` tag improves searchability on the Web. For more information, see “Creating Meta Tag Keywords in Word”.
PageStyle marker type	Specifies the name of a page style defined in the project to apply to a topic. This marker type is an internal marker type that is not displayed in Stationery Designer. You cannot create a marker type with a different name and assign it this functionality. For more information, see “Assigning Custom Page Styles in Word”.
PassThrough	Specifies that ePublisher place the contents of the marker directly into the generated output without processing the content in any way. For example, you

Marker Type	Description
	could use a PassThrough marker if you wanted to embed HTML code within your generated output.
Popup marker type	Specifies the start of the content to include in a popup window. The content is displayed in a popup window when you hover over the link. When you click the link in some output formats, the topic where the popup text is stored, such as the glossary, is displayed. For more information, see “Using Markers to Create Popup Windows in Word”.
PopupEnd marker type	Marks the end of the content to include in a popup window. For more information, see “Using Markers to Create Popup Windows in Word”.
PopupOnly marker type	Specifies the start of the content to include in only a popup window. Browsers display the content in a popup window when you hover over or click the link. For more information, see “Using Markers to Create Popup Windows in Word”.
RubiComposite marker type	No longer supported.
SeeAlsoKeyword marker type	Specifies an internal identifier for a topic. SeeAlsoLink markers in other topics can list this identifier to create a link to this topic. Used in conjunction with a See Also paragraph style or character style. For more information, see “Creating See Also Links in Word”.
SeeAlsoLink marker type	Identifies an internal identifier from another topic to include in the list of See Also links in this topic. Used in conjunction with a See Also paragraph style or character style. For more information, see “Creating See Also Links in Word”.
SeeAlsoLinkDisplayType marker type	Specifies whether to display the target topics on a popup menu or in a window. By default, the links are displayed in the Topics Found window. To display a popup menu, set the value to `menu`. This marker type is supported only in HTML Help. For more information, see “Creating See Also Links in Word”.
SeeAlsoLinkWindowType marker type	Specifies the name of the window defined in the `.hhp` file, such as TriPane or Main, that the topic opens in when the user clicks the link. This marker type is supported only in HTML Help. For more information, see “Creating See Also Links in Word”.
TableStyle marker type	Specifies the name of a table style to apply to a table. This marker type is an internal marker type that is not displayed in the Style Designer as a marker. You cannot create a marker type with a different name and assign it this functionality. For more information, see “Applying Table Styles in Word”
TableSummary marker type	Specifies an alternate text summary for a table, which is used when you create accessible content. This text is added to the `summary` attribute of the `table` tag in the output. Screen readers read this description when you create accessible content. For more information, see “Assigning Alternate Text (Summaries) to Tables in Word”.
TableSummaryNotReq marker type	Specifies that a summary is not required for a table, which bypasses this accessibility check for that table. For more information, see “Excluding Tables from Accessibility Report Checks in Word”.

Marker Type	Description
TOCIconHTMLHelp marker type	Identifies the image to use as the table of contents icon for a topic in the HTML Help output format. For more information, see “Customizing Table of Contents Icons in Word”.
TOCIconJavaHelp marker type	Identifies the image to use as the table of contents icon for a topic in the Sun JavaHelp output format. For more information, see “Customizing Table of Contents Icons in Word”.
TOCIconOracleHelp marker type	Identifies the image to use as the table of contents icon for a topic in the Oracle Help output format. For more information, see “Customizing Table of Contents Icons in Word”.
TOCIconWWHelp marker type	Identifies the image to use as the table of contents icon for a topic in the WebWorks Help output format. For more information, see “Customizing Table of Contents Icons in Word”.
TopicAlias marker type	Specifies an internal identifier for a topic that can be used to create a context-sensitive link to that topic. For more information, see “Specifying Context-Sensitive Help Links in Word”.
Topic Description marker type	Specifies a topic description for a context-sensitive help topic in Eclipse help systems. For more information, see “Creating Context-Sensitive Help in Word”.
WhatIsThisID marker type	Identifies a What’s This help internal identifier for creating context-sensitive What’s This field-level help for Microsoft HTML Help. For more information, see “Creating What’s This (Field-Level) Help in Word”.
WindowType marker type	Specifies the name of the window defined in the Help project that the topic should be displayed in. In Microsoft HTML Help, the window names are defined in the `.hhp` file. This marker type is supported in Microsoft HTML Help and Oracle Help. For more information, see “Opening Topics in Custom Windows in Word”.

Paragraph and Character Formats in Word

ePublisher projects use the paragraph styles and character styles defined by the Stationery designer to implement online features when generating output. Before you begin using paragraph styles and character styles to implement online features, talk to the Stationery designer and verify which online features your Stationery supports. Your Stationery only recognizes the paragraph styles and character styles defined by the Stationery designer in your Stationery. If you try to implement online features using paragraph styles and character styles not supported in your Stationery, ePublisher does not recognize these items when generating output.

When the Stationery designer creates the Stationery, the Stationery designer specifies the names of paragraph styles and character styles used to implement an online feature. Consult with the Stationery designer to obtain the names of the paragraph styles and character styles defined by the Stationery designer to support each online feature you want to implement.

The following table lists the default names of paragraph styles and character styles used to implement online features. Always verify with the Stationery designer the names of the styles formats and character styles you should use when implementing online features before you use these items in your source documents.

style	Description
AbbreviationTitle character style	Specifies abbreviation alternate text for browsers to display for abbreviations such as SS# when a user hovers over the abbreviation in output. Screen readers also can read the abbreviation alternate text. Used in combination with the AbbreviationTitle marker type. For more information, see “Assigning Alternate Text to Abbreviations in Word”.
AcronymTitle character style	Specifies acronym alternate text for browsers to display for acronyms such as HTML when a user hovers over the acronym in output. Screen readers can also read the acronym alternate text. Used in combination with the AcronymTitle marker type. For more information, see “Assigning Alternate Text to Acronyms in Word”.
Citation character style	Specifies the source of a quote using a fully qualified Uniform Resource Identifier (URI) when a user hovers over the quote in output. Screen readers can also read the URI for the quote. Used in combination with the Citation marker type. For more information, see “Providing Citations for Quotes in Word”.
Expand/Collapse paragraph style	Specifies the content you want to include in an expand/collapse section. Used in conjunction with a DropDownEnd marker type. For more information, see “Creating Expand/Collapse Sections (Drop-Down Hotspots) in Word”.
Popup paragraph style	Specifies the popup content to display in both a popup window and in a standard help topic. Applied to the first paragraph of popup content. For more information, see “Creating Expand/Collapse Sections (Drop-Down Hotspots) in Word”.

style	Description
Popup Append paragraph style	Specifies the popup content to display in a popup window and in a standard help topic. Applied to additional popup paragraphs when you have more than one paragraph of popup content. For more information, see “Creating Expand/Collapse Sections (Drop-Down Hotspots) in Word”.
Popup Only paragraph style	Specifies the popup content to display in only a popup window. Applied to the first paragraph of popup content. For more information, see “Creating Expand/Collapse Sections (Drop-Down Hotspots) in Word”.
Popup Only Append paragraph style	Specifies the popup content to display in only a popup window. Applied to additional popup paragraphs when you have more than one paragraph of popup content. For more information, see “Creating Expand/Collapse Sections (Drop-Down Hotspots) in Word”.
Related Topic paragraph style	Specifies related topics links. For more information, see “Creating Related Topics in Word”.
See Also character style	Specifies the text you want to include in a See Also button. For more information, see “Creating See Also Links in Word”.
See Also paragraph style	Specifies the text you want to include in a See Also inline text link. For more information, see “Creating See Also Links in Word”.

Obtaining and Applying the Latest Microsoft Word Template

An efficient, effective, and consistent ePublisher online content generation process relies upon the use of templates. Templates define paragraph, character, and table styles and standards. Templates may also contain standard variables and cross-reference definitions that you can use when creating and working with source documents used to generate online content. Templates help control the look and feel of source documents and generated output across multiple writers, multiple projects, and multiple types of generated output.

The ePublisher content generation process assumes that you use marker types and paragraph, character, and table styles defined in a Microsoft Word template prepared by a Stationery designer as you create content and format your source documents. Using Microsoft Word templates and the marker types and paragraph, character, and table styles and other layout styles and characteristics defined in templates ensures that you format content in your source documents consistently and also ensures ePublisher can use your source documents effectively to generate output.

If your source documents do not use templates or do not use the same marker types, styles and standards defined in your Stationery by the Stationery designer, your generated output may not conforms to the styles and standards defined by the Stationery designer for output. You may also not be able to implement some online features if you do not use the correct templates or the correct marker types and styles defined in the templates.

As a part of preparing your Microsoft Word source documents for output generation, ensure your source documents use the correct Microsoft Word templates from the Stationery designer and you have applied all paragraph, character, and table styles specified in the template correctly.

Working with the WebWorks Transit Menu for Word

This section explains how to work with the WebWorks Transit menu for Microsoft Word. For more information about using the WebWorks Transit menu to prepare Microsoft Word documents for output generation, see “Implementing Online Features in Word”.

WebWorks Transit Menu for Word

The WebWorks Transit menu for Microsoft Word allows you to insert the following items into your Microsoft Word source documents:

- Markers, including filename markers and TopicAlias markers. For more information about markers in Microsoft Word, see “Custom Marker Types in Word”.
- Conditions. For more information about conditions in Microsoft Word, see “Using Conditions in Word”.

Note: Writers authoring content in non-Word formats do not use the WebWorks Transit menu for Microsoft Word.

Installing the WebWorks Transit Menu for Word

If you have Microsoft Word installed on you computer, running the ePublisher Express installer will automatically install the WebWorks Transit plug-in.

If you did not have installed Microsoft Word before installing ePublisher Express, you can do a repair installation after installing Word.

Make sure every time you run the ePublisher Express installer, to close and save your Word documents.

Running Transit Menu in Secure Environments

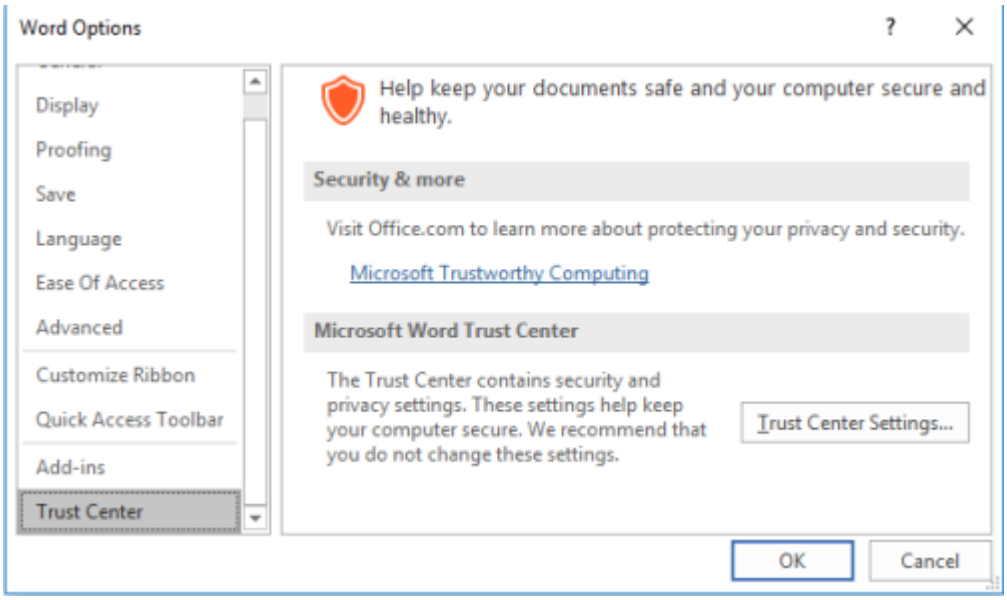
Thankfully, Microsoft eventually got serious about security. Office 2003 added a macro security level feature. By default, only macros signed with a trusted certificate could run. And guess what!!! Now Quadralay Corporation has a trusted certificate for macro documents.

Modern versions of Microsoft Office are even more restrictive. Office 2013 is set to disable all macros by default, providing a notification that the macro wasn't allowed to run. That's why after you install ePublisher Express you still need to say Microsoft Word that you do trust in us.

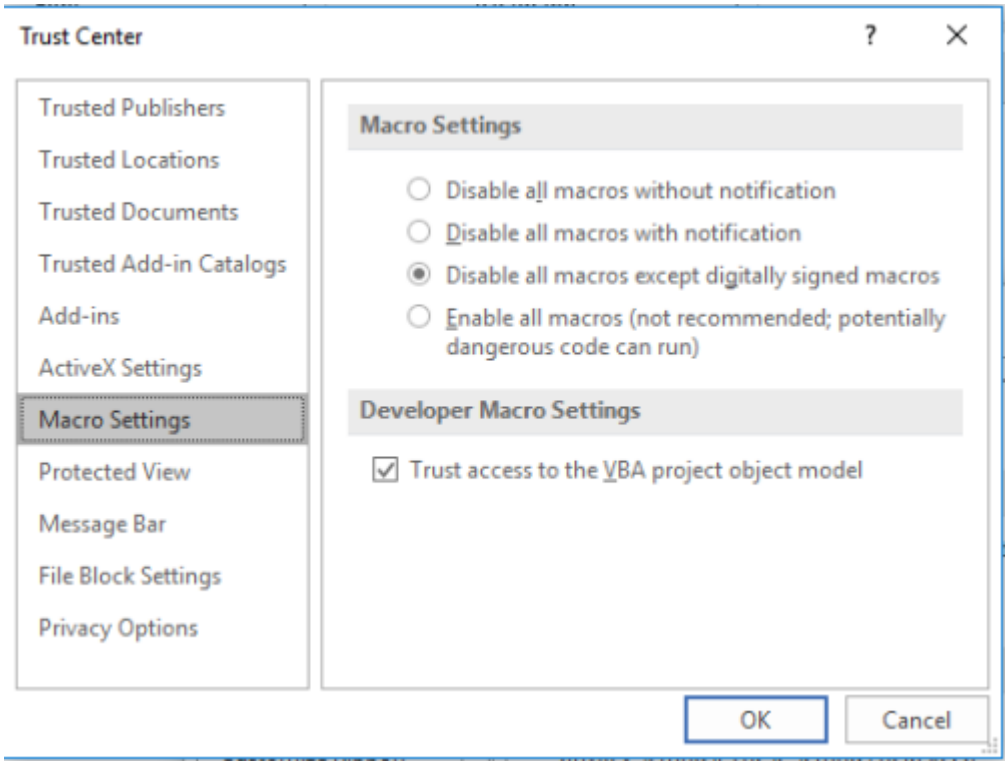
If you need to know a little bit more on how macros work and what are them, you can go to https://www.howtogeek.com/171993/macros-explained-why-microsoft-office-files-can-be-dangerous/.

To run the WebWorks Transit menu for Microsoft Word in a Secure Environment

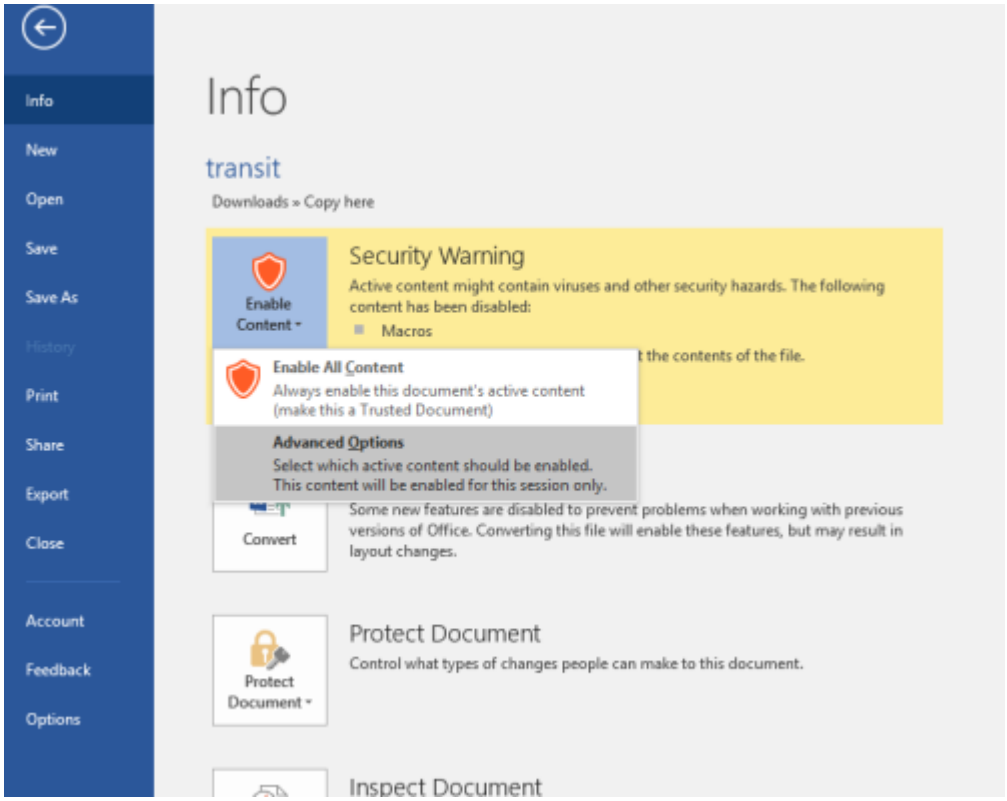
1. Open a Microsoft Word document, or even a Blank document.
2. On the **Word** menu, click **File** and then **Options**.
3. In the window that pops out in the left side menu click on **Trust Center**, and on the right side select **Trust Center Settings**. You should see a window similar to the following figure.



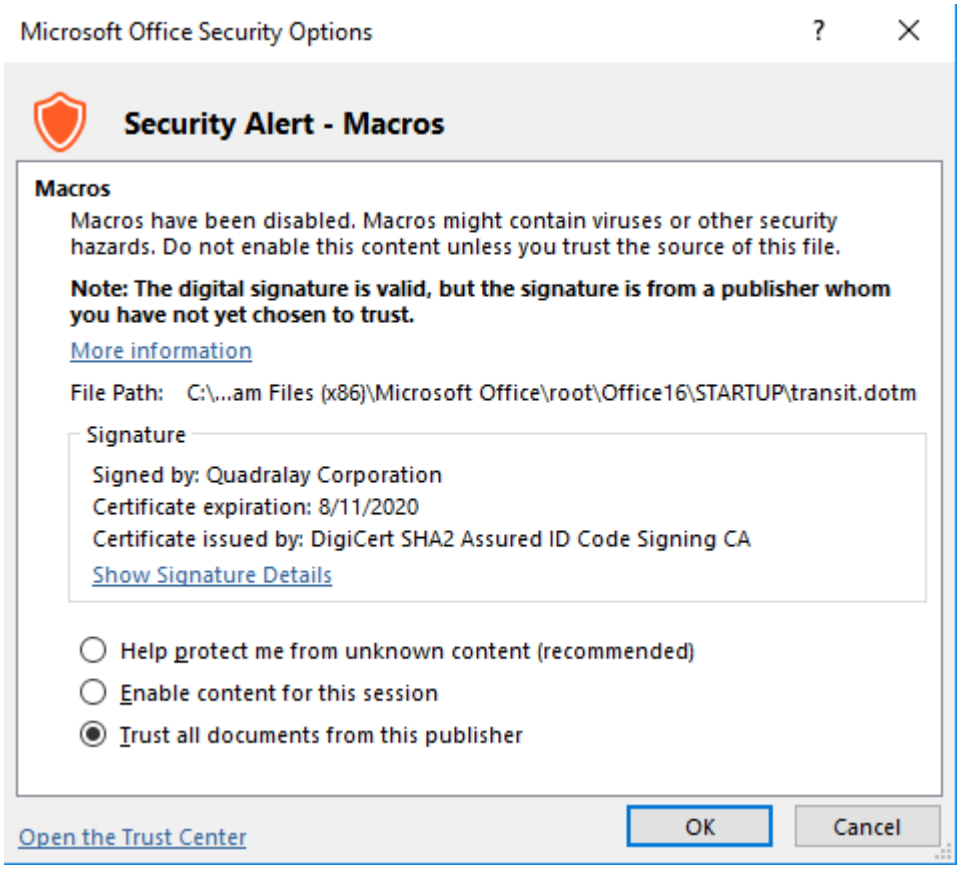
4. After selecting **Trust Center Settings** you'll see a new window, click on the left side on **Macro Settings** and on the right panel check **Disable all macros except digitally signed macros** and **Trust access to the VBA project object model**. You should see a window similar to the following figure.



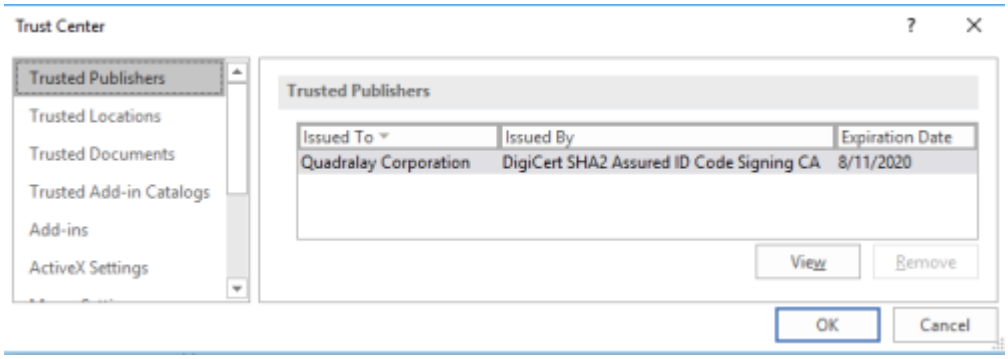
5. Then you can click OK to close all your opened windows and go again to the **Word** menu, click **File** and then **Info**. I you get the Security Warning then click on the **Enable Content** button and then **Advanced Options**. On the right side you'll see a window similar to the next one before clicking **Advanced Options**.



6. On the **Microsoft Office Security Options** check **Trust all documents from this publisher**. As you can see in the following figure: **The digital signature is valid, but the signature is from a publisher whom you have not yet chosen to trust**.



7. Finally you can double check going through Step 2 and 3 above, and then click on the left side **Trusted Publishers** and you'll see *Quadralay Corporation* there, as the next figure shows.

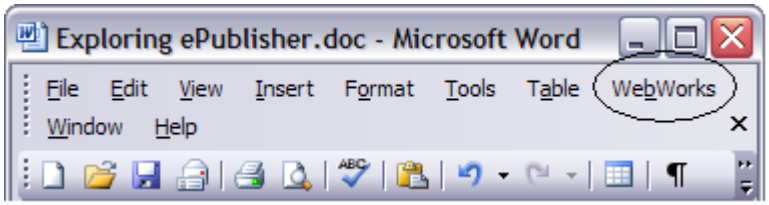


Initializing the WebWorks Transit Menu for Microsoft Word

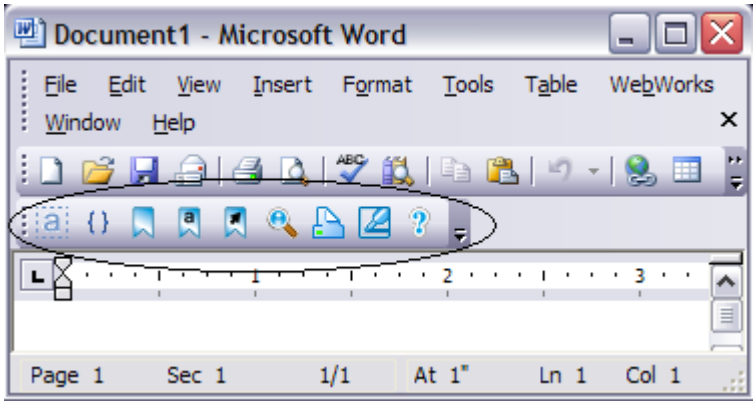
After you install the WebWorks Transit menu for Microsoft Word, you must initialize the WebWorks Transit menu before you can use it. For more information about installing the WebWorks Transit Menu for Microsoft Word, see “Installing the WebWorks Transit Menu for Word”.

To initialize the WebWorks Transit menu for Microsoft Word

1. Open Microsoft Word.
2. Verify that Microsoft Word displays the **WebWorks** menu on the Microsoft Word menu bar. Your Microsoft Word menu should be similar to the following figure.



3. ***If you do not have the WebWorks Transit menu for Microsoft Word installed***, install it. For more information, see “Installing the WebWorks Transit Menu for Word”.
4. ***If this is the first time you are using the WebWorks Transit menu for Microsoft Word***, on the **WebWorks** menu, click **Initialize Menu**. WebWorks initializes the WebWorks Transit menu for Microsoft Word and displays the WebWorks Transit menu in the Microsoft Word window. Your Microsoft Word Window should be similar to the following figure.



Displaying and Hiding the WebWorks Transit Menu in Word

You can choose to display or hide the WebWorks Transit menu in Microsoft Word.

For example, you may want to display the WebWorks Transit menu when you are working in Microsoft Word documents that you will use to generate output. However, you may want to hide the WebWorks Transit menu when you are working in Microsoft Word source documentation that you will not use to generate output.

To display or hide the WebWorks Transit menu for Microsoft Word

1. Open Microsoft Word.
2. On the **WebWorks** menu, click **Preferences**.
3. ***If you want to display the WebWorks Transit menu for Microsoft Word in Microsoft Word***, select the Enable Toolbar check box.
4. ***If you do not want to display the WebWorks Transit menu for Microsoft Word in Microsoft Word***, clear the Enable Toolbar check box.
5. Click **OK**.

Creating Custom Marker Types Using the WebWorks Transit Menu in Word

Typically your Stationery designer will provide the list of markers, which are a type of private or custom Microsoft Word field codes, that you can use in Microsoft Word source documents to create online features. In most cases, you should not need to create a custom marker type for use in Microsoft Word source documents. However, if you need to create a custom marker type to implement an online feature, verify with the Stationery designer that your Stationery supports the custom marker type before you create the custom marker and insert and use the custom marker in a source document.

Occasionally your Stationery may also support a custom marker type that is not defined in the WebWorks Transit menu for Microsoft Word. In this situation, first confirm with the Stationery designer that your Stationery supports the custom marker type. After confirming your project supports the custom marker type, you can create the custom marker type using the WebWorks Transit menu for Word.

To create a custom marker type using the WebWorks Transit menu for Word

1. In your Microsoft Word source document, on the **WebWorks** menu, click **Markers**.
2. Click the plus (+) icon.
3. In the **Type** field, type `Custom`*`MarkerTypeName`* to create a custom marker type, where `Custom`*`MarkerTypeName`* is the name of the custom marker type you want to create.

   Note: The custom marker type name you type must match the name of the custom marker type supported in your ePublisher Stationery. If you specify a name for the custom marker type that is different than the name of the custom marker type supported in your ePublisher Stationery, ePublisher will not be able to recognize and use the custom marker type when generating output.
4. Click **OK**.
5. Click **OK** again to close the window.

The WebWorks Transit menu for Word saves the custom marker you specified in the marker list and inserts the marker in your Microsoft Word source document.

Creating a Passthrough Marker in Word

A passthrough marker is a marker that allows you to insert content that you do not want ePublisher to process when you generate output. For example, if you have embedded multimedia files in your source documents, such as Audio Video Interleave files (`.avi`) or Adobe Software Flash files (`.swf`), you can insert a passthrough marker with a value that is set to the HTML code that you do not want ePublisher to process.

The following example shows `.avi` code to which you could insert using a passthrough marker.

```
<embed src="sample.avi" width="400"
 height="300" pluginspage=";>
</embed>
```

To create a passthrough marker in a Microsoft Word source document

1. In your Microsoft Word source document, locate the paragraph in which you want to insert the passthrough marker.
2. Insert your cursor in the location on the page where you want to insert the **Passthrough** marker.
3. On the **WebWorks** menu, click **Markers**.
4. In the **Marker** field, select **Passhtrough** from the list of markers.
5. In the **Value** field, type the html code that you would like to not be processed by ePublisher such as the Flash embed code indicated in the previous topic.r
6. Click **OK**. ePublisher inserts the Passthrough marker into your source document.
7. Save your Microsoft Word source document.
8. Generate output for your project. For more information, see “Generating Output”.
9. In Output Explorer, verify ePublisher created the appropriate result for your embedded html code. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Working with Tables in Word

This section explains how to prepare tables in source documents for output generation. Obtain your latest templates and apply the latest table formats from the template to tables in your source documents. If your tables do not have header rows, create a header row for each table.

Applying Table Styles in Word

Table styles define the appearance of your tables, and ePublisher uses table styles to define the appearance of tables in generated output. When you work with tables in your Microsoft Word source documents, ensure you apply the correct table styles to your tables. The Stationery designer defines the table styles you can use in your Microsoft Word source documents in the Microsoft Word templates you associate with your Microsoft Word source documents. If you want to specify a different table styles for sets of tables in your generated output, first ensure the different table styles you want to apply are available in your Microsoft Word source document. Then apply the different table styles to tables in your Microsoft Word source documents as appropriate.

For example, you may have a small set of tables that contain information about a specific component in a product. If you decide you want to modify the appearance of these tables in your generated output by specifying that the tables associated with this component display with a yellow background in your generated output, apply a table style available in your Microsoft Word source document that the Stationery designer created to meet this requirement. When you generate output, the Stationery designed by the Stationery designer specifies that any tables created with a table style configured to display tables with a yellow background display in your output with a yellow background.

If you are working in Microsoft Word 2000 or earlier, Microsoft did not provide support for named tabled styles in these versions of Word. You can use TableStyle markers to manually assign table style names to tables. To use the TableStyle marker, insert the TableStyle marker using the WebWorks Transit menu for Word into a cell in the table header row, then save your document, generate output, and verify in Output Explorer that ePublisher applied the table style you specified correctly.

If you are working in Microsoft Word XP (2002) or later, you may use TableStyle markers to assign table styles if you prefer that approach to using Microsoft's available table style assignment controls.

Note: TableStyle markers take precedence over table style names derived from Word assigned table styles.

Because of the ambiguity between the different table styles present in the Microsoft Word ribbon for the versions of 2007 and beyond, a recommended approach would be to use the TableStyle marker in lieu of assigning table styles,

To set a TableStyle marker in a Microsoft Word source document (instructions for Word 2007 or higher)

1. In your Microsoft Word source document, locate the table for which you want to assign a tablestyle marker
2. Insert cursor any where inside the table cells.
3. Click the **Add-Ins** tab to access the **WebWorks Transit** menu
4. Select the **Markers** option from the menu
5. In the **Configure Markers** dialog box, select the **TableStyle** marker from the list
6. In the text box for **Value:**, enter in desired name of the TableStyle

7. Click **OK**

For more information about generating output and using Output Explorer to view output files, see "Generating and Regenerating Output" and "Viewing Output in Output Explorer".

Creating Table Header Rows in Word

Header rows are rows that contain information that help identify the content of a particular column. If the table spans several pages of a print layout, the header row will usually repeat itself at the beginning of each new page.

When you create a table in Microsoft Word, by default Microsoft Word does not create a header row. However, if you create header rows in your Microsoft Word source documents, you can quickly and easily specify the appearance that you want for table header rows in your generated output.

Note: You cannot create table footer rows in Microsoft Word source documents. Microsoft Word does not support the creation of table footer rows.

The following procedure provides an example of how to create table header rows in Microsoft Word source documents using Microsoft Word 2003. Steps for creating table header rows in Microsoft Word may be different in other versions of Microsoft Word.

To create a table header row in a Microsoft Word source document

1. In your Microsoft Word source document, locate the table for which you want to create a table header row.
2. Select the row or rows of an existing table you want to use to create the header row.
3. On the **Table** menu, click **Table Properties**.
4. On the **Row** tab, in the **Options** area, verify that the **Repeat as header row at the top of each page** check box is selected.
5. ***If the check box is not selected***, select the check box to create a header row for the table.
6. Click **OK**.

Working with Images in Word

Many writers include images when producing documents using Microsoft Word. Most writers typically insert images into Microsoft Word source documents in one of the following ways:

- Inserting images in Microsoft Word source documents, also known as embedding images
- Inserting links to image files in the Microsoft Word source documents

If you insert an image into a Microsoft Word source document, Microsoft Word inserts, or embeds, the image in the Microsoft Word source document, and the image becomes a part of the document. Embedded images move with the text of the paragraph in the document. Embedded images in Microsoft Word are also sometimes called **inline shapes**.

If you insert a link to an image is Microsoft Word source documents, Microsoft Word inserts a link to the image and displays the image in the Microsoft Word source document. The link becomes a part of the document, but the actual image file is not inserted into the document, although the actual image files is displayed in the document. If you update the image file referenced by the link, Microsoft Word displays the updated image referenced by the link automatically. Linked images in Microsoft Word are also sometimes called shapes.

There are benefits and drawbacks to inserting images directly into Microsoft Word source documents and inserting links to images used in Microsoft Word source documents.

For example, if you insert images in Microsoft Word source documents, you do not have worry about breaking links between the Microsoft Word source documents and the image files. If you link to images in Microsoft Word source documents, you must ensure that you keep the same file structure for the image files in order to not break links between the Microsoft Word source document and the image file.

However, linking images in Microsoft Word source files, rather than inserting or embedding images, provides some of the following benefits:

- You can update image files without reinserting the image file into your Microsoft Word source documents.
- If you have one image used in multiple places, you can update the image in one place, rather than reinserting the image into multiple places.
- You can manage your documentation files and image files separately, which makes organizing images easier.
- Source documents with linked images are smaller in size than source documents with inserted, or embedded, images.

When you work with Microsoft Word source documents that you will use to generate output, ensure you follow the guidelines specified by the Stationery designer for the following items:

- Method used to insert images
- Correct DPI to use for inserted images

- Correct image file format to use for inserted images

Inserting Images in Word

Before you insert images into Microsoft Word source documents you plan to use to generate output, review image considerations. For more information, see “Working with Images in Word”.

The following procedure provides an example of how to insert an image in Microsoft Word source documents using Microsoft Word 2003. Steps for inserting an image in Microsoft Word may be different in other versions of Microsoft Word.

After you insert your images, validate your images. For more information, see “Validating Images in Word”.

To insert an image in a Microsoft Word source document

1. In your Microsoft Word source document, on the **Insert** menu, click **Picture > From File**.
2. Browse to the location of the image file you want to insert in your Microsoft Word source document, and then select the image file.
3. ***If you want to embed the image in your Microsoft Word source document***, click the **Insert** button. Microsoft Word inserts the image in your source document and displays the image.
4. ***If you want to insert a link to the image file in your Microsoft Word source document***, click the drop-down button on the **Insert** button and then click **Link to File**. Microsoft Word inserts a link to the image in your source document and displays the image in your source document.

After you insert an image, you can assign alternate text or a long description to the image. For more information, see “Assigning Alternate Text to Images and Image Maps in Word” and “Assigning Long Descriptions to Images in Word”.

Validating Images in Word

If you inserted images in your Microsoft Word source documents, you can validate the images you inserted using the WebWorks Transit menu for Microsoft Word before you generate output. For more information about inserting images into Microsoft Word source documents, see “Inserting Images in Word”.

In the WebWorks Transit menu for Microsoft Word, embedded images are referred to as **inline shapes**, and images referenced by links in Microsoft Word source documents are referred to as **shapes**.When you validate embedded images or images referenced by links in Microsoft Word source documents, if ePublisher detects an issue with the image in your source document, ePublisher displays an error and highlights the image with the issue in your Microsoft Word source document.

To validate images in a Microsoft Word source document:

1. Open the Microsoft Word source document that contains the images you want to validate.
2. ***If you want to validate embedded images***, complete the following steps:
    a. On the **WebWorks** menu, click **Tools > Validate Inline Shapes**.
    b. ePublisher displays a message that tells you how many embedded images are in the source document. Click **OK** to continue with the validation.
    c. ***If ePublisher did not detect any issues with the embedded images***, ePublisher displays a status message that tells you all embedded images were validated successfully.
    d. ***If ePublisher detects an issue with an embedded image***, ePublisher displays an error message. Complete one of the following steps:
        - ***If you want to fix the image with the issue***, click **No** to stop the validation scan. Go to the highlighted inline shape with the issue, fix the inline shape by re-adding the inline shape to your Microsoft Word source document, and then run the inline shape validation scan again.
        - ***If you want to continue the scan without fixing the image with the issue***, click **Yes** to continue with the validation scan.
3. ***If you want to validate images referenced by links***, complete the following steps:
    a. On the **WebWorks** menu, click **Tools > Validate Shapes**.
    b. ePublisher displays a message that tells you how many images referenced by links are in the source document. Click **OK** to continue with the validation.
    c. If ePublisher did not detect any issues with the images referenced by links, ePublisher displays a status message that tells you all images were validated successfully.
    d. If ePublisher detects an issue with an image referenced by a link, ePublisher displays an error message. Complete one of the following steps:

- ***If you want to fix the image with the issue***, click **No** to stop the validation scan. Go to the highlighted shape with the issue, fix the shape by re-adding the link to the shape in your Microsoft Word source document, and then run the inline shape validation scan again.
- ***If you want to continue the scan without fixing the image with the issue***, click **Yes** to continue with the validation scan.

Creating Image Links in Word

You can create image links that allow users who click the image to link to content in another location. For example, if you include your company logo in a source document, you can define a link for the logo so that when users click the logo, they link to your company home page.

The following procedure provides an example of how to create an image link in Microsoft Word source documents using Microsoft Word 2003. Steps for creating an image link in Microsoft Word may be different in other versions of Microsoft Word.

To create an image link in a Microsoft Word source document

1. In your Microsoft Word source document, insert the image for which you want to create an image link. For more information, see “Inserting Images in Word”.
2. Select the image for which you want to create an image link.
3. On the **Insert** menu, click **Hyperlink**.

   Note: If Microsoft Word does not display **Hyperlink** on the **Insert** menu, you cannot use this procedure to create a hyperlinked image. However, you can create a hyperlinked text box to create a hyperlinked image. For more information, see “Creating Clickable Regions for Image Maps in Word”.
4. In the Insert Hyperlink window, select the object you want to link to and specify the appropriate options. For example, you can link to an existing file or web page, a location in a document, or an email address.
5. Click **OK**.
6. Save your Microsoft Word source document.
7. Generate output for your project. For more information, see “Generating Output”.
8. In Output Explorer, verify ePublisher created the image link using the link information you specified on the page by clicking on the image. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Creating Clickable Regions for Image Maps in Word

An image map can be a single image separated with clickable regions or a composite image made up of multiple images grouped together, yet still separated with clickable regions. For example, you could create an image of the countries of Europe and then define an image map for the image that allows users to link to a topic about each country when they click on an area of the image. User can click France to see information about France, Italy to see information about Italy, and so on.

When you define an image map, you can also define alternate text for each clickable region. For example, you might define alternate text for the Italy region as “Click here for more information about Italy.” For more information about assigning alternate text to image maps, see “Assigning Alternate Text to Images and Image Maps in Word”.

Creating Image Maps for Single Images in Word

You create an image map for a single image by inserting the image into a drawing canvas and then creating text boxes with hyperlinks that link to a location with additional content. You can also create image maps for composite images. For more information about creating image maps for composite images, see "Creating Image Maps for Composite Images in Word".

The following procedure provides an example of how to create an image map for a single image in Microsoft Word source documents using Microsoft Word 2003. Steps for creating an image map for a single image in Microsoft Word may be different in other versions of Microsoft Word.

To create an image map for a single image in a Microsoft Word source document:

1. Insert your cursor on the line where you want to insert the single image you want to use for your image map.
2. On the **Insert** menu, click **Text Box**. Microsoft Word inserts a drawing canvas. You will insert your image and the text boxes that contain hyperlinks for each clickable area you want to specify for the image into this drawing canvas.
3. Click in the drawing canvas to insert a text box in the drawing canvas.
4. On the **Insert** menu, click **Picture > From File**.
5. Browse to the location of the image you want to use for your image map, select the image, and then click **Insert**, **Link to File**, or **Insert and Link** based on the image insertion method you use for your projects. Microsoft Word inserts the image into the drawing canvas.
6. Add a text box that covers each region in the image that you want to be able to click by completing the following steps:
    a. Select the drawing canvas.
    b. On the **Insert** menu, click **Text Box**.
    c. Click on the drawing canvas, and then drag and drop the text box over an area of the image you want to make clickable.
    d. Right-click the text box, and then click **Format Text** box.
    e. On the **Colors and Lines** tab, in the **Fill** area, in the **Color** field, select **No Fill** from the list.
    f. In the **Line** area, in the **Color** field, select **No Line** from the list.
    g. Click **OK**.
7. Specify a hyperlink for each text box by completing the following steps:
    a. Right-click the text box, and then click **Hyperlink** from the right-click menu.
    b. Specify the location to which you want to link, and then click **OK**. For example, you can link to a Web site or you can link to a location in the source document.

8. Press and hold the SHIFT key, and then click the image and the text box you created for each hyperlinked area you want to use to create the clickable image map for the area.
9. When you have the image and all of the hyperlinked text boxes you created for the image map selected, continue to press and hold the SHIFT key, then right-click the selection and then click **Grouping > Group** on the context menu.
10. Save your Microsoft Word source document.
11. Generate output for your project. For more information, see “Generating Output”.
12. In Output Explorer, verify ePublisher created the image map using the link information you specified by clicking on the page that contains the image map and then clicking on each area of the image where you created a link. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Creating Image Maps for Composite Images in Word

You can create composite images by inserting the composite images into a drawing canvas and then grouping the composite images with hyperlinked text boxes.

The following procedure provides an example of how to create image maps for composite images in Microsoft Word source documents using Microsoft Word 2003. Steps for creating image maps for composite images in Microsoft Word may be different in other versions of Microsoft Word.

To create an image map for a composite image in a Microsoft Word source document

1. Insert your cursor on the line where you want to insert the composite image you want to use for your image map.
2. On the **Insert** menu, click **Picture > New Drawing**. Microsoft Word inserts a drawing canvas. You will insert the images that make up your composite image and the text boxes that contain hyperlinks for each clickable area you want to specify for the image into this drawing canvas.
3. Select the drawing canvas, and then on the **Insert** menu, click **Picture > From File**.
4. Browse to the location of each image that makes up your composite image, select the image, and then click **Insert**, **Link to File**, or **Insert and Link** based on the image insertion method you use for your projects. Microsoft Word inserts the image into the drawing canvas.
5. Position each image that makes up your composite image in the drawing canvas by dragging and dropping the image into its correct position.
6. Add a text box that covers each region in the image that you want to be able to click by completing the following steps:
    a. Select the drawing canvas.
    b. On the **Insert** menu, click **Text Box**.
    c. Click on the drawing canvas, and then drag and drop the text box over an area of the image you want to make clickable.
    d. Right-click the text box, and then click **Format Text** box.
    e. On the **Colors and Lines** tab, in the **Fill** area, in the **Color** field, select **No Fill** from the list.
    f. In the **Line** area, in the **Color** field, select **No Line** from the list.
    g. Click **OK**.
7. Specify a hyperlink for each text box by completing the following steps:
    a. Right-click the text box, and then click **Hyperlink** from the right-click menu.
    b. Specify the location that you want to link to, and then click **OK**. For example, you can link to a Web site or you can link to a location in the source document.

8. Press and hold the SHIFT key, and then click each image that makes up your composite image and the text box you created for each hyperlinked area you want to use to create the clickable image map for the area.

9. When you have the image and all of the hyperlinked text boxes you created for the image map selected, continue to press and hold the SHIFT key, then right-click the selection and then click **Grouping > Group** on the context menu.

   Note: After grouping the image and the text boxes, do not use the Hyperlink command on the right-click menu to assign a hyperlink to the entire group. If you do, the hyperlink you assign for the group will override the hyperlinks you assigned to the individual text boxes in the group.

10. Save your Microsoft Word source document.

11. Generate output for your project. For more information, see “Generating Output”.

12. In Output Explorer, verify ePublisher created the image map using the link information you specified by clicking on the page that contains the image map and then clicking on each area of the image where you created a link. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Assigning Image Scales in Word

When ePublisher converts images inserted into your source documents, it can scale images to make them display larger or smaller in your generated output. By default, ePublisher uses the scaling factor applied to images as specified by the image style you apply to each image. For example, if you apply an image style to images and the Stationery designer defined the image style to scale images to 80% of their original size, all images that have this image style applied to them will be scaled to 80% in the generated output.

Typically, using the standard scaling factor specified in the image style is sufficient. Occasionally, however you may want to override the scaling factor for an individual image. For example, while most `.gif` images scale to 80%, you may have one large image that you want scaled to 60% in your generated output. You can manually override the standard scaling factor specified in your Stationery for a specific image by using the GraphicScale marker.

To assign a scale to a specific image, your Stationery and template must have the GraphicScale marker type configured. Your output format must also support scaling by image.

The following procedure provides an example of how to specify image scaling for an image Microsoft Word source documents using Microsoft Word 2003. Steps for specifying image scaling for an image in Microsoft Word may be different in other versions of Microsoft Word.

To specify an image scale for an image in a Microsoft Word source document

1. In your Microsoft Word source document, locate the image for which you want to specify image scaling.
2. Right-click the image, and then click **Format Picture** or **Format Object**.
3. Change the layout setting of the image to **Top and Bottom** by completing the following steps:

   Note: By default when you insert images into Microsoft Word, Microsoft Word inserts the image using the **Inline with text layout** setting. In order to specify the image scale for image output files, you must group the image and the text box that contains the GraphicScale marker. However, you cannot group images using the **In line with text** layout setting in Microsoft Word. To work around this known Microsoft Word issue, if you have an image that uses an **In line with text** layout setting, use the **Top and Bottom** layout setting for the image while you insert the GraphicScale marker, and then reapply the **In line with text** layout setting after you group the image and the GraphicScale marker.

   a. On the **Layout** tab, click **Advanced**.

   b. On the **Text Wrapping** tab, click **Top and Bottom**.

   c. Click **OK**, and then click **OK** again to close the window.
4. Select your image.
5. On the **Insert** menu, click **Text Box**, and then click to the right of your image. Microsoft Word inserts a text box.
6. Insert your cursor into the text box, and then complete the following steps:

a. On the **WebWorks** menu, click **Markers**.

b. In the **Markers** field, select **GraphicScale** from the list of markers.

c. In the **Value** field, type a scaling value for the image.

For example, if you want the image in your Microsoft Word source document reduced by 50% when you generate output, type `50`.

Click **OK**. ePublisher inserts the GraphicScale marker into the text box.

d. Select the text box.

e. Right-click the selected text box, and then click **Format Text Box**.

f. On the **Colors and Lines** tab, in the **Fill** area, in the **Color** field, select **No Fill**.

g. In the **Line** area, in the **Color** field, select **No Line**.

h. Click **OK**.

7. Drag and drop the text box onto the image.

8. Select the text box and the image.

9. Right-click the selected text box and image, and then click **Grouping > Group**.

Note: When you select **Group**, the location of the image in your Microsoft Word source document may change in relation to the text in your source document. For example, the image may move up or down in your Microsoft Word source document. This is known Microsoft Word behavior. You may need to scroll up or down in your source document to the new location of the image to find the image.

10. ***If your image previously used the In line with text layout setting for the image***, reassign this style to your image by completing the following steps:

a. Right-click ***only*** the image, and then click **Format Object**.

Note: You must ensure you right-click ***only*** the image, and not on the text box or the grouped text box and image. If you right-click on the text box or the grouped text box and image, Microsoft Word does not display the **Format Object** menu option on the context menu.

b. On the **Layout** tab, click **In line with text**.

c. Click **OK**, and then click **OK** again to close the window.

11. Save your Microsoft Word source document.

12. Generate output for your project. For more information, see “Generating Output”.

13. In Output Explorer, verify ePublisher created the image using the image scale you specified in the GraphicScale marker by clicking on the page that contains the image for which you specified image scaling. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Assigning Image Styles in Word

Typically you do not need to specify an image style for images when you generate output. By default, each image generated by ePublisher is associated with the default image style defined in the Stationery used by your Stationery. However, if you want to change the image style of one image or a small set of images, you can specify the image style you want to use for an image in your source document using the GraphicStyle marker type.

For example, if you want to specify a yellow border around a set of screen shot images that illustrate a particular piece of product functionality, you can specify that each of the screen shots images in the set have a yellow border around them through the use of the GraphicStyle marker type.

To assign a style to a specific image, your Stationery and template must have the GraphicStyle marker type configured. Your output format must also support specifying image styles.

The following procedure provides an example of how to specify image styles for images in Microsoft Word source documents using Microsoft Word 2003. Steps for specifying image styles for images in Microsoft Word may be different in other versions of Microsoft Word.

To specify an image style for an image in a Microsoft Word source document

1. In your Microsoft Word source document, locate the image for which you want to specify an image style.
2. Select the image, and then select the **WebWorks** menu to insert a **GraphicStyle** marker next to the image. To insert the marker follow these steps.
    a. On the **WebWorks** menu, click **Markers**.
    b. In the **Markers** field, select **GraphicStyle** from the list of markers.
    c. In the **Value** field, type the name of the image style the Stationery designer configured for the Stationery used by your ePublisher project.

       For example, if the Stationery designer configured an image style called GreenBorder in your Stationery, type `GreenBorder`.

       Click **OK**. ePublisher inserts the GraphicStyle marker into the text box.
3. Save your Microsoft Word source document.
4. Generate output for your project. For more information, see “Generating Output”.
5. In Output Explorer, verify ePublisher created the image using the image style you specified by clicking on the page that contains the image for which you specified an image style and verifying ePublisher applied the image style you specified in the generated output. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Creating Index Entries in Word

An index lists the terms and topics discussed in a document and the page or pages on which they appear. An online index provides the user with a point-and-click resource for quickly navigating online content.

ePublisher uses the same native index entry features used in source documents to create a printed index to create an online index. If you include index entries in your source documents, ePublisher detects the index entries and uses the index entries to create an online index in your generated output.

Microsoft Word inserts index entries as an XE (Index Entry) field in a field code. To create index entries in a Microsoft Word source document, insert index entries into your Microsoft Word source document. ePublisher then uses the index entries to create an online index when you generate output.

Before you insert index entries, verify with the Stationery designer that your Stationery is configured to support online index generation. By default, ePublisher enables online index generate for output, but this functionality can be disabled in your Stationery by the Stationery designer. Also confirm that your output format supports online index creation.

The following procedure provides an example of how to insert index entries in Microsoft Word source documents using Microsoft Word 2003. Steps for inserting index entries in Microsoft Word may be different in other versions of Microsoft Word.

To insert an index entry in a Microsoft Word source document

1. In your Microsoft Word source document, select the word you want to include in your index.
2. Press `ALT+SHIFT+X`. Microsoft Word displays the selected text in the **Main entry** field on the Mark Index Entry window.
3. Specify the appropriate options for the index entry, and then click **Mark**. For more information about the options for the index entry, see the Microsoft Word Help.

   Microsoft Word inserts each index entry as an XE (Index Entry) field in a field code. Field codes use hidden text format. If you don't see the XE field after you insert your index entry, click the **Paragraph** symbol on the **Standard** toolbar.
4. After you insert your index entries, update all of your inserted index entries by completing the following steps:

   a. On the **Edit** menu, click **Select All**.

   b. Press `F9`. Microsoft Word updates all of the field codes in the Microsoft Word source document, including the XE (Index Entry) field codes.
5. Hide the XE (Index Entries) in your source document by clicking the **Paragraph** symbol on the **Standard** toolbar to hide the index field codes and hidden text.
6. Save your Microsoft Word source document.
7. Generate output for your project. For more information, see "Generating Output".

8. In Output Explorer, verify ePublisher created the index correctly by clicking on the page or tab that displays the index and then clicking on the index entries. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”

Using Variables in Word

A variable serves as a placeholder for information that may change frequently. Using variables in source documents allows you to quickly and easily control the content in your generated output. When you change the value of a variable in an ePublisher project, it changes the value in only your generated output. The variable value does not change in your source document.

Once you insert variables into your source documents, whenever the value of a item defined by a variable needs to change, you can make the change in a single location, rather than searching and replacing for all instances of the item. For example, you can use variables in the following ways:

- If you have publication dates or release dates in your source documents that you need to update periodically, you can set up the date as a variable.
- If you work with products that have names or versions that frequently change, you can set up variables for product names and versions.
- If you need to produce documentation sets for a product with multiple brands, you can use variables to help you produce documentation for each different brand using the same set of source files.

Creating Variables in Word

Microsoft Word implements variables as `DocProperty` field codes. When you work with Microsoft Word source documents, typically you use variables defined in a Microsoft Word template by a Stationery designer. The variables in these templates will also include the built-in document properties such as `Author` or `Subject`. You import these variables into your Microsoft Word source documents when you apply the template to your source documents. After you import the variables, you insert the variables as appropriate.

Typically you should not need to create variables in your Microsoft Word source files if you use a Microsoft Word template created by a Stationery designer. However, in some cases you may need to create a variable in a Microsoft Word source document if you do not have a Microsoft Word template that includes a variable you need for your project.

The following procedure provides an example of how to create variables in Microsoft Word source documents using Microsoft Word 2007 and Word 2010.

> **Note:** Newer versions of Word may or may not use this exact procedure, however this may provide enough information to get working with variables in Word.

To create a variable in Word 2007 or Word 2010

1. Go to **File**, and click **Info**
2. Click the Properties tab in the right-hand side of the window and click the **Advanced Properties** option from the dropdown
3. Click the **Custom** tab
4. Type in the Name of your variable in **Name,** for example `BookName`
5. For the **Value,** type in the information you want the variable to represent, for example *User Guide*
6. Click **Add** to add the variable to the list

Inserting Variables into Word

You can insert a variable into a source document after you apply a Microsoft Word template that contains the variables to your source document. If you want to use a variable that is not defined in a Microsoft Word template, you must create the variable in your source document before you can insert it. For more information about creating a variable, see “Creating Variables in Word”.

The following procedure provides an example of how to insert variables in Microsoft Word source documents using Microsoft Word 2007 or 2010.

Note: Newer versions of Word may or may not use this exact procedure, however this may provide enough information to get working with variables in Word.

To insert a variable (DocProperty field) into a Microsoft Word document

1. Open the Microsoft Word source document in which you want to insert a variable.
2. Place your cursor in the location where you want to insert the variable.
3. Go to the **Insert** ribbon and click on the **Quick Parts** dropdown and then select **Field...**
4. In the **Field names** selection box, select **DocProperty**
5. In the adjacent selection box labeled **Property,** select the appropriate variable name to insert in your document

Changing Variable Values in Word

You can change the value assigned to a variable in a Microsoft Word source document.

Note: After you change a variable value in your source document, update the variable value for the variable from the previous value to the new value by selecting all the content in your Microsoft Word source document and then pressing the F9 key.

The following procedure provides an example of how to change a variables in Microsoft Word source documents using Microsoft Word 2007 or 2010.

Note: Newer versions of Word may or may not use this exact procedure, however this may provide enough information to get working with variables in Word.

To change a variable value in a Microsoft Word source document

1. Open the Microsoft Word source document that contains the variable with a value you want to change.
2. On the **File** menu, click **Info**.
3. Click the Properties tab in the right-hand side of the window and click the **Advanced Properties** option from the dropdown
4. Click the **Custom** tab and from the **Name** list box, select the variable you wish to change
5. In the **Value** field, type a new value for the variable. The value you type is the value that Microsoft Word displays in your Microsoft Word document.
6. Click **OK**
7. On the **Edit** menu, click **Select All**.
8. Press the F9 key. Microsoft Word updates the variable value in each place in your source document where you inserted the variable.
9. Click **Done** again to close the window.

Deleting Variables in Word

Delete a variable in a Microsoft Word source document when you no longer want to use the variable. Before you delete a variable, ensure you search for the variable and delete or replace all references to the variable. If your source document still contains a reference to a variable after you delete it, Microsoft Word displays errors in places where a reference to a deleted variable still exists.

The following procedure provides an example of how to delete variables in Microsoft Word source documents using Microsoft Word 2007 or 2010.

To delete a variable in a Microsoft Word source document

1. Open the Microsoft Word source document that contains the variable you want to delete.
2. Press `Alt+F9` to display all field codes in the source document.
3. Search for and replace all references to the variable you want to delete.
4. On the **File** menu, click **Info**
5. Click **Properties** and select Advanced Properties
6. On the **Custom** tab, in the **Properties** field, select the name of the variable you want to delete in the **Name** column.
7. Click **Delete**.
8. Click **OK**.
9. On the **Edit** menu, click **Select All**.
10. Press the `F9` key. Microsoft Word updates the fields in your Microsoft Word source document. If there are any fields that reference the deleted variable in your source document, Microsoft Word displays an error message in the field.
11. Search for the word Error in your Microsoft Word source document to verify no references to the deleted variable remain in your source document.
12. Save your Microsoft Word source document.

Using Conditions in Word

Conditions allow you to show or hide information in your source documents and in your online output. You apply conditions to the content in your source documents, and then you set the visibility for those conditions either in your source documents or in your ePublisher project.

For example, your source documents might contain some content that should be displayed in only the printed version and other content that should be displayed in only the online version. You can use the same set of source documents for both printed and online versions through the use of conditions. You can create one condition called PrintOnly specifically for printed content, and then you can create another condition called OnlineOnly specifically for online content. After you create the PrintOnly and OnlineOnly conditions, you can apply them to the appropriate content in your source documents.

Use the WebWorks Transit menu plug-in for Microsoft Word to work with conditions in your Microsoft Word source documents. ePublisher installs the WebWorks Transit menu plug-in for Microsoft Word by default when you install ePublisher Express. You also have the option to install the WebWorks Transit menu plug-in for Microsoft Word when you install ePublisher Designer or ePublisher AutoMap. For more information about installing the WebWorks Transit menu plug-in for Microsoft Word, see “Installing ePublisher Components”.

After you apply conditions in your source documents, ePublisher can use the conditions defined in your source document to control the visibility of content when it generates output. You can also change the visibility specified for any condition in your ePublisher project. Changing the visibility specified for any condition in your ePublisher project does not change the visibility specified for the condition in your source documents.

Creating Conditions in Word

The WebWorks Transit menu for Microsoft Word allows you to quickly and easily create conditions you can then use to control content in your source documents. Obtain a list of supported conditions from the Stationery designer, and then create each supported condition in each of your Microsoft Word source documents using the WebWorks Transit menu for Microsoft Word.

The following procedure provides an example of how to create conditions in Microsoft Word source documents using Microsoft Word 2003. Steps for creating conditions in Microsoft Word may be different in other versions of Microsoft Word.

To create a condition in a Microsoft Word source document

1. Open Microsoft Word.
2. Ensure that the WebWorks Transit Menu for Microsoft Word is installed on your computer and initialized. For more information, see “Working with the WebWorks Transit Menu for Word”.
3. In your Microsoft Word source document, on the **WebWorks** menu, click **Conditions**.
4. Click the **Add** icon.
5. In the **Type** field, type a name for the condition.

   For example, if you want to create a condition for content that you want to display in only online content, type `OnlineOnly`. If you want to create a condition for content that you want to display in only printed content, type `PrintOnly`.
6. ***If you want the content the condition is applied to hidden in Microsoft Word***, select the **Hidden** check box.
7. ***If you want to highlight the content the condition is applied to in Microsoft Word***, in the **Highlight** field, select a color from the drop-down list. Highlighting the content the condition is applied to allows you to more easily see the conditionalized content in your Microsoft Word source documents.
8. Click **OK**.
9. Click **OK** again.

Applying Conditions in Word

After you have created conditions in your Microsoft Word source documents, you can apply conditions to content. For more information about creating conditions in Microsoft Word source documents, see "Creating Conditions in Word".

The following procedure provides an example of how to apply conditions in Microsoft Word source documents using Microsoft Word 2003. Steps for applying conditions in Microsoft Word may be different in other versions of Microsoft Word.

To apply a condition to content in a Microsoft Word source document

1. In your Microsoft Word source document, select the content to which you want to apply the condition.
2. On the **WebWorks** menu, click **Conditions**.
3. Select a condition.
4. Click **Apply Condition**.
5. Click **OK**.

Validating Conditions in Word

An unbalanced condition is a condition that does not have either an opening or closing tag. You may accidently create an unbalanced condition if you delete an opening or closing tag.

The following is an example of a balanced condition:

```
{PRIVATE WWMTS PrintOnly} Timing Devices {PRIVATE WWMTE PrintOnly}
```

The following is an example of an unbalanced condition:

```
{PRIVATE WWMTS PrintOnly} Timing Devices
```

If you have any unbalanced conditions in your Microsoft Word source documents, ePublisher cannot apply the condition when it generates output.

If you use conditions in your Microsoft Word source documents, validate your conditions and verify that your conditions are balanced before you generate output. When you validate conditions, if you have unbalanced conditions in your Microsoft Word source document ePublisher displays the following error.

If ePublisher displays the error, you can go either go to the location of the error, fix the unbalanced condition in your source document, and then continue the validation, or you can cancel the validation.

To validate conditions in a Microsoft Word source document

1. In your Microsoft Word source document, on the **WebWorks** menu, click **Tools > Validate Conditions**. ePublisher scans the Microsoft Word source document for unbalanced conditions.
2. ***If the validation scan detects an unbalanced condition***, click **Navigate to error** to go to the unbalanced condition and correct the error.

3. ***If you want the validation scan to continue without correcting the unbalanced condition***, click **Continue scan**.

4. ***If you want to cancel the validation scan***, click **Cancel scan**.

Removing Conditions in Word

If you no longer want to apply a condition to content in a Microsoft Word source document, you can remove the applied condition from the content.

The following procedure provides an example of how to remove conditions from content in Microsoft Word source documents using Microsoft Word 2003. Steps for removing conditions from content in Microsoft Word may be different in other versions of Microsoft Word.

To remove a condition from content in a Microsoft Word source document

1. In your Microsoft Word source document, select the content with the condition you want to remove.
2. On the **WebWorks** menu, click **Conditions**.
3. Click the **Delete** icon.
4. ***If you want to remove the condition from the content but keep the content in your Microsoft Word source document***, click **OK**.
5. ***If you want to remove both the condition from the content and delete the content from your Microsoft Word source document***, select the **Delete applied content** check box, and then click **OK**.
6. Click **OK** again.

Modifying Conditions in Word

You can edit the name of the condition, specify whether you want the content to which you applied the condition hidden or displayed in Microsoft Word, and change the color assigned to a condition.

The following procedure provides an example of how to modify conditions in Microsoft Word source documents using Microsoft Word 2003. Steps for modifying conditions in Microsoft Word may be different in other versions of Microsoft Word.

To modify a condition in a Microsoft Word source document

1. In your Microsoft Word source document, on the **WebWorks** menu, click **Conditions**.
2. Select the condition you want to modify.
3. Click the **Edit** icon.
4. ***If you want to change the name of the condition***, in the **Type** field, type a new name for the condition.
5. ***If you want the content the condition is applied to hidden in Microsoft Word***, select the **Hidden** check box.
6. ***If you want the content the condition is applied to displayed in Microsoft Word***, clear the **Hidden** check box.
7. ***If you want to change the color used to highlight the content to which the condition is applied***, in the **Highlight** field, select a color from the drop-down list. Specifying a color for the condition allows you to more easily see the content the condition is applied to in your Microsoft Word source document.
8. Click **OK**.
9. Click **OK** again.

Highlighting All Conditions in Word

You can use WebWorks Transit menu functionality to highlight conditions you applied in your Microsoft Word source document. Highlighting all of the conditions you applied in your Microsoft Word source document allows you to see where all of the conditional content is in your Microsoft Word source document.

The following procedure provides an example of how to highlight all conditions in Microsoft Word source documents using Microsoft Word 2003. Steps for highlighting all conditions in Microsoft Word may be different in other versions of Microsoft Word.

To highlight all conditions in a Microsoft Word source document

1. In your Microsoft Word source document, on the **WebWorks** menu, click **Preferences**.
2. Select the **Show highlighting** check box.
3. Click **OK**.

Displaying Conditionalized Content with Conflicting Settings in Word

You can apply more than one condition to content in your Microsoft Word source documents. If you apply more than one condition to content in your Microsoft Word source document and the conditions that you applied to the content have different show and hide settings, you can specify how you want conditionalized content with conflicting show and hide settings displayed in your Microsoft Word source documents.

For example, you may have content with three conditions applied to it. Two of the conditions applied may be set to show, or display, in the Microsoft Word source document, while one of the conditions may be set to hide, or not display, in the Microsoft Word source document.

If you have multiple conditions applied to content in your source documents with conflicting show and hide settings, you can choose if you want to display the content with conflicting conditions in a Microsoft Word source document or hide the content.

The following procedure provides an example of how to display conditionalized content with conflicting show and hide settings in Microsoft Word source documents using Microsoft Word 2003. Steps for displaying conditionalized content with confecting show and hide settings in Microsoft Word may be different in other versions of Microsoft Word.

To display conditionalized content with conflicting show and hide settings in a Microsoft Word source document

1. In your Microsoft Word source document, on the **WebWorks** menu, click **Preferences**.
2. ***If you want to show content that has conditions applied with conflicting show and hide settings***, select the **Give priority to show conditions** check box. This check box is selected by default.
3. ***If you want to hide content that has conditions applied with conflicting show and hide settings***, clear the **Give priority to show conditions** check box.
4. Click **OK**.

Using Passthrough Conditions in Word

A passthrough condition is a condition you apply to content that you do not want ePublisher to process when you generate output. For example, if you have embedded multimedia files in your source documents, such as Audio Video Interleave files (`.avi`) or Adobe Software Flash files (`.swf`), you can apply a passthrough condition to the code so that ePublisher does not process the code.

The following example shows `.avi` code to which you can apply a passthrough condition.

```
<embed src="sample.avi" width="400"
 height="300" pluginspage=";>
</embed>
```

The following example shows `.swf` code to which you can apply a passthrough condition.

```
<embed src="sample.swf" width="400"
     height="300" pluginspage="
     http://www.macromedia.com/shockwave/download/index.cgi?
     P1_Prod_Version=ShockwaveFlash";>
</embed>
```

If you have code in your Microsoft Word source documents that you do not want ePublisher to process, create a passthrough condition and then apply the passthrough condition to the code. For more information, see “Creating Conditions in Word” and “Applying Conditions in Word”.

You can also use Passthrough markers and the Passthrough paragraph styles and character styles options to insert content directly into your output without being transformed and coded for your output.

Deleting Conditions in Word

Delete a condition in a Microsoft Word source document when you no longer want to apply the condition to content in the source document.

The following procedure provides an example of how to delete conditions in Microsoft Word source documents using Microsoft Word 2003. Steps for deleting conditions in Microsoft Word may be different in other versions of Microsoft Word.

To delete a condition in a Microsoft Word source document

1. In your Microsoft Word source document, on the **WebWorks** menu, click **Conditions**.
2. Select the condition you want to delete.
3. Click the **Delete condition** icon.
4. ***If you want to delete the condition from the list of available conditions and remove the condition from any content to which it was applied in the source document***, click **OK**. ePublisher removes the condition from the list of conditions and removes the condition from any content in the source document to which you applied the condition.
5. ***If you want to delete the condition from the list of available conditions and also delete any content to which the condition was applied***, select the **Delete applied content** check box, and then click **OK**. ePublisher removes the condition from the list of conditions and deletes any content to which the condition was applied in the source document.

Specifying Output File Names in Word

By default, ePublisher automatically assigns file names to your generated output files for topics (pages) and for embedded images (graphics).

Note: If you insert your images using the **Link to File** or **Insert and Link** option in the Insert Picture window in Microsoft Word, ePublisher preserves the original file names. For more information, see "Working with Images in Word".

You can customize this naming convention using one of the following methods:

- Inserting Filename markers into your source documents
- Specifying the topic (page) and image (graphic) naming patterns for ePublisher to use in the target settings for your output

This section explains how you can specify page and image output file names in your Word source documents using Filename markers. For more information about using target settings to specify output file names using page and image naming patterns, see "Specifying Page, Image, and Table File Naming Patterns".

Specifying Page Output File Names in Word

Specify specific names for page output files when you generate output using Filename markers. Insert Filename markers into your source document for each page you want to specify a file name for when you generate content.

Note: You can also use page naming patterns to specify names for page output files and embedded image output files. For more information, see "Specifying Page, Image, and Table File Naming Patterns".

To specify a file name for a page output file, your Stationery and template must have the Filename marker type configured. Your output format must also support this feature.

The following procedure provides an example of how to specify page output file names in Microsoft Word source documents using Microsoft Word 2003. Steps for specifying page output file names in Microsoft Word may be different in other versions of Microsoft Word.

To specify page output file names in a Microsoft Word source document

1. In your Microsoft Word source document, locate the page for the topic to which you want to assign a specific file name. For more information about creating pages using page breaks, see "Specifying Page Breaks Settings".
2. Insert your cursor at the beginning of the first heading on the page.
3. On the **WebWorks** menu, click **Insert Filename Marker**.
4. In the **Filename** field, complete the following steps:
    a. Type the file name you want to specify for the output page file. Do not include the output file extension when you type the file name text.
    b. Click **OK**. ePublisher inserts the Filename marker into your Microsoft Word source document.
5. Save your Microsoft Word source document.
6. Generate output for your project. For more information, see "Generating Output".
7. In Output Explorer, verify ePublisher created an output file using the file name you specified. For more information, see "Viewing Output in Output Explorer".

Specifying Image Output File Names in Word

Specify specific names for image output files when you generate output if you embed, or insert images directly into the source document instead of inserting and linking or linking images.

Many writers do not need use Filename markers to specify image output file names because many writers prefer to insert images in Microsoft Word source documents as references, or links, using the **Link to File** or **Insert and Link** option in the Insert Picture window in Microsoft Word. When writers use one of these methods to insert images, ePublisher automatically uses the name of the image file referenced by the link as the name of the image output file when generating image output files from Microsoft Word source documents.

However, some writers prefer to insert images directly into Microsoft Word source documents using the **Insert** option in Microsoft Word. If you use the **Insert** option, Microsoft Word inserts the image directly into the Microsoft Word source document. When you use the **Insert** option to insert images directly into Microsoft Word source documents, by default, ePublisher assigns image output file names for the inserted images using an image naming pattern. For more information about image naming patterns, see "Specifying Page, Image, and Table File Naming Patterns".

However, if you use the **Insert** option to insert images directly into Microsoft Word source documents, you can also use Filename markers to specify image output file names. Insert Filename markers into your source document for each image you want to specify a file name for when you generate content. To specify a file name for an image output file, your Stationery must have the Filename marker type configured. Your output format must also support this feature.

When you specify image output file names, you create a text box, insert a Filename marker in the text box, and then group the image and the text box that contains the Filename marker. The following procedure provides an example of how to specify image output file names in Microsoft Word source documents using Microsoft Word 2003. Steps for specifying image output file names in Microsoft Word may be different in other versions of Microsoft Word.

To specify image output file names for inserted images in a Microsoft Word source document

1. In your Microsoft Word source document, locate the inserted image for which you want to assign an output image file name.

   Note: Only perform this procedure if you have inserted the image directory into the file using the **Insert** option when you inserted the image into your Microsoft Word source document. Do not perform this procedure if you have inserted the image using the **Link to File** or **Insert and Link** options when you inserted the image into your Microsoft Word source document.

2. Right-click the image, and then select **Format Picture** or **Format Object**.

3. Change the layout setting of the image to **Top and Bottom** by completing the following steps:

   Note: By default when you insert images into Microsoft Word, Microsoft Word inserts the image using the **Inline with text** layout setting. In order to specify an image output file name for an inserted image, you must group the image and the text box that contains the Filename marker. However, you cannot group images using the **In line with text** layout setting in

Microsoft Word. To work around this known Microsoft Word issue, if you have an image that uses an **In line with** text layout setting, use the **Top and Bottom** layout setting for the image while you insert the Filename marker, and then reapply the **Inline with text** layout setting after you group the image and the Filename marker.

   a. On the **Layout** tab, click **Advanced**.

   b. On the **Text Wrapping** tab, click **Top and Bottom**.

   c. Click **OK**, and then click **OK** again to close the window.

4. Select the image.

5. On the **Insert** menu, click **Text Box**, and then click to the right of your image. Microsoft Word inserts a text box.

6. Insert your cursor into the text box, and then complete the following steps:

   a. On the **WebWorks** menu, click **Insert Filename Marker**.

   b. In the **Filename** field, type the file name you want to specify for the output image file, and then click **OK**. ePublisher inserts a Filename marker into the text box.

   c. Select the text box.

   d. Right-click the selected text box, and then click **Format Text Box**.

   e. On the **Colors and Lines** tab, the **Fill** area, in the **Color** field, select **No Fill**.

   f. In the **Line** area, in the **Color** field, select **No Line**.

   g. Click **OK**.

7. Drag and drop the text box onto the image.

8. Select the text box and the image.

9. Right-click the selected text box and image, and then click **Grouping > Group**.

   Note: When you select **Group**, the location of the image in your Microsoft Word source document may change in relation to the text in your source document. For example, the image may move up or down in your Microsoft Word source document. This is known Microsoft Word behavior. You may need to scroll up or down in your source document to the new location of the image to find the image.

10. If your image previously used the **In line with text** layout setting for the image, reassign this style to the image by completing the following steps:

    a. Right-click only the image, and then click **Format Object**.

       Note: You must ensure you right-click only the image, and not on the text box or the grouped text box and image. If you right-click on the text box or the grouped text box and image, Microsoft Word does not display the **Format Object** menu option on the context menu.

    b. On the **Layout** tab, click **In line with text**.

c. Click **OK**, and then click **OK** again to close the window.

11. Save your Microsoft Word source document.
12. Generate output for your project. For more information, see “Generating Output”.
13. In Output Explorer, verify ePublisher created an image output file using the file name you specified in the Filename marker. For more information, see “Viewing Output in Output Explorer”.

Creating Context-Sensitive Help in Word

This section explains how you can use ePublisher to create links to context-sensitive help content in Microsoft Word source documents

Context-Sensitive Help in Word

Context-sensitive help links provide content based on the context of what the user is doing. In many cases, this help content is based on the window that is open and active. For example, the Help button on a window in a software product can open a specific Help topic that provides important information about the window:

- What the window allows you to do
- Brief concepts needed to understand the window
- Guidance for how to use the window
- Descriptions about each field on the window, valid values, and related fields
- Links to related topics, such as concepts and tasks related to the window

The Help topic can also be embedded in the window itself, such as an HTML pane that displays the content of the Help topic. Providing this content when and where the user needs it, without requiring the user to search through the help, keeps the user productive and focused. This type of help also makes the product more intuitive by providing answers when and where needed.

There are several methods for creating context-sensitive Help. In addition, output formats use different mechanisms to support context-sensitive Help. You can reference a topic in the following ways:

File name

Use a Filename marker to assign a file name to a topic. Each topic can have no more than one Filename marker by default. However, you can create a custom mapping mechanism using file names. Then, you can open the specific topic with that file name. However, if your file naming changes, you need to change the link to the topic. This file naming approach delivers context-sensitive help capabilities in output formats that do not provide a mapping mechanism.

Internal identifier (topic alias)

Use a TopicAlias marker to define an internal identifier for each topic. The benefit of using an internal identifier is that it allows file names to change without impacting the links from the product. The writer inserts this marker in a topic and specifies a unique value for that topic. Then, the mapping mechanism of your output format determines how that internal identifier is supported. Some output formats, such as HTML Help, use a mapping file that defines these topic aliases.

To simplify the coding of your source documents, the Stationery designer can also configure your Stationery to define both the file name and the topic alias for each topic file.

Before you begin to insert Filename markers or TopicAlias markers into your source documents, consult with your Stationery designer. Confirm that your Stationery supports context-sensitive help links, and discuss with your Stationery designer the type of marker you should use to define context-sensitive help link in your source documents.

For more information about configuring Filename and TopicAlias markers for context-sensitive help links, see the following topics:

- “Defining Context-Sensitive Help Links”

- “Defining Filename Markers for Context-Sensitive Help Links”

“Defining Filename Markers for Context-Sensitive Help Links”***If you generate Eclipse Help output,*** you also can choose the topic description you want to display for each context-sensitive link. When you use a TopicAlias marker to create context-sensitive links, Eclipse creates a `contexts.xml` file that lists all of the context IDs for the Eclipse Help system you created using TopicAlias markers. In the `contexts.xml` file, Eclipse also provides a description of the context-sensitive link. By default, the description Eclipse provides for the context-sensitive link is the text of the first paragraph of the topic. However, if you want to specify a different description for the context-sensitive link, you can do this by using the TopicDescription marker. For more information about using the TopicDescription marker, see “Specifying Context-Sensitive Help Links in Word”.

Planning for Context-Sensitive Help in Word

Creating context-sensitive help requires you to collaborate with application developers. Because topic IDs and map numbers must be embedded in both the software application and in your source documents, you and the application developers must agree in advance on the values to use.

Before you create context-sensitive help topics, first confirm with your application developers that the application supports context-sensitive help. Then work with your application developers to decide how to choose the topic ID for each context-sensitive help topic:

You choose the topic IDs

You can choose a set of topic IDs and embed them in your source documents using TopicAlias markers. When you generate output, ePublisher can generate a mapping file using those topic IDs and assign a unique number to each topic ID. You can provide the generated mapping file to your application developers, who can embed the topic IDs in the application code. You can then manually maintain this mapping file, or you can allow ePublisher to generate a new file each time you generate the help. Remember to give the updated help system and mapping file to your application developers each time.

Your developers choose the topic IDs

Your application developers can choose a set of topic IDs and embed them in the application code. Then, you can get a copy of the mapping file from your application developers, specify this mapping file in your project settings, and embed the topic IDs in your source documents using TopicAlias markers. In this case, ePublisher does not generate the mapping file.

Before you begin to implement context-sensitive help, meet with your application developers to select one of these methods for assigning the topic IDs to use for context-sensitive help links. Once you choose a set of topic IDs, embed them in your source documents using TopicAlias markers and do not change them.

Note: Because of the way Microsoft Word uses markers with the Transit menu, in order for ePublsher to best pickup a marker such as a TopicAlias, please place this marker after the heading and not before.

Specifying Context-Sensitive Help Links in Word

You can use TopicAlias markers that contain topic IDs, or Filename markers that specify file names, to create context-sensitive help. If your output format supports the use of mapping files and topic IDs, typically you use TopicAlias markers to create context-sensitive help. If your output format does not support the use of mapping files and topic IDs, typically you use Filename markers to create context-sensitive help.

If you are generating Eclipse Help, you can also choose to specify a topic description for each context-sensitive help link your created using a TopicAlias marker by using a TopicDescription marker in conjunction with the TopicAlias marker. For more information about how TopicAlias markers and TopicDescription markers can work together when generating Eclipse Help, see "Context-Sensitive Help in Word".

To specify a context-sensitive help link, your Stationery and template must have a TopicAlias or Filename marker type configured. If you are generating Eclipse Help and you want to be able to specify topic descriptions for your context-sensitive help links, you Stationery and template must also have a TopicDescription marker type configured. Consult with the Stationery designer to determine which marker type you should use to create context-sensitive help links and topic descriptions in your source documents. Your output format must also support this feature.

The following procedure provides an example of how to create context-sensitive help links and topic descriptions in Microsoft Word source documents using Microsoft Word 2003. Steps for creating context-sensitive help links in Microsoft Word may be different in other versions of Microsoft Word.

To create a context-sensitive help link in a Microsoft Word source document

1. Open the Microsoft Word source document that contains the context-sensitive topic you want to link to when users click a help button or help icon from within an application.
2. Insert your cursor at the end of the heading paragraph (or body paragraph if no heading) to which you want to link.
3. On the **WebWorks** menu, click **Markers**.
4. Select the marker type the Stationery designer configured your Stationery to support from the drop-down list. For example, select **TopicAlias** or **Filename**.
5. In the **Value** field, type the topic ID you want to specify for the topic.
6. Click **OK**.
7. ***If you are generating Eclipse Help and you want to specify topic descriptions for each context-sensitive help link you are creating***, complete the following steps:
    a. Insert your cursor in the topic after the TopicAlias marker you inserted for the Eclipse context-sensitive help topic.
    b. On the **WebWorks** menu, click **Markers**.
    c. Select the **TopicDescription** marker type from the list.

d. ***If the TopicDescription marker type is not on the list***, check with the Stationery designer to obtain the name of the marker type the Stationery designer created to support this functionality, and then use the marker type specified by the Stationery designer. For more information, refer to "Implementing Online Features in Word".

e. In the **Value** field, type the topic description you want to use.

f. Click **OK**.

8. Save your Microsoft Word source document.

9. Generate output for your project. For more information, see "Generating Output".

10. In Output Explorer, complete the following steps:

    a. Verify that ePublisher inserted the topic ID into the map file when it generated output.

    b. ***If you generated Eclipse Help and specified topic descriptions for your context-sensitive help topics***, verify that the `contents.xml` file for your Eclipse Help system contains the topic descriptions you specified for context-sensitive help topics.

    c. Test the generated output using the application and verify that the application links to the appropriate context-sensitive help topic. This testing ensures the context-sensitive help link you created displays correctly within the application.

Creating Popup Windows in Word

A popup window is a window that is smaller than standard windows and typically does not contain some of the standard window features such as tool bars or status bars. Popup windows display when users hover over or click on a link. The popup window closes automatically as soon as the users click somewhere else.

A typical use of popup windows is to display glossary terms. For example, in printed documentation, terms and definitions are typically grouped in a separate glossary document. However, in online content, you can display glossary definitions in popup windows. With glossary popup windows, users can choose whether or not they want to view the definition of a term.

You create popup windows by creating a link between the word or phrase in a topic and the content you want to display in the popup window. After you create the link, you then insert Popup markers or apply Popup paragraph styles to define the content you want to display in the popup window.

If the Stationery designer configured the Stationery to support popup windows using markers, you use the following Popup markers to create popup windows:

Popup

Specifies the start of the content to include in a popup window. The content displays in a popup window when users hover over or click on the link. In some output formats users can also view the content in a standard help topic window in addition to viewing the content in a popup window. For example, if you insert a Popup marker in front of a glossary definition, the glossary definition displays in both a popup window and in a glossary topic that contains the definition.

PopupEnd

Specifies the end of the content to display in the popup window.

PopupOnly

Specifies that the popup content displays only through a popup window. For example, if you insert a PopupOnly marker in front of a glossary definition, the glossary definition displays only in a popup window.

If the Stationery designer configured the Stationery to support popup windows using paragraph styles, you use the following paragraph styles to create popup windows:

Popup and Popup Append paragraph styles

Specifies that content displays both in popup windows and in standard help topics. You apply the Popup paragraph style to the first paragraph of content you want displayed in the popup window. If you have more than one paragraph of content you want to display, you apply the Popup Append style to the additional paragraphs.

For example, if you apply a glossary term and glossary definitions style for a glossary using the Popup and Popup Append styles, the terms and definitions in your output display in both a popup window and in a glossary topic that contains the definitions.

Popup Only and Popup Only Append paragraph styles

Specifies that content displays only in popup windows. You apply the Popup Only paragraph style to the first paragraph of content you want displayed in the popup window. If you have more than one paragraph of content you want to display, you apply the Popup Only Append style to the additional paragraphs.

For example, if you apply a glossary term and glossary definition style for a glossary using the Popup Only and Popup Only Append paragraph styles, the terms and definitions in your output display in only popup windows. The content is not displayed in an additional glossary topic that contains the definitions.

Creating Popup Window Links in Word

Your first step in creating a popup window is to create a link between a word or phrase in a topic and the popup content you want to display when users hover over or click the link. Use native Microsoft Word functionality to create a link between the word or phrase in a topic and the content you want to display in a popup window. You can create a link in Microsoft Word with a bookmark and a cross-reference or hyperlink.

Before you create popup window links, verify that your output format supports this feature.

The following procedure provides an example of how to create a a popup window link in Microsoft Word source documents using Microsoft Word 2003. Steps for creating popup window links in Microsoft Word may be different in other versions of Microsoft Word.

To create a link between a word or phrase and popup content in a Microsoft Word source document

1. In your Microsoft Word source document, locate the text you want to create a link to and display in the popup window.
2. ***If you want to create a link that includes the link target text***, create the link using a bookmark and a cross-reference by completing the following steps:
    a. Select the text to which you want to link.
    b. On the **Insert** menu, click **Bookmark**.
    c. In the **Bookmark name** field, type a name for the bookmark in CamelCase. The bookmark name cannot include spaces.

       For example, if you are creating a bookmark for the definition of WebWorks Help in your source document, type `WebWorksHelpDefinition`.
    d. Click **Add**. Microsoft Word inserts a hidden bookmark.
    e. In your Microsoft Word source document, locate the word or phrase for which you want to create a link.
    f. Using your cursor, select the text you for which you want to create a link.

       For example, if you want to specify WebWorks Help as a link, select **WebWorks Help**.
    g. On the **Insert** menu, click **Reference** > **Cross-reference**.
    h. In the **Reference type** field, select **Bookmark**.
    i. In the **Insert reference to** field, select **Bookmark text**.
    j. Select the **Insert as hyperlink** check box.
    k. In the **For which bookmark** field, click the name of the bookmark for the text you want to display in the popup.

For example, if you created a bookmark named WebWorksHelpDefinition for text that provides a definition for WebWorks Help, click **WebWorksHelpDefinition**.

l. Click **Insert**, and then click **Close**.

3. ***If you want to create a link that does not include the link target text***, create the link using a bookmark and a hyperlink by completing the following steps:

   a. Insert your cursor in front of the text to which you want to link.

   b. On the **Insert** menu, click **Bookmark**.

   c. In the **Bookmark name** field, type a name for the bookmark in CamelCase. The bookmark name cannot include spaces.

      For example, if you are creating a bookmark for the definition of WebWorks Help in your source document, type `WebWorksHelpDefinition`.

   d. Click **Add**. Microsoft Word inserts a hidden bookmark.

   e. In your Microsoft Word source document, locate the word or phrase for which you want to create a link.

   f. Using your cursor, select the text you for which you want to create a link.

      For example, if you want to specify WebWorks Help as a link, select **WebWorks Help**.

   g. On the **Insert** menu, click **Hyperlink**.

   h. In the **Link to** area, click **Place in This Document**.

   i. In the **Select a place in this document** field, under **Bookmarks**, click the name of the bookmark for the text you want to display in the popup.

      For example, if you created a bookmark named WebWorksHelpDefinition for text that provides a definition for WebWorks Help, click **WebWorksHelpDefinition**.

   j. Click **OK**.

4. Verify that the link goes to the appropriate location in the source document by pressing and holding down the `CTRL` key and then clicking the link.

5. Save your Microsoft Word source document.

After you create a link between a word or phrase in a topic and the popup content you want to display in the popup window, define the content you want to display in the popup window using one of the following methods:

- Create popup windows using Popup markers. For more information, see “Using Markers to Create Popup Windows in Word”.
- Create popup windows using Popup paragraph styles. For more information, see “Using Paragraph Styles to Create Popup Windows in Word”.

Using Markers to Create Popup Windows in Word

You can insert Popup markers into your Microsoft Word source documents to create popup windows. To use Popup markers to create popup windows, your Stationery must have the following items configured:

- Popup marker type
- PopupEnd marker type
- PopupOnly marker type

Your output format must also support this feature.

The following procedure provides an example of how to insert Popup markers in Microsoft Word source documents using Microsoft Word 2003. Steps for inserting Popup markers in Microsoft Word may be different in other versions of Microsoft Word.

Note: Popup content is created from whole paragraphs. You cannot include a subset of a paragraph in a popup.

To use popup markers to create popup windows in a Microsoft Word source document

1. In your Microsoft Word source document, create a link between a word or phrase in the topic and the content you want to display in the popup window. For more information, see "Creating Popup Window Links in Word".
2. Insert your cursor in front of the text you want to display in the popup window.
3. On the **WebWorks** menu, click **Markers**.
4. ***If you want the popup content to display in both a popup window and in a standard help topic***, complete the following steps:
    a. Select **Popup** from the list of markers in the **Marker** field.
    b. Leave the **Value** field blank.
    c. Click **OK** to insert the marker.
5. ***If you want the popup content to display only in a popup window*****,** complete the following steps:
    a. Select **PopupOnly** from the list of markers in the **Marker** field.
    b. Leave the **Value** field blank.
    c. Click **OK** to insert the marker.
6. Specify where you want the popup content to end by completing the following steps:
    a. Insert your cursor at the end of the content you want to display in the popup window.
    b. On the **WebWorks** menu, click **Markers**.
    c. Select **PopupEnd** from the list of markers in the **Marker** field.

   d. Leave the **Value** field blank.

   e. Click **OK** to insert the marker.

7. Save your Microsoft Word source document.
8. Generate output for your project. For more information, see “Generating Output”.
9. In Output Explorer, go to the page where you created the popup window and verify that ePublisher created the popup window that the popup window displays the content you specified. For more information, see “Viewing Output in Output Explorer”.

Using Paragraph Styles to Create Popup Windows in Word

You can use Popup paragraph styles in your Microsoft Word source documents to create popup windows. To use Popup paragraph styles to create popup windows, your Stationery and Microsoft Word template must have the following items configured:

- Popup and Popup Append paragraph style behaviors if you want your content to display both in popup windows and in standard help topics.
- Popup Only and Popup Only Append paragraph style behaviors if you want your content to display only in popup windows.

Your output format must also support this feature.

The following procedure provides an example of how to use Popup paragraph styles to create popup windows in Microsoft Word source documents using Microsoft Word 2003. Steps for using Popup paragraph styles to create popup windows in Microsoft Word may be different in other versions of Microsoft Word.

To create popup windows using Popup paragraph styles in a Microsoft Word source document

1. In your Microsoft Word source document, create a link between a word or phrase in the topic and the content you want to display in the popup window and ensure that the link resolves in the document. For more information, see “Creating Popup Window Links in Word”.
2. Save your Microsoft Word source document.
3. In the ePublisher **Style Designer**, configure the destination paragraph styles with the appropriate popup behavior via the **Options** panel.
4. Generate output for your project. For more information, see “Generating Output”.
5. In Output Explorer, go to the page where you created the popup window and verify that ePublisher created the popup window and that the popup window displays the content you specified. For more information, see “Viewing Output in Output Explorer”.

Creating Expand/Collapse Sections (Drop-Down Hotspots) in Word

You can create sections of content that expand and collapse when you click a link or hot spot. This structure allows you to create items, such as tasks with numbered procedures, bulleted lists, or definitions, that are easy to scan. Users can then expand individual items to display additional information.

Hot spots for expand/collapse sections initially display in one of the following states:

- The content is initially collapsed and will expand beneath the hotspot when the user clicks the hotspot. Clicking the hotspot a second time causes the expanded content to return to its original collapsed state.
- The content is initially expanded and will collapse or disappear from beneath the hotspot when the user clicks the hotspot.

You create expand/collapse sections in Microsoft Word source documents by using the following items:

- An Expand/Collapse paragraph style
- A DropDownEnd marker

You use an Expand/Collapse paragraph style to start expand/collapse sections and a DropDownEnd marker to specify where the content in the expand/collapse section ends. The Stationery defines whether the sections should initially be expanded (shown) or collapsed (hidden) and the image used to show the state of the section.

To create expand/collapse sections, your Stationery and template must have the following items configured:

- An Expand/Collapse paragraph style
- A DropDownEnd marker

Your output format must also support this feature.

The following procedure provides an example of how to create expand/collapse sections in Microsoft Word source documents using Microsoft Word 2003. Steps for creating expand/collapse sections in Microsoft Word may be different in other versions of Microsoft Word.

To create an expand/collapse section in a Microsoft Word source document

1. In your Microsoft Word source document, identify a topic that contains text for which you want to create an expand/collapse section.
2. Apply an Expand/Collapse paragraph style to the text you want users to click to expand or collapse content.

For example, in the following sample procedure, you would apply the Expand/Collapse paragraph style to the *To open a project* text.

To open a project

a. On the **File** menu, click **Open**.

b. Browse to the location of the project on your local computer.

c. Select the project, and then click **Open**.

3. Insert your cursor at the end of the content you want to display in the expand/collapse section.

For example, in the following sample procedure, you would insert your cursor after the period in the last sentence of the procedure, *Select the project, and then click Open*.

To open a project

a. On the **File** menu, click **Open**.

b. Browse to the location of the project on your local computer.

c. Select the project, and then click **Open**.

4. On the **WebWorks** menu, click **Markers**.

5. In the **Markers** field, select the **DropDownEnd** marker.

6. Leave the **Value** field blank.

7. Click **OK**. ePublisher inserts a DropDownEnd marker at your insertion point. This marker identifies where the contents of your expand/collapse section will end.

8. Save your Microsoft Word source document.

9. Generate output for your project. For more information, see “Generating Output”.

10. In Output Explorer, go to the page where you created the expand/collapse section and verify that ePublisher created the expand/collapse section and that the expand/collapse section displays the content you specified. For more information, see “Viewing Output in Output Explorer”.

Creating Related Topics in Word

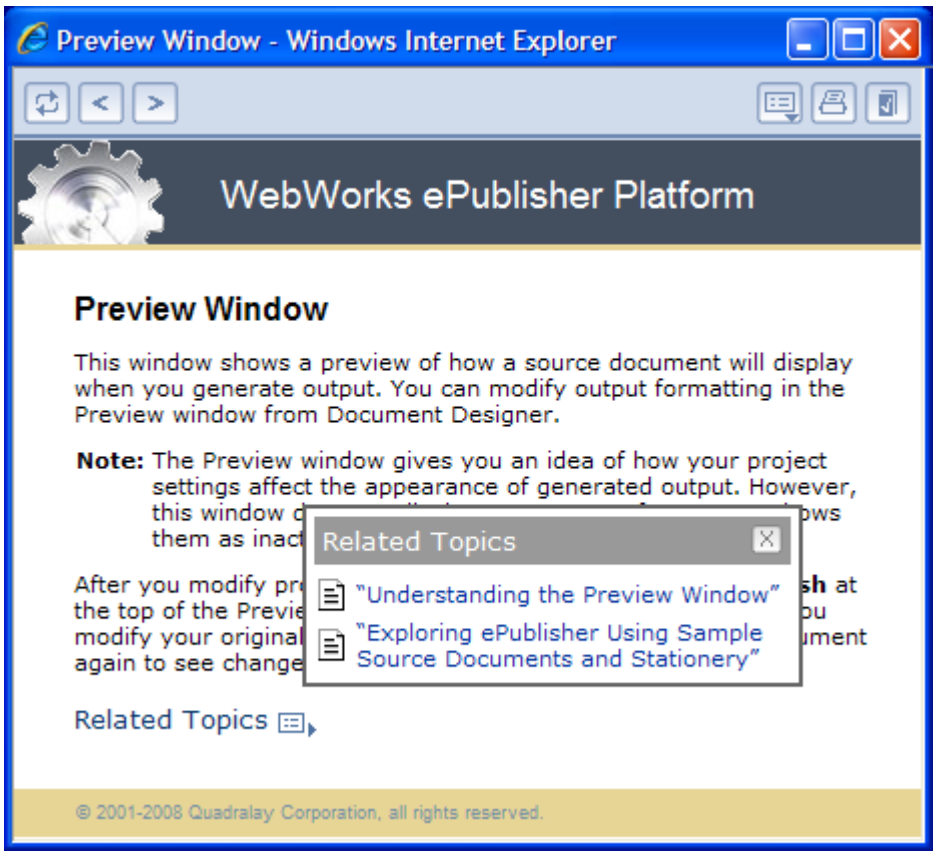
Related topics provide a list of other topics that may be of interest to the user viewing the current topic. For example, you could have a section called Creating Web Pages in your help. You may also have many other topics, such as HTML Tags and Cascading Style Sheets, that related to creating Web pages. Identifying these related topics for users can help them find the information they need and identify additional topics to consider. However, providing these types of links as cross-references within the content itself may not be the most efficient way to present the information. By utilizing related topics links, you combine the capabilities of cross-references with the efficiency of a related topics button.

Related topics and See Also links provide similar capabilities, but there are several important differences:

- Related topics can link to headings in a Help system that do not start a new page.
- Relate topics links are static and defined in the source documents as links. You must have all the source documents to create the link and generate the output.
- If a related topics list contains a broken link in the source document, that link is broken in the generated output. In a See Also link list, the broken link is not included in the output.

The Stationery designer can configure related topics to display in the following ways:

- Included as a list in the topic itself.
- Displayed in a popup window when the user clicks a button, as show in the following figure.



Note: If a related topic link is broken in the source document, in most cases that link is broken in the generated output. WebWorks Help and WebWorks Reverb provide an additional feature by removing broken links from related topics lists that are displayed in a popup window when a user clicks the Related Topics button.

To create related topics links, your Stationery and template must have a Related Topics paragraph style configured. Your output format must also support this feature.

The following procedure provides an example of how to create related topics links in Microsoft Word source documents using Microsoft Word 2003. Steps for creating related topics links in Microsoft Word may be different in other versions of Microsoft Word.

To create a related topics list in a Microsoft Word source document

1. Identify the topic in which you would like to insert a related topics list.
2. Identify the different topics you want to link to from this topic.

   Note: Generally, you should only create one related topics list for each section of your source document that corresponds to a help topic. For example, if the Stationery designer specified in your Stationery that there will be a page break at each Heading 1 section, then you should only create one related topics list for each Heading 1 section within your source document.

3. Create a cross-reference to each topic you want to include in the related topics list by completing the following steps:

    a. Insert your cursor in the location in your Microsoft Word source document where you want to insert the link to the related topic.

    b. On the **Insert** menu, click **Reference > Cross-reference**.

    c. In the **Reference type** field, select **Heading**.

    d. In the **Insert reference to** field, click **Heading text**.

    e. Select the insert as hyperlink checkbooks.

    f. In the **For which heading** field, select the heading to which you want to cross reference.

    g. Click **Insert**.

    h. Click **Close**.

4. Apply the Related Topic paragraph style to the cross-references in your related topics list.

5. ***If you want to display the list of related topics in only your generated output***, apply an OnlineOnly condition to the list of related topics. For more information about applying conditions, refer to “Applying Conditions in Word”.

6. Save your Microsoft Word source document.

7. Generate output for your project. For more information, see “Generating Output”.

8. In Output Explorer, go to the page where you created the related topics list and verify that ePublisher created the related topics and that the related topics list displays the topics you specified. For more information, see “Viewing Output in Output Explorer”.

Creating Links to PDF in Word

You have the ability to link to different information in an external document such as a PDF with a hyperlink to the content.

To create an external hyperlink to a PDF document

1. In the Word menu, go to **Insert** > **Hyperlink**.
2. Select **Existing File or Web Page** for the **Link to** label located on the left.
3. In the **Text to display** text box, enter the text you would like for the hyperlink
4. Navigate to the file location on the system to link against for the PDF.
5. Save the Word document.

Creating See Also Links in Word

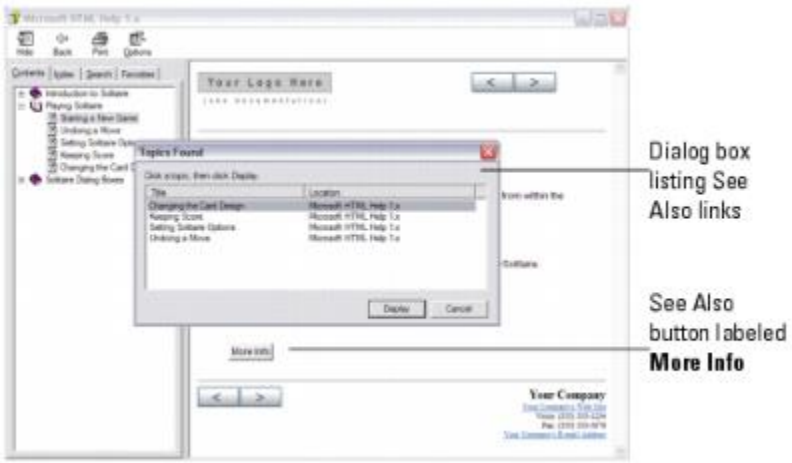
See Also links, also known as ALinks, or associative links, are links that may be of interest to the user viewing the current topic. These links use internal identifiers to specify the links and the link list is built dynamically based on the topics available when the user clicks to display the links. See Also links are important to use with larger help sets and merged help sets.

Related topics and See Also links provide similar capabilities, but there are several important differences:

- See Also links must link to styles that start a new topic, such as a heading.
- See Also links are dynamic and the lists of links are built at display time instead of during help generation.
- Since see Also link lists are dynamically built, they do not include links to topics that are not available when the user displays the links. If a related topics list contains a broken link in the source document, that link is broken in the generated output for most output formats.

See Also links are useful if you plan to merge help systems. For example, if you have a multiple help systems that you merge into one main help system at run time and if your topics in the merged help systems contain See Also keywords that are also used in the main help system, links to those topics are included in the See Also lists in the main project.

You can create See Also links as buttons or as inline text links in Microsoft HTML Help and WebWorks Help. The following example shows how the two different types of See Also links display in a Microsoft HTML Help system.



Create See Also links by applying the See Also paragraph style or character style to text in your Microsoft Word source documents and inserting markers into your Microsoft Word source documents.

Create See Also links by applying the See Also paragraph style or character style to text in your Microsoft Word source documents and inserting markers into your Microsoft Word source documents. To create See Also links, your Stationery and template must have the following items configured:

- See Also paragraph style if you want to create See Also links with buttons
- See Also paragraph style if you want to create see Also links as inline text links
- SeeAlsoKeyword marker type
- SeeAlsoLink marker type
- SeeAlsoLinkDisplay marker type if you generate Microsoft HTML Help and you want to display the target topics in a popup menu

SeeAlsoLinkWindowType marker type if you generate Microsoft HTML Help and you want to display the target topics in a custom window

The following procedure provides an example of how to create See Also links in Microsoft Word source documents using Microsoft Word 2003. Steps for creating See Also links in Microsoft Word may be different in other versions of Microsoft Word.

To create a See Also link in a Microsoft Word source document

1. Identify each topic to which you want to link from a See Also link, and then complete the following steps for each topic:
   a. Insert your cursor into the topic to which you want to link.
   b. On the **WebWorks** menu, click **Markers**.
   c. In the **Marker** field, select the **SeeAlsoKeyword** marker.
   d. In the **Value** field, type a text string that is a unique identifier as the See Also keyword for the topic. See Also keywords are case sensitive and cannot contain punctuation or spaces.

      For example, if you have a unique topic called About WebWorks Help, type `AboutWebWorks` help in the **Value** field.
   e. Click **OK**. ePublisher inserts a SeeAlsoKeyword marker at your insertion point. This marker identifies the topic for See Also links.
2. Identify the topic where you want to insert a list of See Also links.
3. Enter the text you want to display for the See Also button or for the See Also inline text link on a separate line in the source document where you want the See Also button or inline text link to display.

   For example, if you want to create a button with the text See Also on the button, type `See Also`. If you want to create inline text with the text Additional Information for the link, type `Additional Information`.
4. ***If you want to create a See Also button for your See Also links***, apply the See Also paragraph style to the text you want to display in the See Also button.
5. ***If you want to create a See Also inline text link for your See Also links***, apply the See Also character style to the text you want to display in the See Also inline text link.

6. Apply an OnlineOnly condition to the See Also text. Applying an OnlineOnly condition to the See Also button or See Also inline text displays the See Also link in your generated output, but does not display the See Also button or link in your printed content.

7. Insert your cursor inside the text you specified for the See Also button or See Also inline text link.

8. For each topic to which you want to link from a See Also link, complete the following steps:

   a. On the **WebWorks** menu, click **Markers**.

   b. In the **Marker** field, select the **SeeAlsoLink** marker.

   c. In the **Value** field, type the text string that is a unique identifier for the topic to which you want to link. This text string is the text string you typed when you created the **SeeAlsoKeyword** marker for the topic.

      For example, if you created a **SeeAlsoKeyword** marker with the text string AboutWebWorksHelp, type `AboutWebWorksHelp` in the **Value** field for the **SeeAlsoLink** marker.

   d. Click **OK**. ePublisher inserts a **SeeAlsoLink** marker at your insertion point. This marker identifies the topics users can link to when they click the See Also button.

9. ***If you generate Microsoft HTML Help output and you want to display the target topics in a popup menu***, complete the following steps:

   a. Insert your cursor inside the text you specified for the See Also button or inline text link.

   a. On the **WebWorks** menu, click **Markers**.

   b. In the **Marker** field, select the **SeeAlsoLinkDisplayType** marker.

      Note: This marker type is supported only in Microsoft HTML Help.

   c. In the **Value** field, type `menu`. By default, Microsoft HTML Help displays See Also links in the Topics Found window. To display See Also links in a popup menu, specify menu for the marker value.

   d. Click **OK**. ePublisher inserts a SeeAlsoDisplayType marker at your insertion point.

10. ***If you generate Microsoft HTML Help output and you want to display the target topics in a custom window***, complete the following steps:

    a. Insert your cursor inside the text you specified for the See Also button or See Also inline text link.

    b. On the **WebWorks** menu, click **Markers**.

    c. In the **Marker** field, select the **SeeAlsoLinkWindowType** marker.

       Note: This marker type is supported only in Microsoft HTML Help.

    d. In the **Value** field, type the name of a custom window defined for Microsoft HTML Help by the Stationery designer.

For example, if the Stationery designer defined a custom window called ContextHelp to use to when displaying context-sensitive help topics, type `ContextHelp` in the **Value** field for the SeeAlsoLinkWindowType marker.

e. Click **OK**. ePublisher inserts a SeeAlsoDisplayType marker at your insertion point.

11. Save your Microsoft Word source document.

12. Generate output for your project. For more information, see “Generating Output”.

13. In Output Explorer, go to the page where you created the See Also links and verify that ePublisher created the See Also button or See Also inline text and that the See Also button or inline text displays the links you specified. For more information, see “Viewing Output in Output Explorer”.

Creating Meta Tag Keywords in Word

Meta tags are lines of code placed between the `<head>` and `</head>` tags in HTML pages. Meta tags give web search engines information about the content of the web page and how search engines should treat the web page. Users viewing web pages do not see the meta tags, but meta tags can be used to influence the way web pages on a web site appear in web search engine results. Users also see the text you specify for meta tags right following the title of your page when your page comes up in search results.

In help systems, search ranking works like ranking in an Internet search engine. If you generate help system output, you can use meta tag keywords to specify terms for pages for help topics where you want to improve searchability. For example, assume that in your help system you have a topic called See Also links. However, you know that See Also links are also sometimes referred to as ALinks, and you think that some users of your help system may search for information about See Also links by typing `ALinks` into the **Search** field for your help system. In this example, you can insert ALinks as a meta tag keyword for each page that discusses See Also links, so users who search your system for information about ALinks can find the information they are looking for in your See Also link topics.

To assign meta tag keywords, your Stationery and template must have the Keywords marker type configured. Your output format must also support this feature.

The following procedure provides an example of how to create meta tag keywords in Microsoft Word source documents using Microsoft Word 2003. Steps for creating meta tag keywords in Microsoft Word may be different in other versions of Microsoft Word.

To create meta tag keywords for a page in a Microsoft Word source document

1. In your Microsoft Word source document, find the first paragraph in the page for the page for which you want to create a meta tag keyword.
2. On the **WebWorks** menu, click **Markers**.
3. In the **Marker** field, select **Keywords** from the list of markers.
4. In the **Value** field, type the comma-delimited list of keywords that you want web search engines to use when crawling Web sites and to display immediately following the title of your page when your page comes up in search results.

   For example, type `keyword1`, `keyword2`, `keyword3`, where *`keyword`* is the keyword you want web search engines to use when crawling your Web site.
5. Click **OK**.
6. Save your Microsoft Word source document.
7. Generate output for your project. For more information, see “Generating Output”.
8. In Output Explorer, verify that ePublisher inserted your meta tag keywords correctly by completing the following steps:

a. On the **View** menu, click **Output Explorer**.

b. In the *TargetName**ProjectName* folder, open the page to which you assigned meta tag keywords in Notepad, where *TargetName* is the name of your target and *ProjectName* is the name of your project.

c. Verify that the text you specified for your meta tag displays in the `meta name` attribute between in the `<head>` and `</head>` tags section of your web page. For example, if you typed `keyword1`, `keyword2`, `keyword3`, for your meta tag keywords, your meta tags in for the page should be similar to the following entry:

```
<meta name="keywords" content="keyword1, keyword2, keyword3" />
```

Assigning Custom Page Styles in Word

By default, each page generated by ePublisher is associated with the default page style defined in the Stationery used by your ePublisher project. This means that typically you do not need to specify a page style for pages when you generate output. However, if you want to change the page style of one page or a smaller set of pages, you can specify the page style you want to use for a page in your Microsoft Word source document using the PageStyle marker.

For example, you may want to use one page style in your help system for all concept and procedure topic pages, and another page style for all context-sensitive window description topic pages in your help system. In this example, you can use the default page style for all of your concept and procedure topic pages, and then you can use a second custom page style defined in your Stationery for all context-sensitive window description topic pages in your help system.

Before you begin, obtain the names of the custom page styles you can use with your Stationery from the Stationery designer. Then insert a PageStyle marker with the page style name into the topic you want to display using a custom page style. After you assign a custom page style to a topic using the PageStyle marker, the generated output displays the topic using the specified page style.

To assign custom page styles, your Stationery and template must have the following items configured:

- Custom page styles defined for your Stationery by the Stationery designer
- PageStyle marker type

Your output format must also support this feature.

The following procedure provides an example of specifying page styles for pages in Microsoft Word source documents using Microsoft Word 2003. Steps for specifying page styles for pages in Microsoft Word may be different in other versions of Microsoft Word.

To specify a custom page style for a page in a Microsoft Word source document

1. In your Microsoft Word source document, locate the page for the topic to which you want to assign a page style.
2. Insert your cursor in the location on the page where you want to insert the **PageStyle** marker.
3. On the **WebWorks** menu, click **Markers**.
4. In the **Marker** field, select **PageStyle** from the list of markers.
5. In the **Value** field, type the name of the page style you want to associate with the page.

   For example, if you Stationery designer configured a page style for your Stationery called YellowBackground, type `YellowBackground`.
6. Click **OK**. ePublisher inserts the PageStyle marker into your source document.

7. Save your Microsoft Word source document.
8. Generate output for your project. For more information, see “Generating Output”.
9. In Output Explorer, verify ePublisher created the page using the page style you specified by clicking on the page and verifying ePublisher applied the page style you specified in the generated output. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Creating What's This (Field-Level) Help in Word

If you generate Microsoft HTML Help output, you can implement What's This help for product dialog boxes and windows. What's This Help is also known as field-level help. Only Microsoft HTML Help supports field-level help. In addition, not all products are designed to support field-level help for product dialog boxes and windows. Before you begin implementing field-level help, consult your product team to determine if field-level help part of the product design. If field-level help is part of the product design, you will also need to obtain the appropriate ID from your product team for each field-level help topic you need.

Users can view the field-level help you create using one of following methods:

- Users click on the question mark icon in the upper right corner of the dialog box or window.

  When users click on the question mark icon, their cursor changes to a question mark. Users can then move the question mark cursor over the fields on the dialog box or window, and Windows displays the field-level help you created in a popup window when they hover over a specific field.

- Users right-click a field on a dialog box or window and then select the **What's This?** option from the single option menu Windows displays.

  After users select this option, Windows displays the field-level help you specify in in a popup window.

Users close the popup window that provides the field-level help by pressing the `Esc` key on the keyboard. When users press the `Esc` key, their cursor returns to the regular cursor shape for the user.

To create What's This help, your Stationery and template must have the following items configured:

- What Is This help paragraph style

WhatIsThisHelpIDThe following procedure provides an example of how to create What's This help in Microsoft Word source documents using Microsoft Word 2003. Steps for creating What's This help in Microsoft Word may be different in other versions of Microsoft Word.

To create What's This help in a Microsoft Word source documents

1. Identify a topic that contains field-level help.
2. Apply the What Is This paragraph style to the text that contains the field-level help.
3. Insert your cursor into the field-level help text.
4. On the **WebWorks** menu, click **Markers**.
5. In the **Marker** field, select **WhatIsThisID** from the list of markers.
6. In the **Value** field, type the appropriate ID for the field-level description. Obtain appropriate IDs for each field-level description from your product team.
7. Click **OK**.

8. Save your Microsoft Word source document.
9. Generate output for your project. For more information, see “Generating Output”.
10. In Output Explorer, verify ePublisher created the What’s This help you specified by completing the following steps:
    a. On the **View** menu, click **Output Directory**.
    b. Open the `ProjectName` folder, where *`ProjectName`* is the name of your project.
    c. Open the `whatisthis.txt` file and verify that the field-level help you created is associated with the correct ID you received from your product development team.
    d. Open the `whatisthis.h` file and verify that each new string you added is listed in the file.

Opening Topics in Custom Windows in Word

You can open topics in custom windows in Microsoft HTML Help and Oracle Help. By default, Microsoft HTML Help displays content in the standard Microsoft HTML Help tri-pane window. The Stationery designer can modify the size, position, and other characteristics of the tri-pane window in your Microsoft HTML Help project. The Stationery designer can also define custom windows for you to use in a Microsoft HTML Help project. If the Stationery designer defines custom windows in a Microsoft HTML Help project, you can specify which topics you want to display in the custom window using the WindowType marker.

By default, Oracle Help displays content in the standard Oracle Help viewer. The Stationery designer can modify the size, position, and other characteristics of Oracle Help windows. The Stationery designer can also define custom windows for you to use in an Oracle Help project. If the Stationery designer defines custom windows in an Oracle Help project, you can specify which topics you want to display in the custom window using the WindowType marker.

For example, if you want your context-sensitive help topics to display in a different type of window than other content, after you create a context-sensitive help topic you can use the WindowType marker to specify that you want the context-sensitive help topics to display in a custom window. After you assign a custom window to a topic using the WindowType marker, the help system displays the topic in your generated output in the custom window whenever users access the topic from the table of contents, index, a standard hyperlink, a related topics list, or a See Also link.

To open topics in custom windows, your Stationery and template must have the following items configured:

- Custom window styles defined for your Stationery by the Stationery designer
- PageStyle marker type

The following procedure provides an example of how to specify topics open in custom Microsoft HTML Help or Oracle Help windows in Microsoft Word source documents using Microsoft Word 2003. Steps for specifying topics open in custom Microsoft HTML Help or Oracle Help windows in Microsoft Word may be different in other versions of Microsoft Word.

To specify topics open in a custom window in a Microsoft Word source document

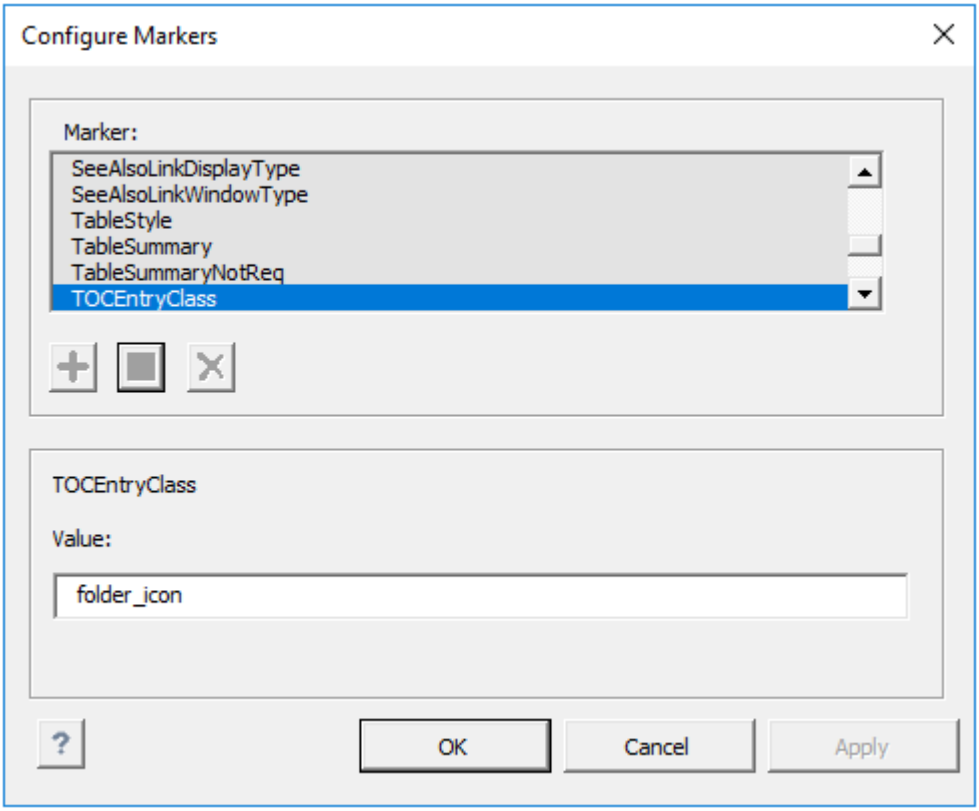
1. Obtain the names of custom windows configured in the Stationery you use for your ePublisher project from the Stationery designer.
2. In your Microsoft Word source document, locate the topic that you want to open in a custom window.
3. Insert your cursor into the topic.
4. On the **WebWorks** menu, click **Markers**.
5. In the **Marker** list, select **WindowType** from the list.

6. In the **Value** field, type the name of the custom window configured by the Stationery designer that you want to specify for the topic.
7. Click **OK**.
8. Save your Microsoft Word source document.
9. Generate output for your project. For more information, see “Generating Output”.
10. In Output Explorer, verify the topic displays in the custom window you specified for the topic. For more information about viewing output files in Output Explorer, see “Viewing Output in Output Explorer”.

Customizing TOC Entry in Word

Use these steps to customize a TOC entry in your **Reverb 2.0** output. Your Word file must have a nested heading structure for TOC Icons to appear.

1. In your Word document, click in or highlight the header that will have the customized TOC entry.
2. Click Markers from the Transit menu in Word.
3. Select TOCEntryClass.



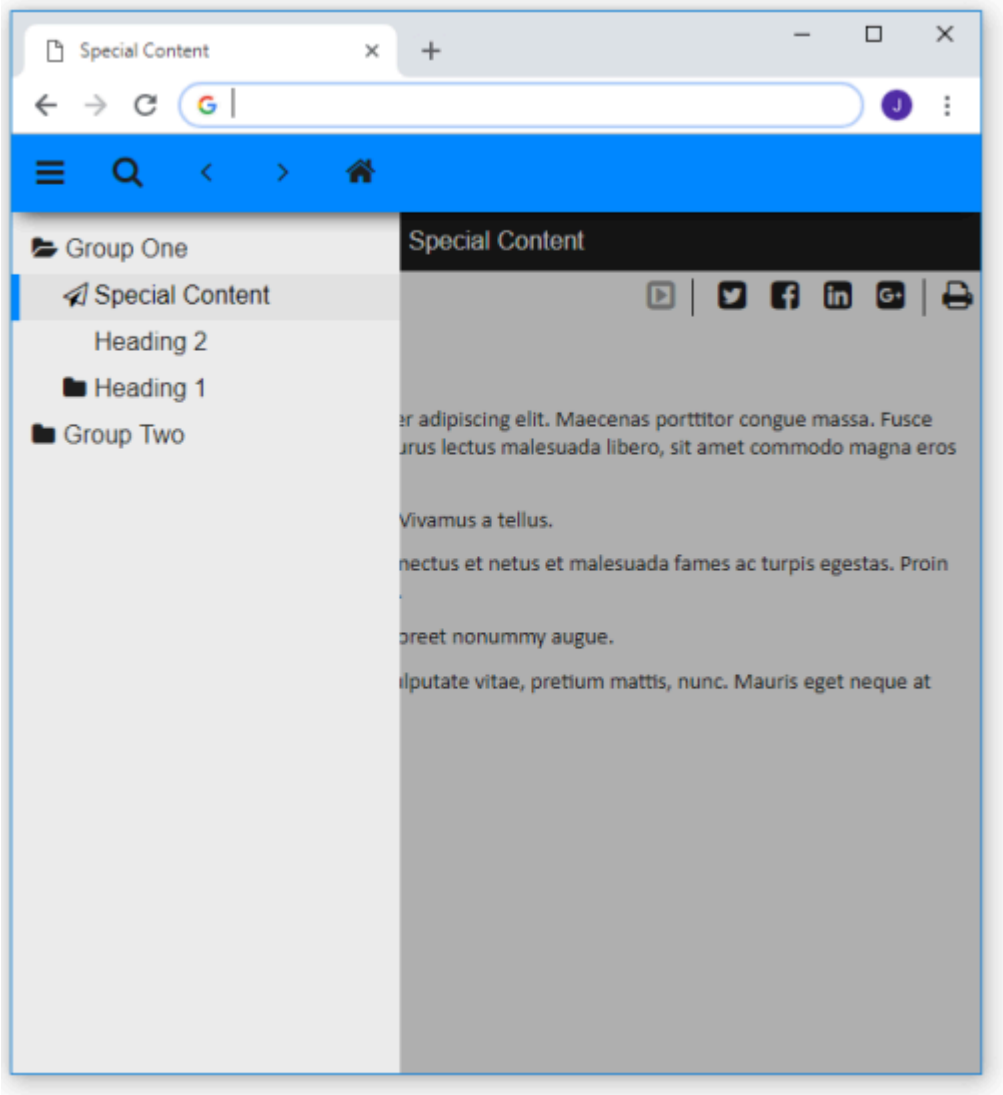
4. Give the TOCEntryClass a value. This value will become the class. In the example, `folder_icon` is used.
5. Save your Word Document.
6. Scan the document in ePublisher Designer.
7. Open the **Style Designer**.
8. Open **Marker Styles**.
9. Locate the **Marker Type Option** from the **Options** tab and set its value to `TOC Entry Class`.

10. In this example, the assigned class for the Menu TOC entry will be the value of the marker: `folder_icon`.

11. Add the following to a target override of `_icons.scss`. Notice how the CSS class is the name of the value given in the Marker Text Window. In this example we change the icon color and the icon of the TOC entry. You are able to make other customizations such as adding a border, or changing the background color.

```
.folder_icon {

  > div > span > i {

 color: black;

 &:before {

       content: $folder_icon;

    }

}

 }
```

12. Save your project and generate the output.


Special Content
Group One
Special Content
Heading 2
Heading 1
Group Two
Special Content
er adipiscing elit. Maecenas porttitor congue massa. Fusce
urus lectus malesuada libero, sit amet commodo magna eros
Vivamus a tellus.
nectus et netus et malesuada fames ac turpis egestas. Proin
oreet nonummy augue.
ılputate vitae, pretium mattis, nunc. Mauris eget neque at

Customizing Table of Contents Icons in Word

By default, the **Contents** tab in a Microsoft HTML Help, Oracle Help, and WebWorks Help uses book and page icons to identify entries. By default, the **Contents** tab in Sun JavaHelp uses folder and page icons to identify entries. You can also customize the table of contents icons.

For example, if you want to make new topics stand out by using a unique icon specific to new books, pages, or folders, you can insert a marker into a topic and specify the icon you want to display for the book, page, or folder in your help system table of contents.

To customize a table of contents icon, your Stationery and template must have the following items configured:

- TOCIconHTMLHelp for Microsoft HTML Help
- TOCIconOracleHelp for Oracle Help
- TOCIconJavaHelp for Sun JavaHelp
- TOCIconWWHelp for WebWorks Help

The following procedure provides an example of how to customize table of contents icons for topics in Microsoft Word source documents using Microsoft Word 2003. Steps for customizing table of contents icons for topics in Microsoft Word may be different in other versions of Microsoft Word.

To specify a custom table of contents icon in a Microsoft Word source document

1. ***If you want to specify a custom table of contents icon for Microsoft HTML Help***, identify the number of the image you want to use for the table of contents image for the topic in the `.hhp` file for your Microsoft HTML Help project by completing the following steps:
    a. On the **View** menu, click **Output Directory**.
    b. Open the `ProjectName` folder, where *`ProjectName`* is the name of your project.
    c. Open the `ProjectName.hhp` file where *`ProjectName`* is the name of your project.
    d. On the **Contents** tab, select a table of contents entry, and then click the **Pencil** icon.
    e. On the **Advanced** tab, in the **Image index** field, use the up and down arrows to identify the table of contents image you want to use for the topic.
    f. Note the number of the image you want to use for the table of contents image for the topic.

    For example, if you want to use a question mark icon with a red star for the table of contents icon for new topics, note that the number for this icon is 10.
    g. Close HTML Help Workshop.
2. ***If you want to specify a custom table of contents icon for Oracle Help or Sun JavaHelp***, create the graphic file for the custom table of contents icon in `.gif` format. The default graphics used as Sun JavaHelp or Oracle Help table of contents icons are 17 x 17 pixels. The custom graphics you

create for Sun JavaHelp or Oracle Help table of contents icons should also be 17 x 17 pixels. You can assign any name to the graphic files.

3. ***If you want to specify a custom table of content icon for WebWorks help***, create graphics files containing the collapsed and expanded versions of the icons you want to use, then save the graphic files in `.gif` format. The default graphics used as WebWorks Help table of contents icons are 17 x 17 pixels. The custom graphics you create for WebWorks Help table of contents icons should also be 17 x 17 pixels. You can assign any name to the graphic files.

4. Copy the graphic files you want to use as icons in the table of contents into the following folder:

   Note: If the folder does not exist, first create the folder using the specified folder structure and then copy the graphic files you want to use as icons into the folder. You do not need to perform this step when specifying custom table of contents icons for Microsoft HTML Help.

   - ***If you are generating Oracle Help***, copy the graphic files you want to use into the following folder:

     ProjectName`\Formats\Oracle Help\Files\images` folder, where *ProjectName* is the name of your project.

   - ***If you are generating Sun JavaHelp 1.1.3***, copy the graphic files you want to use into the following folder:

     ProjectName`\Formats\Sun Java Help 1.1.3\Files\images` folder, where *ProjectName* is the name of your project.

   - ***If you are generating Sun JavaHelp 2.0***, copy the graphic files you want to use into the following folder:

     ProjectName`\Formats\Sun Java Help 2.0\Files\images` folder, where *ProjectName* is the name of your project.

   - ***If you are generating WebWorks Help***, in your *ProjectName*`\Files` folder, where *ProjectName* is the name of your project, create a `wwhelp\images` subfolder and copy the graphic files you want to use into this folder. Your project file structure should be similar to the following structure:

     ProjectName`\Files\wwhelp\images,` where *ProjectName* is the name of your project.

5. In your Microsoft Word source document, locate the topic where you want to use the custom table of contents icon.

6. Insert your cursor into the heading for the topic.

7. On the **WebWorks** menu, click **Markers**.

8. In the **Marker** list, select the appropriate TOCIcon marker type from the list.

9. In the **Value** field, type the following text:

   - ***If you are generating Microsoft HTML Help***, type the number of the icon that you want to use for the table of contents image.

For example, if you want to use a question mark icon with a red star for the table of contents icon for new topics, type `10`.

- ***If you are generating Oracle Help or Sun JavaHelp***, type the following text:

  `images/TOCIcon.gif`

  where `TOCIcon.gif` is the name of the table of contents icon you want to display for the topic.

- ***If you are generating WebWorks Help***, type the following text:

  `c="collapsed.gif" e="expanded.gif"`

  where `collapsed.gif` is the name of the icon you want to use when the table of contents entry is collapsed, and `expanded.gif` is the name of the icon you want to use when the table of contents entry is expanded. If the table of contents entry is for a page instead of a book, the entry will never be expanded, so you can omit the `e="expanded.gif"` portion of the entry for pages.

  For example, you might create a special icon to highlight books that are new for a particular release of your WebWorks Help system. If you named these icons `newbookopen.gif` and `newbookclosed.gif`, you would type the following text into the **Value** field:

  `c="newbookclosed.gif" e="newbookopen.gif"`

10. Click **OK**.
11. Save your Microsoft Word source document.
12. Generate output for your project. For more information, see "Generating Output".
13. In Output Explorer, verify ePublisher created the table of contents using the table of contents icon you specified for the topic. For more information about viewing output files in Output Explorer, see "Viewing Output in Output Explorer".

Specifying Context Plug-ins in Word

You can specify Eclipse Help context plug-ins by using Context Plugin markers in your source documents. ePublisher places the context plug-ins you specify in your source documents in the `plugin.xml` file generated for each source document group you have in Document Manager. You can then have developers use the context plug-ins defined in `plugin.xml` files to call your Eclipse Help system as appropriate from Eclipse plug-ins.

For example, assume you have the following three top-level groups in Document Manager for your Eclipse Help system target:

- Component A group - contains the source documents for ComponentA Feature1 and ComponentA Feature2
- Component B group - contains the source documents for ComponentB Feature1 and ComponentB Feature 2
- Component C group - contains the source documents for ComponentC Feature1 and ComponentC Feature 2

You insert the following Context Plugin markers into the source documents for each group:

- ComponentAFeature1 and ComponentAFeature2 Context Plugin markers in source documents contained in the ComponentA group
- ComponentBFeature1 and ComponentBFeature2 Context Plugin markers in source documents contained in the ComponentB group
- ComponentCFeature1 and ComponentCFeature2 Context Plugin markers in source documents contained in the ComponentC group

When you generate your Eclipse Help system, ePublisher creates the following folder structure in the *`ProjectName`*`\Output\`*`TargetName`* folder, where *ProjectName* is the name of your ePublisher project, and *TargetName* is the name of your target:

- `ComponentA` folder, which contains a `plugin.xml` file with the following entries:

  ```
  plugin="ComponentAFeature1ContextPlugin"
  plugin="ComponentAFeature2ContextPlugin"
  ```

- `ComponentB` folder, which contains a `plugin.xml` file with the following entries:

  ```
  plugin="ComponentBFeature1ContextPlugin"
  plugin="ComponentBFeature2ContextPlugin"
  ```

- `ComponentC` folder, which contains a `plugin.xml` file with the following entries:

  ```
  plugin="ComponentCFeature1ContextPlugin"
  plugin="ComponentCFeature2ContextPlugin"
  ```

You can then provide the context plug-in IDs in your `plugin.xml` files to the appropriate Eclipse developers to use. The Eclipse developers use the context plug-ins defined in `plugin.xml` files to call your Eclipse Help system as appropriate from Eclipse plug-ins.

To specify a context plug-in in a Microsoft Word source document

1. Identify a topic in a source document where you want to insert the context plug-in.
2. On the **WebWorks** menu, click **Markers**.
3. In the **Marker Type** field, select **Context Plugin** from the list of markers.
4. In the **Value** field, type the appropriate ID for the context plug-in.

   Note: If you are responsible for defining the ID, ensure you supply the context plug-in ID to your developers to use as appropriate for their Eclipse plug-ins. If your developers define the ID, use the context plug-in ID you obtained from your developers.
5. Click **OK**.
6. Save your Microsoft Word source document.
7. Generate output for your project. For more information, see “Generating Output”.
8. In Output Explorer, verify ePublisher generated a `plugin.xml` file that contains the context plug-in IDs you specified by completing the following steps:
   a. On the **View** menu, click **Output Directory**.
   b. Open the *`ProjectName`* folder, where *`ProjectName`* is the name of your project.
   c. Open the group folder for a group that contains the source documents where you specified your context plug-in ID.
   d. Open the `plugin.xml` file in Notepad and verify that the context plug-in IDs you specified in your source documents are listed in the `plugin.xml` file. Your context plug-in IDs should be listed in the Contexts area of the file. Following is an example of the how the context plug-in IDs you specified in your source documents should be displayed in the `plugin.xml` file:

```
<!-- Contexts -->
<!-- -->
<extension point="org.eclipse.help.contexts">
<contexts file="contexts.xml" plugin="ComponentAFeature1ContextPlugin" />
</extension>
<extension point="org.eclipse.help.contexts">
<contexts file="contexts.xml" plugin="ComponentAFeature2ContextPlugin" />
</extension>
```

Creating Accessible Online Content in Word

Accessible content is content that can be easily accessed by users with certain disabilities. This section explains how you can prepare your Microsoft Word source documents to ensure your content is accessible to users using assistive technologies.

Accessible Content in Word

Images and tables are helpful ways to convey information to end users. However, users with disabilities often cannot access the important information provided by images and table layouts in online content. You should document images and other non-text items such as table layouts so that users using assistive technologies to access online content can access the information these items provide.

Content that must easily be accessed by people with disabilities must conform to certain guidelines published by both the W3C and the United States government in order to produce accessible online output, also known as Section 508 compliant output. These guidelines are intended to help writers produce accessible content.

You can use ePublisher to help you produce online content that conforms to the W3C Web Content Accessibility Guidelines 1.0 (WCAG), Section 508 of the U.S. Rehabilitation Act of 1998, and the Americans with Disabilities Act (ADA). If you are required to generate accessible content, typically you provide the following items in your online content:

- Alternate text and descriptions for all images and image maps. For more information, see "Assigning Alternate Text to Images and Image Maps in Word".
- Long descriptions for all images. For more information, see "Assigning Long Descriptions to Images in Word".
- Summaries for all tables. For more information, see "Assigning Alternate Text (Summaries) to Tables in Word".

You may also choose to provide the following items in your online content:

- Alternate text for abbreviations. For more information, see "Assigning Alternate Text to Abbreviations in Word".
- Alternate text for acronyms. For more information, see "Assigning Alternate Text to Abbreviations in Word".
- Citations for quotes. For more information, see "Providing Citations for Quotes in Word".

You must prepare source documents and configure your ePublisher project in order to create accessible content. You prepare your source documents by inserting markers into your source documents and by applying character formats and paragraph formats. You configure accessibility settings in the ePublisher project. ePublisher uses the information in your source documents and your ePublisher project to generate accessible online output.

For more information about producing accessible content and to check your content further for compliance, see the following Web sites:

- For the complete W3C note on the WCAG, visit http://www.w3c.org/TR/WCAG10-CORE-TECHS.
- For information about the related Web Accessibility Initiative, visit http://www.w3.org/WAI.
- For information about Section 508 of the U.S. Rehabilitation Act of 1998, visit http://www.w3.org/WAI/Policy/#508.

Accessible Content Navigation in Word

Users can navigate through the accessible content using keys on the keyboard. The following output formats support navigation keys:

- Dynamic HTML
- Microsoft HTML Help
- Oracle Help
- WebWorks Help

Note: For the Dynamic HTML, navigation key behavior may vary based on the browser the user uses. For example, in Netscape and Mozilla, users must hold down the `Alt` key while pressing the navigation keys. In Internet Explorer, users must first hold down the `Alt` key while pressing the navigation key, and then press `Enter.`

The following table lists the how each output format supports navigation keys.

Navigation Key	Function	Format
1	Display the TOC	• Dynamic HTML • WebWorks Help 5.0
2	Display the Index	• Dynamic HTML • WebWorks Help 5.0
3	Display the Search tab	WebWorks Help 5.0
4	Go to the previous page	• Dynamic HTML • Microsoft HTML Help • Oracle Help • WebWorks Help 5.0 ***If you are using Microsoft HTML Help***, `Alt+4` works only if the topic pane has the focus. If the topic pane does not have the focus, you must press `Alt+0` and then `Alt+4`. ***If you are using Oracle Help***, you must press Enter after pressing `Alt +4`.
5	Go to the next page	• Dynamic HTML • Microsoft HTML Help 1.x • Oracle Help • WebWorks Help 5.0 ***If you are using Microsoft HTML Help***, the `Alt+5` key works only if

Navigation Key	Function	Format
		the topic pane has the focus. If the topic pane does not have the focus, you must press `Al+ 0` and then `Alt +5`. ***If you are using Oracle Help***, you must press `Enter` after pressing `Alt+5`.
6	Shift the focus to the related topics list displayed at the bottom of the current page	WebWorks Help 5.0 After you press the `6` key, you can press `Tab` to cycle through the entries in the related topics list.
7	Display a blank feedback e-mail (equivalent to clicking the e-mail button in the toolbar frame)	WebWorks Help 5.0
8	Print the current page (equivalent to clicking the Print button in the toolbar frame)	WebWorks Help 5.0
9	Bookmark the current page (equivalent to clicking the Bookmark button in the toolbar frame)	WebWorks Help 5.0
10	Shift the focus to the topic frame (equivalent to clicking within the topic frame)	WebWorks Help 5.0

Validating Accessible Content in Word

After you configure your source documents and configure the appropriate settings, ePublisher uses Accessibility conformance reports to perform the following checks to verify that the generated output conforms to accessibility standards:

- Alternate text for all images
- Alternate text for all clickable regions in all image maps
- Long descriptions for all images
- Summaries for all tables

Note: ePublisher does not verify that you have provided alternate text for abbreviations or acronyms or verify that you have included citations for quotes. For more information about understanding and using the Accessibility conformance reports ePublisher provides, see "Configuring Reports", and "Generating Reports".

Assigning Alternate Text to Images and Image Maps in Word

This section provides information about how to create accessible images and image maps in your generated output by assigning alternate text to images.

Image and Image Map Alternate Text in Word

One of the largest accessibility challenges with online content today is the lack of alternative text for images and image maps. Sight-impaired users often use screen readers or refreshable Braille devices to read online content. However, when these assistive technologies come across images or image maps without alternative text, also known as alternate text, they are unable to provide users with information about the image or image map and its meaning.

The Web Content Accessibility Guidelines require that alternate text be provided for all images and image maps in online content. The alternate text is an image label that describes the image or each area of the image map. Online content should display alternate text for images and image maps when users perform the following actions:

- The user hovers the mouse pointer over an image or section of an image map.
- The user browser has been configured to disable display of images and image maps.
- The user browser is a text-only browser such as Lynx.
- The user uses assistive technology such as a screen reader.

The alternate text you assign to an image or sections of an image map should be as accurate and as succinct as possible and provide users with a brief description of the image and how the image relates to the page they are viewing. Make sure that your alternate text conveys all of the important information related to the image or image map section, but do not burden users with excessively long alternative text. Screen readers or refreshable Braille devices always read the alternative text, so if your page has several images or complex image maps with long descriptions, it can take a long time for the assistive devices to read image-heavy pages with long descriptions. If you need to provide a description of the image or image map section that is more than a few words or a few short sentences, you should provide a brief alternate text description of the image or image map section and then assign a longer description the image using either the `longdesc` attribute or a description. Once you specify a long description using the `longdesc` attribute, you can also optionally display a D link next to the image. For more information about assigning long descriptions to images, see “Assigning Long Descriptions to Images in Word”.

Assigning Alternate Text to Images in Word

Use the **Web** tab on the Format Picture window to assign alternate text to images in Microsoft Word source documents.

The following procedure provides an example of how to assign alternate text to images in Microsoft Word source documents using Microsoft Word 2003. Steps for assigning alternate text to images in Microsoft Word may be different in other versions of Microsoft Word.

To assign alternate text to an image in a Microsoft Word source document

1. In your Microsoft Word source document, locate the image for which you want to specify image scaling.
2. Right-click the image, and then click **Format Picture** or **Format Object**.
3. On the **Web** tab, in the **Alternative text** field, type the alternate text you want to specify for the image.
4. Click **OK**.
5. Save your Microsoft Word source document.
6. Generate output for your project. For more information, see “Generating Output”.
7. Verify ePublisher assigned the alternate text you specified to the image when it generated output by completing the following steps:
    a. On the **View** menu, click **Output Directory**.
    b. In the *`TargetName`* folder, open the page that has the image to which you assigned alternate text in Notepad, where *`TargetName`* is the name of your target.
    c. Verify that the alternate text you specified is included in the `alt` tag for the image.

Assigning Alternate Text to Image Maps in Word

Use the **Web** tab on the Format Text Box window to assign alternate text to areas of an image map in Microsoft Word source documents.

The following procedure provides an example of how to assign alternate text to an image map in Microsoft Word source documents using Microsoft Word 2003. Steps for assigning alternate text to an image map in Microsoft Word may be different in other versions of Microsoft Word.

To assign alternate text to an image map in a Microsoft Word source document

1. In your Microsoft Word source document, locate the image map for which you want to specify alternate text.
2. For each clickable area of the image map, complete the following steps:
    a. Right-click the text box that defines a clickable region for the image map.
    b. On the **Web** tab, in the **Alternative text** field, type the alternate text you want to specify for the image.
    c. Click **OK**.
3. Save your Microsoft Word source document.
4. Verify ePublisher assigned the alternate text you specified to each area of the image map when it generated output by completing the following steps:
    a. On the **View** menu, click **Output Directory**.
    b. In the *`TargetName`* folder, open the page that has the image map to which you assigned alternate text in Notepad, where *`TargetName`* is the name of your target.
    c. Verify that the alternate text you specified is included in the `alt` tag for each area of the image map.

Assigning Long Descriptions to Images in Word

This section explains how to create accessible images in your generated output by assigning long descriptions to images.

Image Long Descriptions

The Web Content Accessibility Guidelines and Section 508 guidelines require you to include long descriptions for each image in an HTML document. You can use the `longdesc` attribute and a long descriptions stored in an external `.txt` file to assign a long description to an image. When you use this approach, the long descriptions are referenced in the HTML `<img>` tag in the `longdesc` attribute as shown in the following example:

```
<img src="mission.gif" height="240" width="386" alt="The Mission"
longdesc="mission.txt" />
```

The `longdesc` attribute in the `<img>` tag provides a link to a separate page where a long description is available. The link is invisible to sighted users, but when a conformant screen reader application reads the `longdesc` attribute, it loads the file referenced in the `longdesc` attribute and reads it. In the previous example, the screen reader would load and read the `mission.txt` file.

ePublisher provides the following options for assigning long descriptions to images:

- You can use the ImageLongDescText marker to assign a long description to an image. With this method, you assign a long description to an image using a description you include in a marker you insert into your source document. For more information, see "Specifying Long Descriptions for Images in Word".
- You can use the ImageLongDescByRef marker to assign a long description to an image by referencing a long description saved in an external text (`.txt`) file. With this method, you specify the path to the external text file in a marker. For more information, see "Using Text in External Files to Assign Long Descriptions to Images in Word".

If you assign long descriptions to some, but not all of you images, you can use the ImageLongDescNotReq marker. Use this marker when you use accessibility reports to verify that all images have long description but you have certain images in your source document that do not require a long description. For more information, see "Excluding Images from Accessibility Report Checks in Word".

Although using the `longdesc` attribute is recommended in the Web Content Accessibility Guidelines and in 508 guidelines, older screen readers and many current browsers do not support this attribute and few online content developers use this attribute. As a result, the `longdesc` attributed benefits a only a small number of users. Only users who use modern screen readers can access the `longdesc` attribute easily. Older screen readers did not support this attribute. In addition, even users who use the latest version of screen reader may be unfamiliar with the `longdesc` attribute and may not know how to access long descriptions using their screen reader because the `longdesc` attribute is used so infrequently in online content.

If you use the ImageLongDescText marker to assign long descriptions to images, as an interim solution ePublisher allows you to display a D link immediately after the image. The D link is an upper case letter D link that directs users to another page that contains the text you specified in the ImageLongDescText marker. Although a D link is not required for accessible Web pages, it can be used in addition to the `longdesc` attribute. The D link technique works in all browsers, but it is less elegant than using the `londesc` attribute. Some users may be confused when they see a D link on the page, while other users will ignore the D link.

If you want to use D links in addition to the `longdesc` attribute when you generate output, your Stationery must have the D link option enabled. If you have permissions to modify target settings in ePublisher, you can enable the D link option setting in an project. For more information about enabling the D link option in an project, see “Specifying Accessibility Settings”. For more information about permissions required to modify target settings using ePublisher Express, see “Working with Target Settings”.

Specifying Long Descriptions for Images in Word

To assign a long description to an image, your Stationery and template must have the ImageLongDescText marker type configured. Your output format must also support this feature.

When you use the ImageLongDescText marker to assign long descriptions to images, ePublisher generates an external text file that contains the long description you specify. When a conformant screen reader application reads the generated page, it loads the `.txt` file referenced in the `longdesc` attribute on the page and reads the file.

The following procedure provides an example of how to specify long descriptions for images in Microsoft Word source documents using Microsoft Word 2003. Steps for specify long descriptions for images in Microsoft Word may be different in other versions of Microsoft Word.

To assign a long description to an image using marker text in a Microsoft Word source document

1. In your Microsoft Word source document, locate the image to which you want to assign a long description.
2. Right-click the image, and then click **Format Picture** or **Format Object**.
3. Change the layout setting of the image to **Top and Bottom** by completing the following steps:

   Note: By default when you insert images into Microsoft Word, Microsoft Word inserts the image using the **Inline with text layout** setting. In order to specify the image scale for image output files, you must group the image and the text box that contains the ImageLongDescText marker. However, you cannot group images using the **In line with text** layout setting in Microsoft Word. To work around this known Microsoft Word issue, if you have an image that uses an **In line with text** layout setting, use the **Top and Bottom** layout setting for the image while you insert the ImageLongDescText marker, and then reapply the **In line with text** layout setting after you group the image and the ImageLongDescText marker.

   a. On the **Layout** tab, click **Advanced**.

   b. On the **Text Wrapping** tab, click **Top and Bottom**.

   c. Click **OK**, and then click **OK** again to close the window.
4. Select your image.
5. On the **Insert** menu, click **Text Box**, and then click to the right of your image. Microsoft Word inserts a text box.
6. Insert your cursor into the text box, and then complete the following steps:

   a. On the **WebWorks** menu, click **Markers**.

   b. In the **Markers** field, select **ImageLongDescText** from the list of markers.

   c. In the **Value** field, type the long description you want to specify for the image.

d. Click **OK**. ePublisher inserts the ImageLongDescText marker into the text box.

e. Select the text box.

f. Right-click the selected text box, and then click **Format Text Box**.

g. On the **Colors and Lines** tab, in the **Fill** area, in the **Color** field, select **No Fill**.

h. In the **Line** area, in the **Color** field, select **No Line**.

i. Click **OK**.

7. Drag and drop the text box onto the image.

8. Select the text box and the image.

9. Right-click the selected text box and image, and then click **Grouping > Group**.

Note: When you select **Group**, the location of the image in your Microsoft Word source document may change in relation to the text in your source document. For example, the image may move up or down in your Microsoft Word source document. This is known Microsoft Word behavior. You may need to scroll up or down in your source document to the new location of the image to find the image.

10. ***If your image previously used the In line with text layout setting for the image***, reassign this style to your image by completing the following steps:

a. Right-click ***only*** the image, and then click **Format Object**.

Note: You must ensure you right-click ***only*** the image, and not on the text box or the grouped text box and image. If you right-click on the text box or the grouped text box and image, Microsoft Word does not display the **Format Object** menu option on the context menu.

b. On the **Layout** tab, click **In line with text**.

c. Click **OK**, and then click **OK** again to close the window.

11. Save your Microsoft Word source document.

12. Generate output for your project. For more information, see "Generating Output".

13. Verify ePublisher assigned the long description to the image by completing the following steps:

a. On the **View** menu, click **Output Directory**.

b. In the *TargetName*`\images` folder, verify that ePublisher created a `.txt` file that contains the long description you specified in the ImageLongDescText marker, where *TargetName* is the name of your target.

For example, if you specified a long description for *ImageName*`.png`, verify that ePublisher created an *ImageName*`.txt` file in the `images` folder, where *ImageName* is the name of the image to which you assigned a long description.

c. In the *TargetName*`\`*ProjectName* folder, open the page that contains the image to which you assigned the long description in Notepad and verify that the `longdesc` attribute references the *ImageName*`.txt` file ePublisher created for the image, where *TargetName* is the name

of your target, *ProjectName* is the name of your project, and *ImageName* is the name of the image to which you assigned a long description.

d. ***If you used the ImageLongDescText marker and the Stationery designer configured your Stationery to support D links***, open the page in a browser, verify that the D link displays in the browser, and then click the D link and verify that a page opens that displays the long description that you specified in the ImageLongDescText marker.

Using Text in External Files to Assign Long Descriptions to Images in Word

Use the ImageLongDescByRef marker to assign long descriptions to images using text in external files. To assign a long description to an image, your Stationery and template must have the ImageLongDescText marker type configured. Your output format must also support this feature.

The following procedure provides an example of how to use text in external files to assign long descriptions to images in Microsoft Word source documents using Microsoft Word 2003. Steps for using text in external files to assign long descriptions to images in Microsoft Word may be different in other versions of Microsoft Word.

To assign a long description to an image using marker text in a Microsoft Word source document

1. Create a `.txt` file that contains each image long description.
2. Place each image long description text file in a folder in the *`ProjectName`*`\Formats\`*`TargetName`*`\Files` folder for your project, where *`ProjectName`* is the name of your ePublisher project and *`TargetName`* is the name of your target.

   For example, place the each image long description in the following location:

   `ProjectName``\Formats\`*`TargetName`*`\Files\`*`longdescriptions`*`\`*`imagelongdescription`*`.txt`

   where *`ProjectName`* is the name of your ePublisher project, *`TargetName`* is the name of your target, *`longdescriptions`* is the name of the folder where you placed the image long description, and *`imagelongdescription`* is the name of the `.txt` file that contains the image long description.
3. In your Microsoft Word source document, locate the image to which you want to assign a long description.
4. Right-click the image, and then click **Format Picture** or **Format Object**.
5. Change the layout setting of the image to **Top and Bottom** by completing the following steps:

   Note: By default when you insert images into Microsoft Word, Microsoft Word inserts the image using the **Inline with text layout** setting. In order to specify the image scale for image output files, you must group the image and the text box that contains the ImageLongDescByRef marker. However, you cannot group images using the **In line with text** layout setting in Microsoft Word. To work around this known Microsoft Word issue, if you have an image that uses an **In line with text** layout setting, use the **Top and Bottom** layout setting for the image while you insert the ImageLongDescText marker, and then reapply the **In line with text** layout setting after you group the image and the ImageLongDescText marker.

   a. On the **Layout** tab, click **Advanced**.

   b. On the **Text Wrapping** tab, click **Top and Bottom**.

   c. Click **OK**, and then click **OK** again to close the window.
6. Select your image.

7. On the **Insert** menu, click **Text Box**, and then click to the right of your image. Microsoft Word inserts a text box.

8. Insert your cursor into the text box, and then complete the following steps:

   a. On the **WebWorks** menu, click **Markers**.

   b. In the **Markers** field, select ImageLongDescByRef from the list of markers.

   c. In the **Value** field, type the path to the `.txt` file that contains the long description you want to assign to the image.

      For example, type:

      `./longdescriptions/imagelongdescription.txt`

      where *`longdescriptions`* is the name of the folder where you placed the image long description, and *`imagelongdescription`* is the name of the `.txt` file that contains the image long description.

   d. Click **OK**. ePublisher inserts the ImageLongDescText marker into the text box.

   e. Select the text box.

   f. Right-click the selected text box, and then click **Format Text Box**.

   g. On the **Colors and Lines** tab, in the **Fill** area, in the **Color** field, select **No Fill**.

   h. In the **Line** area, in the **Color** field, select **No Line**.

   i. Click **OK**.

9. Drag and drop the text box onto the image.

10. Select the text box and the image.

11. Right-click the selected text box and image, and then click **Grouping > Group**.

    Note: When you select **Group**, the location of the image in your Microsoft Word source document may change in relation to the text in your source document. For example, the image may move up or down in your Microsoft Word source document. This is known Microsoft Word behavior. You may need to scroll up or down in your source document to the new location of the image to find the image.

12. ***If your image previously used the In line with text layout setting for the image***, reassign this style to your image by completing the following steps:

    a. Right-click ***only*** the image, and then click **Format Object**.

       Note: You must ensure you right-click ***only*** the image, and not on the text box or the grouped text box and image. If you right-click on the text box or the grouped text box and image, Microsoft Word does not display the **Format Object** menu option on the context menu.

    b. On the **Layout** tab, click **In line with text**.

    c. Click **OK**, and then click **OK** again to close the window.

13. Save your Microsoft Word source document.

14. Generate output for your project. For more information, see “Generating Output”.

15. In Output Explorer, verify ePublisher assigned the long description to the image using the long description in the external file when it generated output by completing the following steps:

    a. On the **View** menu, click **Output Directory**.

    b. In the *TargetName**ProjectName* folder, open the page that contains the image to which you assigned the long description using an external file in Notepad and verify that the `longdesc` attribute references the external text file that contains the long description for the image, where *TargetName* is the name of your target, and *ProjectName* is the name of your project.

Excluding Images from Accessibility Report Checks in Word

In some instances, alternate text is sufficient for an image, and assigning a long description to an image in addition to alternate text would be redundant. However, you may have configured Accessibility reports to check for images without long descriptions and notify you when an image does not have a long description.

In this scenario, while you want an Accessibility report to notify you when you have an image without a long description, you do not want to be notified when you deliberately did not assign a long description to an image because assigning a both a long description and alternative text would be redundant. To address this issue, you can use the ImageLongDescNotReq marker to exclude an image that deliberately does not have a long description from validation when you generate Accessibility reports. For more information about Accessibility reports and configuring and generating Accessibility reports, see “Accessibility Reports”, “Configuring Reports”, and “Generating Reports”.

To exclude images without long descriptions from Accessibility reports, your Stationery and template must have the ImageLongDescNotReq marker type configured. Your output format must also support this feature.

The following procedure provides an example of how to exclude images without long descriptions from Accessibility report checks in Microsoft Word source documents using Microsoft Word 2003. Steps for excluding images without long descriptions from Accessibility report checks in Microsoft Word may be different in other versions of Microsoft Word.

To exclude an image without a long description from Accessibility report checks in a Microsoft Word source document

1. In your Microsoft Word source document, locate the image without a long description that you want to exclude from an Accessibility report check.
2. Change the layout setting of the image to **Top and Bottom** by completing the following steps:

   Note: By default when you insert images into Microsoft Word, Microsoft Word inserts the image using the **Inline with text layout** setting. In order to specify the image scale for image output files, you must group the image and the text box that contains the ImageLongDescNotReq marker. However, you cannot group images using the **In line with text** layout setting in Microsoft Word. To work around this known Microsoft Word issue, if you have an image that uses an **In line with text** layout setting, use the **Top and Bottom** layout setting for the image while you insert the ImageLongDescNotReq marker, and then reapply the **In line with text** layout setting after you group the image and the ImageLongDescNotReq marker.

   a. On the **Layout** tab, click **Advanced**.

   b. On the **Text Wrapping** tab, click **Top and Bottom**.

   c. Click **OK**, and then click **OK** again to close the window.
3. Select your image.

4. On the **Insert** menu, click **Text Box**, and then click to the right of your image. Microsoft Word inserts a text box.
5. Insert your cursor into the text box, and then complete the following steps:
    a. On the **WebWorks** menu, click **Markers**.
    b. In the **Markers** field, select ImageLongDescNotReq from the list of markers.
    c. In the **Value** field, do not enter any text. You do not need to enter any text in this field when you insert a ImageLongDescNotReq marker.
    d. Click **OK**. ePublisher inserts the ImageLongDescText marker into the text box.
    e. Select the text box.
    f. Right-click the selected text box, and then click **Format Text Box**.
    g. On the **Colors and Lines** tab, in the **Fill** area, in the **Color** field, select **No Fill**.
    h. In the **Line** area, in the **Color** field, select **No Line**.
    i. Click **OK**.
6. Drag and drop the text box onto the image.
7. Select the text box and the image.
8. Right-click the selected text box and image, and then click **Grouping > Group**.

    Note: When you select **Group**, the location of the image in your Microsoft Word source document may change in relation to the text in your source document. For example, the image may move up or down in your Microsoft Word source document. This is known Microsoft Word behavior. You may need to scroll up or down in your source document to the new location of the image to find the image.
9. ***If your image previously used the In line with text layout setting for the image***, reassign this style to your image by completing the following steps:
    a. Right-click ***only*** the image, and then click **Format Object**.

        Note: You must ensure you right-click ***only*** the image, and not on the text box or the grouped text box and image. If you right-click on the text box or the grouped text box and image, Microsoft Word does not display the **Format Object** menu option on the context menu.
    b. On the **Layout** tab, click **In line with text**.
    c. Click **OK**, and then click **OK** again to close the window.
10. Save your Microsoft Word source document.
11. Generate output for your project. For more information, see "Generating Output".
12. Generate an Accessibility report and confirm that ePublisher did not generate an `Image is missing a long description` message for the image. For more information about

generating Accessibility reports and Accessibility report messages, see “Generating Reports” and “Accessibility Report Messages”.

Assigning Alternate Text (Summaries) to Tables in Word

Tables, just like images, are a way to visually display information. Although tables typically contain text, the purpose of the table is often not evident from text alone. The organization and display of the table may contain information that is not evident to assistive technologies. However, through the use of table summaries, assistive technologies can convey useful information to users about tables. The Web Content Accessibility Guidelines recommend that you provide summary text for each table in an HTML document. Table alternate text, or table summaries, provide users with information about what type of information the table contains.

You can create accessible tables by typing the table summary into a TableSummary marker. When ePublisher generates content, ePublisher puts the table summary you specify into the table in the `summary` attribute.

To assign alternate text to tables, your Stationery and template must have the TableSummary marker type configured. Your output format must also support this feature.

The following procedure provides an example of how to assign alternate text to tables in Microsoft Word source documents using Microsoft Word 2003. Steps for assigning alternate text to tables in Microsoft Word may be different in other versions of Microsoft Word.

To assign table summaries in a Microsoft Word source document

1. In your Microsoft Word source document, locate the table to which you want to assign a table summary.
2. Insert your cursor in front of the table.
3. On the **WebWorks** menu, click **Markers.**
4. In the **Markers** field, select **TableSummary** from the list of markers.
5. In the **Value** field, type the alternate text for the table.
6. Click **OK**. ePublisher inserts the TableSummary marker into the table.
7. Insert the marker into the table caption by clicking **OK**.
8. Save your Microsoft Word source document.
9. Generate output for your project. For more information, see “Generating Output”.
10. Verify ePublisher assigned the table summary you specified to the table when it generated output by completing the following steps:
    a. On the **View** menu, click **Output Directory**.
    b. In the *`TargetName`* folder, open the page that has the table to which you assigned a table summary in Notepad, where *`TargetName`* is the name of your target.
    c. Verify that the table summary you specified is included in the `summary` attribute for the table.

Excluding Tables from Accessibility Report Checks in Word

Tables used specifically for layout may not need a table summary. For example, if you use a table for layout, you probably would not assign a table summary to the table. However, you may have configured Accessibility reports to check for tables without table summaries and notify you when a table does not have a table summary.

In this scenario, while you want an Accessibility report to notify you when you have a table without a table summary, you do not want to be notified when you deliberately did not assign a table summary to a table because a table summary is not required. To address this issue, you can use the TableSummaryNotReq marker to exclude a table that deliberately does not have a table summary from validation when you generate Accessibility reports. For more information about Accessibility reports and configuring and generating Accessibility reports, see “Accessibility Reports”, “Configuring Reports”, and “Generating Reports”.

To exclude tables from Accessibility report checks, your Stationery must have the TableSummaryNotReq marker type configured. Your output format must also support this feature.

The following procedure provides an example of how to exclude tables without table summaries from Accessibility report checks in Microsoft Word source documents using Microsoft Word 2003. Steps for excluding tables without table summaries from Accessibility report checks in Microsoft Word may be different in other versions of Microsoft Word.

To exclude a table with a table summary from Accessibility report checks in a in Microsoft Word source document

1. In your Microsoft Word source document, locate the table without a table summary that you want to exclude from an Accessibility report check.
2. Insert your cursor in front of the table.
3. On the **WebWorks** menu, click **Markers.**
4. In the **Markers** field, select **TableSummaryNotReq** from the list of markers.
5. In the **Value** field, do not enter any text. You do not need to enter any text in this field when you insert a TableSummaryNotReq marker.
6. Click **OK**. ePublisher inserts the TableSummaryNotReq marker into the table.
7. Save your Microsoft Word source document.
8. Generate output for your project. For more information, see “Generating Output”.
9. Generate the Accessibility report and confirm that ePublisher did not generate an `Table is missing a table summary` message for the table. For more information about generating Accessibility reports and Accessibility report messages, see “Generating Reports” and “Accessibility Report Messages”.

Assigning Alternate Text to Abbreviations in Word

Abbreviations are often used in written communication. Using an Abbreviation character style and an AbbreviationTitle marker, you can specify alternate text for abbreviations. For example, if your source document includes an abbreviation such as SS#, you can specify Social Security Number as alternate text for the abbreviation. When you use an AbbreviationTitle marker and Abbreviation character style to specify alternate text for an abbreviation, ePublisher adds the abbreviation alternate text you specify to the `title` attribute of the `abbr` tag in the output.

Following is an example of the HTML code produced when you specify Social Security Number as alternate text for SS#.

```
<th>First name</th>
<th><abbr title="Social Security Number">SS#</abbr></th>
```

To assign alternate text to abbreviations, your Stationery and template must have the following items configured:

- Abbreviation character style
- AbbreviationTitle marker type

Your output format must also support this feature.

The following procedure provides an example of how to specify alternate text for abbreviations in Microsoft Word source documents using Microsoft Word 2003. Steps for specifying alternate text for abbreviations in Microsoft Word may be different in other versions of Microsoft Word.

To specify alternate text for an abbreviation in a Microsoft Word source document

1. In your Microsoft Word source document, locate the abbreviation for which you want to specify alternate text.
2. Apply the AbbreviationTitle character style to the abbreviation text.
3. Insert your cursor anywhere inside the abbreviation.
4. On the **WebWorks** menu, click **Markers.**
5. In the **Markers** field, select **AbbreviationTitle** from the list of markers.
6. In the **Value** field, type the abbreviation alternate text.
7. Click **OK**. ePublisher inserts the AbbreviationTitle marker into the abbreviation.
8. Save your Microsoft Word source document.
9. Generate output for your project. For more information, see “Generating Output”.
10. Verify ePublisher assigned the abbreviation alternate text you specified when it generated output by completing the following steps:
    a. On the **View** menu, click **Output Directory**.

b. In the *TargetName* folder, open the page that has the abbreviation to which you assigned alternate text in Notepad, where *TargetName* is the name of your target.

c. Verify that the alternate text you specified for the abbreviation is included in the `abbr` tag in the `title` attribute.

Assigning Alternate Text to Acronyms in Word

Acronyms are often used in written communication. Using an Acronym character style and an AcronymTitle marker, you can specify alternate text for acronyms. For example, if your document includes an acronym like NATO you can specify North Atlantic Treaty Organization as alternate text for the acronym. When you use an AcronymTitle marker and an Acronym character style to specify alternate text for an acronym, ePublisher adds the acronym alternate text you specify to the `title` attribute of the `acronym` tag in the output.

Following is an example of the HTML code produced when you specify North Atlantic Treaty Organization as alternate text for NATO.

```
<p><acronym title=”North Atlantic Treaty Organization”>NATO</acronym> is a military
 alliance.<p>
```

To assign alternate text to acronyms, your Stationery and template must have the following items configured:

- Acronym character style
- AcronymTitle marker type

Your output format must also support this feature.

The following procedure provides an example of how to specify alternate text for acronyms in Microsoft Word source documents using Microsoft Word 2003. Steps for specifying alternate text for acronyms in Microsoft Word may be different in other versions of Microsoft Word.

To specify alternate text for an acronym in a Microsoft Word source document

1. In your Microsoft Word source document, locate the acronymn for which you want to specify alternate text.
2. Apply the AcronymTitle character style to the abbreviation text.
3. Insert your cursor anywhere inside the abbreviation.
4. On the **WebWorks** menu, click **Markers.**
5. In the **Markers** field, select **AcronymTitle** from the list of markers.
6. In the **Value** field, type the acronym alternate text.
7. Click **OK**. ePublisher inserts the AcronymTitle marker into the abbreviation.
8. Save your Microsoft Word source document.
9. Generate output for your project. For more information, see “Generating Output”.
10. Verify ePublisher assigned the acronym alternate text you specified when it generated output by completing the following steps:
    a. On the **View** menu, click **Output Directory**.

b. In the *`TargetName`* folder, open the page that has the acronym to which you assigned alternate text in Notepad, where *`TargetName`* is the name of your target.

c. Verify that the alternate text you specified for the acronym is included in the `acronym` tag in the `title` attribute.

Providing Citations for Quotes in Word

A **citation** is a reference or footnote to a book, article, or other material that specifies the source from which a quotation was borrowed. A citation contains all the information necessary to identify and locate the work. Using a Citation character style and the Citation marker, you can specify citations for quotes that enable users to go to a Web site that contains additional information about the quote.

Following is an example of the HTML code produced when you specify a citation for a quote.

```
<blockquote cite="http://shakespeare.mit.edu/lll/full.html"> <p>Remuneration! O!
 that's the Latin word for three farthings.
--- William Shakespeare (Love's Labor Lost).</p> </blockquote>
```

To provide citations for quotes, your Stationery and template must have the following items configured:

- Citation character style
- Citation marker type

Your output format must also support this feature.

The following procedure provides an example of how to specify citations for quotes in Microsoft Word source documents using Microsoft Word 2003. Steps for specifying citations for quotes in Microsoft Word may be different in other versions of Microsoft Word.

To specify citations for quotes in a Microsoft Word source document

1. In your Microsoft Word source document, locate the quotation for which you want to specify a citation.
2. ***If the quotation is a phrase within a paragraph***, complete the following steps:
    a. Apply the Citation character style to the quotation phrase.
    b. Insert your cursor anywhere inside the quotation phrase.
3. ***If the quotation is a full paragraph***, insert your cursor into the paragraph.
    c. On the **Special** menu, click **Marker**.
4. On the **WebWorks** menu, click **Markers.**
5. In the **Markers** field, select **Citation** from the list of markers.
6. In the **Value** field, type the URL for the citation.
7. Click **OK**. ePublisher inserts the Citation marker into the abbreviation.
8. Save your Microsoft Word source document.
9. Generate output for your project. For more information, see “Generating Output”.

10. Verify ePublisher created the citation you specified when it generated output by completing the following steps:

    a. On the **View** menu, click **Output Directory**.

    b. In the *`TargetName`* folder, open the page that has the quotation for which you specified a quotation in Notepad, where *`TargetName`* is the name of your target.

    c. Verify that the citation you specified for the quotation is included in the `cite` attribute.

Troubleshooting Word issues

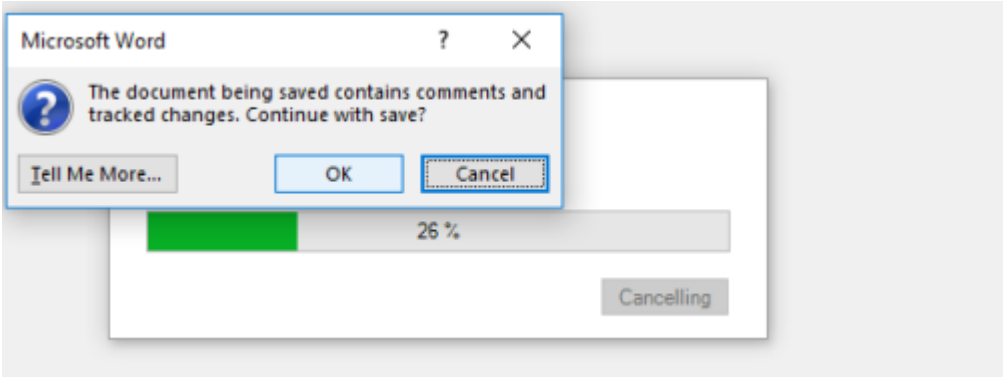
Occasionally there might be issues with the source documents you are using. Below is a list linking to the wiki solutions website that will help you troubleshoot each one:

Issue	For more information, see to...
If you are seeing hidden text in the output	Hidden text in output
If you are seeing inconsistent bullets in output	Inconsistent bullet sizes
If you are seeing an “infinite” number of transit menus in Word 2003	Transit Menu displays "infinite" menu syndrome
If you are using relative images	Relative Images
If you are using master docs	Using Microsoft Word Master Docs
If conversions hang and your documents track changes or comments	Word warning dialogs that interrupt conversions

Word warning dialogs that interrupt conversions

Conversions in ePublisher being stuck because of warning dialogs or just hanging when you are using Word source documents and you have comments or tracking the changes, might be cause by a configuration in your Word application.

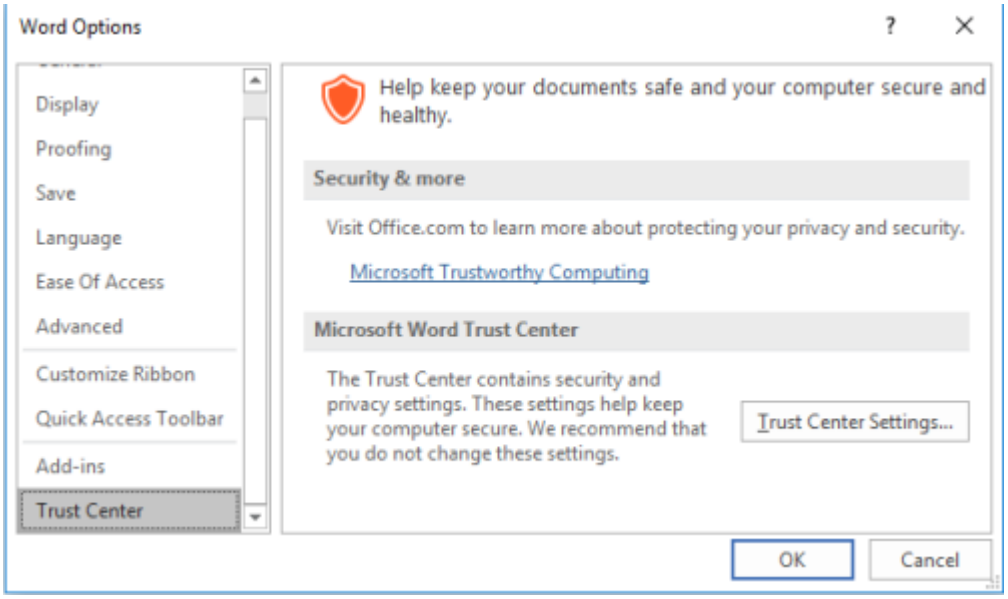
A warning dialog you might see during a conversion in ePublisher should look similar to the following figure.



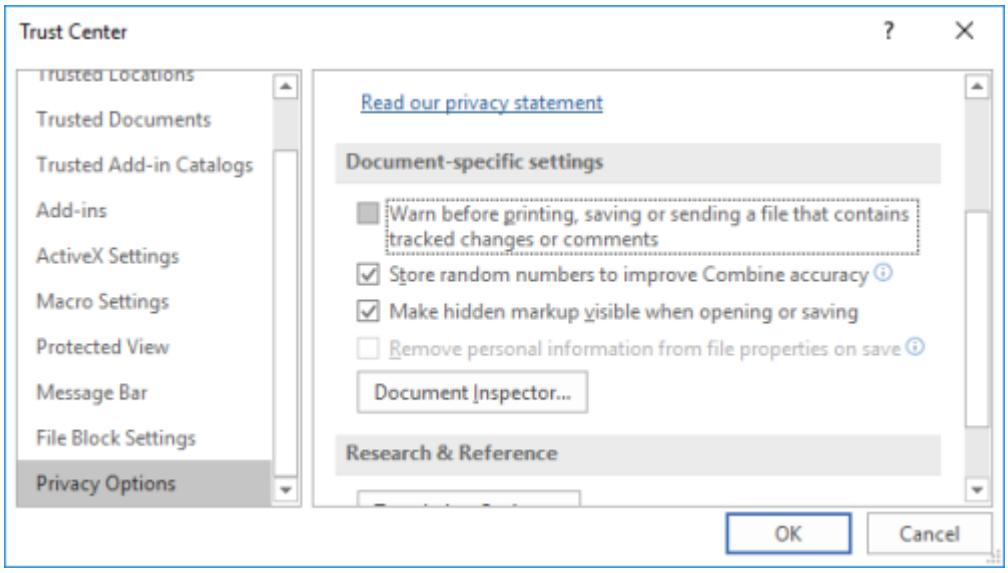
The following procedure provides an example of how to disable the “Warn before printing, saving or sending a file that contains tracked changes or comments” option in Microsoft Word source documents using Microsoft Word 2016.

To disable the “Warn before printing, saving or sending a file that contains tracked changes or comments” in a Microsoft Word source document

1. Open a Microsoft Word document, or even a Blank document.
2. On the **Word** menu, click **File** and then **Options**.
3. In the window that pops out in the left side menu click on **Trust Center**, and on the right side select **Trust Center Settings**. You should see a window similar to the following figure.



4. After selecting **Trust Center Settings** you'll see a new window, click on the left side on **Privacy Options** and on the right panel uncheck **Warn before printing, saving or sending a file that contains tracked changes or comments**. You should see a window similar to the following figure.



5. And finally click **OK**.

DITA - XML

Before you can generate output using DITA source documents, you need to prepare your DITA source documents for output generation. This section explains how to prepare your DITA source files for output generation.

DITA Usage Standards

DITA (Darwin Information Typing Architecture) is an XML-based format for creating and publishing technical content. DITA leverages the strengths of XML and provides a standard set of element definitions used to create technical content. Within the topic types based on the standard topic definition, DITA specifies the information elements used to define the content, such as the title, paragraph, table, and list elements.

This section describes the design usage considerations for DITA source documents. By reviewing how ePublisher processes DITA source documents, you can streamline single-sourcing processes and reduce your production and maintenance costs. This section does not describe all DITA details, but it focuses on the design and usage considerations related to ePublisher.

DITA Standards for Single-Sourcing

ePublisher accepts ditamaps as input source documents. The ditamap identifies the structure and order of the XML files and the DTD. The DTD defines the classes for each element, such as topic/topic and task/task. ePublisher uses these classes to process the content. ePublisher also allows you to add individual XML files as source documents to a project. However, if you publish individual XML files without using a ditamap, links will not be generated in the output.

Note: Publishing individual XML files is only supported when using the DITA-OT version 1.8 and above.

Mapping DITA Classes to ePublisher Styles

The `default.wwconfig` file maps classes to styles in ePublisher, defining both the style type and the style name. You can override the `default.wwconfig` file to specify your own style names and types. The override file does not need to be comprehensive. Your override file can build on the default file. When the same match is defined in both the customized override file and the default file, the match in the customized file overrides the match in default file.

You can override the `default.wwconfig` file for all output formats, a specific output format, or a specific target. The `default.wwconfig` file can also emit WIF information, such as `ww` content for bulleted and numbered list items. For more information, see the default `default.wwconfig` file in the following folder:

```
<ePublisher Installation Location>\ePublisher Designer\Adapters\xml\scripts\dita
```

For more information about override files and locations, see “Stationery, Projects, and Overrides”.

To specify your own class mappings with an override file

1. ***If you want to override the class mappings for all output formats***, complete the following steps:
   a. In your Stationery design project, on the **View** menu, click **Format Override Directory**.
   b. Create the `Formats\Adapters\xml\scripts\dita` folder in your project folder.
2. ***If you want to override the class mappings for one output format***, complete the following steps:
   a. In your Stationery design project, on the **View** menu, click **Format Override Directory**.
   b. Create the `Formats\Adapters\`*`formattype`*`\xml\scripts\dita` folder in your project folder, where *`formattype`* is the name of the output format you want to override, such as `WebWorks Reverb 2.0`.
3. ***If you want to override the class mappings for a target***, complete the following steps:
   a. In your Stationery design project, on the **View** menu, click **Target Override Directory**.
   b. Create the `Targets\`*`targetname`*`\Adapters\xml\scripts\dita` folder in your project folder, where *`targetname`* is the name of the target you want to override.
4. Copy the `default.wwconfig` file from the following folder to the override folder you created within your project folder:

   `Program Files\WebWorks\ePublisher Designer\Adapters\xml\scripts\dita`
5. Open the `default.wwconfig` file you copied to your project override folder.
6. Remove any mappings you do not want to override.
7. Add the mappings you want to override, including a match statement that defines the style name and a match statement that defines the style type.
8. Save and close the `default.wwconfig` file you copied to your project override folder.

The `default.wwconfig` file provides two sections:

`<Styles>` section

Provides the style match statements that map DITA classes to style names in ePublisher. Each style match statement, such as `<Style match=...`, defines the classes and parent classes to use for a match. When a class matches the style match statement, the XSL specifies the ePublisher style name to associate with that class.

`<Types>` section

Provides the type match statements that map DITA classes to style types in ePublisher. Each type match statement, such as `<Type match=...`, defines the classes to use for a match. When a class matches the type match statement, the value attribute specifies the ePublisher style type for the style associated with that class.

Defining Online Features with DITA

You assign formatting and features to styles in ePublisher. These style definitions are stored in your Stationery and mapped to DITA classes. For some online features, such as index entries, you use the standard DITA elements to implement the feature. For more information about creating Stationery and implementing online features, see “Designing, Deploying, and Managing Stationery”.

Review the following considerations to understand how to implement each online feature using DITA elements and the Stationery options within ePublisher:

- For index entries, use the standard DITA tag.
- For related topics, assign related topics options in ePublisher to paragraph styles for related-links elements, such as the Related Task and Related Concept styles.
- For expand/collapse sections, assign dropdown options in ePublisher to the appropriate paragraph styles. You cannot use a marker to identify the end of the expand/collapse section, so you need to use a paragraph style to identify the end.
- For conditions, use attributes and ditaval file filtering, for more information, See “Using Ditaval files in DITA”.
- For variables, use conref and ditaval file filtering.
- For popup windows, use the xref element for the link and define paragraph styles with the popup options in ePublisher.
- For specifying file names, use the othermeta element to define the marker with the name for the file.
- For specifying a topic alias for context-sensitive help links, use the othermeta element to define the marker with the value of the topic ID for that topic.
- For meta tag keywords, use the othermeta element to define the marker with the keywords you want to include in the meta tag in the generated output.
- For opening a topic in a custom window, use the othermeta element to define the marker with the name of the window in which to open the topic.
- For a custom TOC icon, use the othermeta element to define the marker with the name of the TOC icon to use for the generated topic.
- For See Also links, use related topics, which are available in more output formats. See Also link support is being reviewed for future enhancements.
- For accessibility features, such as image alternate text and long descriptions, use standard DITA elements and attributes.

Configuring DITA Open Toolkit Version

In order to support DITA specialization and customization it is sometimes necessary to specify a specific version of the DITA Open Toolkit (OTK). In ePublisher, you can easily change which version of the OTK is used to preprocess all of your content on a project by project basis. You can even configure your stationery to use a specific version.

To change your project's OTK version

1. In your Stationery design project, on the **Project** menu, click **Project Settings...**
2. Expand the **XML Input** tab and locate the row labeled: `DITA Open Toolkit version`.
3. Choose from one of the available versions and then select the **OK** button.

   Note: Usually, it is fine to just use the latest version, however, it may be necessary to use an older version if you have specializations or other customizations that have not yet been migrated.

Customizing the DITA DTD

You can associate an instance of the DITA-OT with your Stationery design project to apply and track DITA specialization through the Stationery.

To customize the DITA DTD and apply the DITA specialization

1. In your Stationery design project, on the **View** menu, click **Project Directory**. For more information about override files and locations, see “Stationery, Projects, and Overrides”.
2. Create the `Formats\Helpers\dita-ot-<version>` folder in your *projectname* folder, where *projectname* is the name of your Stationery design project.
3. Copy the contents of the following folder to the override folder you created within your project folder:

   `Program Files\WebWorks\ePublisher Designer\Helpers\dita-ot-<version>`
4. Open the `dita-ot-<version>` folder you copied to your project override folder and apply the DTD changes you need for your DITA specialization. Typically this means running the `ant integrate` task to install your DITA-OT plug-ins.
5. Save and close the modified files.
6. Regenerate your project to review the changes.

DITA Specialization

Structure is mapped to style in dita, for more information, please refer to the following website:

http://wiki.webworks.com/HelpCenter/Tips/DITA/All Versions/DITA Configuration

DITA Support

ePublisher supports all current OASIS DITA standards, which includes versions 1.1, 1.2, and 1.3. For information on the complete set of DITA 1.3 topic types, elements, attributes, and other specifications you may want to refer to:

https://www.webworks.com/Documentation/DITA_1.3_Language_Reference/.

Keyref elements

You can use the "keyref" feature in DITA as an indirect addressing mechanism. Instead of linking directly to topics or maps, these can be given a symbolic name (key attribute) that points to a topic file path (href attribute). References to the topics are made using a key reference (keyref attribute). If the topic is relocated, the path needs to be updated only in the map where it is defined. All other references will automatically pick up the new location.

Conref extensions

DITA uses of the conref element.

- Conref Range - Allows a single element to reference a range of elements.
- Conref Push - Most commonly used when your component adds to an existing component.
- Conkeyref - Allows you to use a key in the `conref` attribute so you can more easily change the target of the `conref`.

Using Ditaval files in DITA

For conditional text, DITA uses filtering to determine the user or audience that will be viewing the help content. Using a file called a “ditaval” file, you can control the filtering of the content that will be generated as output. You can place ditaval files next to ditamap files using the name of the DITA map files that they are to be applied to similar to this:

<ditamap name>.ditaval

With ePublisher you have 3 additional ways to handle ditaval files. The ditaval definitions are looked for in the following order.

- Per Target: Targets\<Target Name>\Adapters\xml\scripts\dita\default.ditaval
- Per Format: Formats\<Format Name>\Adapters\xml\scripts\dita\default.ditaval
- Entire project: Formats\Adapters\xml\scripts\dita\default.ditaval

The following example shows an example of how filtering works with the ditaval markup:

```
<val>
  <prop att="audience" val="web" action="exclude" />
</val>
```

The attributes within prop determine the action and who sees the material that is defined in the output.

action

include - Includes the content in output. This is the default behavior unless otherwise set.

exclude - Excludes the content from output (if all values in the particular attribute are excluded).

passthrough - Include the content in output, and preserve the attribute value as part of the output stream for further processing by a runtime engine, for example runtime filtering based on individual user settings

flag - include and flag the content on output (if the content has not been excluded).

att

The attribute to be acted. This must be one of props, audience, platform, product, otherprops, or a specialization of props. If the att attribute is absent, then the prop element declares a default behavior for any conditional processing attribute.

val

The value to be acted upon. If the val attribute is absent, then the prop element declares a default behavior for any value in the specified attribute.

Using Passthrough outputclass in DITA

You may want to include some kind of HTML or Javascript in your output that would not be generated as normal output. For this example we are going to use a passthrough attribtue on a paragraph tag in DITA. This will create a Passthrough paragraph style that you would switch to Passthrough in the paragraph options tab in ePublisher.

Below is an example of a Javascript alert in HTML using the CDATA attribute

```
<p outputclass="Passthrough">
<![CDATA[
<div><a href="alert('Yahoo!'">Click Me!</a></div>
]]>
</p>
```

Once you have entered in the paragraph that contains the passthrough attirbute, re-scan the documennt and make sure that this passthrough paragraph has the “Passthrough” option enabled so that the HTML will not be processed.

Depending on the output, for example, browser-based (Dynamic HTML or WebWorks Help 5.0) versus PDF output, you may want to use ditaval filtering to ensure that only the web-based output is getting output. For this instance, you would want to have the following DITA markup

```
<p outputclass="Passthrough" audience="web">
<![CDATA[
<div><a href="alert('Yahoo!'">Click Me!</a></div>
]]>
</p>
```

For the PDF output, for example you would want to have the Target override:

```
Targets\PDF\Adapters\xml\scripts\dita
```

and the default.ditval information:

```
<val>
  <prop att="audience" val="web" action="exclude" />
</val>
```

For the Dynamic HTML output you would want to have the Target override:

```
Targets\Dynamic HTML\Adapters\xml\scripts\dita
```

and the default.ditval information:

```
<val>
  <prop att="audience" val="web" action="include" />
</val>
```

Note: Depending on what your Target name and input is like, the information may be different than what is listed above. For more information on Target overrides, See “Creating Target Overrides”.

Embedding a Video in DITA Source Documents

This section explains how you can embed videos in DITA source documents for ePublisher. For embedded videos we only allow the mp4 and flv formats.

Embedding a Video file

You can embed a video that you have on file into your DITA source document.

To embed a video from your file directory:

1. Create the an object tag for the video, for example: `<object> </object>`
2. Next you are going to want to specify certain attributes for your video object
    - If want to set a stylename, set the outputclass attribute to whatever stylename you want to give it.
    - Specify what width and height you want to give to your video in the width and height attributes
    - Set the data attribute to the path to the video file relative to the source file
    - For example: `<object outputclass="Foo" width="560" height="320" data="../../Video/small.mp4"> </object>`
3. Optional: if you want controls on your video include the following param tag inside your object tag: `<param name=”controls” value=”true”>`

The following is a fully working code example of the following steps:

```
<object outputclass="Foo" width="560" height="320" data="../../Video/small.mp4">
  <param name="allowFullScreen" value="true" />
  <param name="controls" value="true" />
</object>
```

Linking to a Youtube Video

You can embed a Youtube video in your DITA source document.

To embed a Youtube Videa:

1. Create the an object tag for the video, for example: `<object> </object>`
2. Make sure that you are using the **Embed** URL and not the default URL provided in your browser when on the YouTube website. To obtain the proper URL, follow the instructions for obtaining a share link to embed in a website.
3. Next you are going to want to specify certain attributes for your video object
    - Specify what width and height you want to give to your video in the width and height attributes.
    - Set the data attribute to the link to the video file (optional).
    - For example: `<object width="560" height="320" data="https://www.youtube.com/embed/Fy1SB0ClS0k"> </object>`

    Note: The proper URL should contain “embed” in the path.
4. Next you can set certain param tags for your video depending on your own specifications

Note: The `data` attribute on the `object` element is optional.

The following is a fully working code example of the following steps:

```
<object outputclass="Foo" width="560" height="320">
  <param name="movie" value="https://www.youtube.com/embed/Fy1SB0ClS0k" />
  <param name="controls" value="true" />
</object>
```

Creating Context-Sensitive Help in DITA Source Documents

This section explains how you can use ePublisher to create links to context-sensitive help content in DITA source documents

Context-Sensitive Help

Context-sensitive help provides immediate assistance and information to users without requiring users to leave the context in which they are working. It helps answer questions like "What is this?" and "Why would I use this?", and provides information for a particular object and its context.

For example, in many applications, user interface controls such as windows and tabs have a help button. When users click on the help button, the application links users to a help topic specific to the context of the window. Some applications also embedded context-sensitive help topics into the window itself as an HTML pane. The application relies on an identifier such as a topic ID or file name to identify the specific help topic to display.

There are several methods for creating context-sensitive help links. In addition, different use different mechanisms to support context-sensitive help links. For example, some , such as Microsoft HTML Help, create a map file using topic aliases. Applications then use the topic IDs in the map file to provide links to context-sensitive help topics from within the application. Other do not have a mapping mechanism. However, these may support creating links to context-sensitive help topics using file names.

Map Files

Many application support the use of map files to deliver context-sensitive help. The topic IDs and map numbers are listed in a map file, which is a text file that typically has a `.h` extension. Applications can use the information in the map file to link users to the appropriate context-sensitive help topic.

Note: Some developers may use the term header file instead of map file.
There are some variations in the way context-sensitivity works depending on which supported ePublisher output format you use. For example, Microsoft HTML Help, Sun JavaHelp, and Oracle Help use map files.

> **Note:** WebWorks Help, WebWorks Reverb, Dynamic HTML Help, and XML+XSL do not use map files.

When an application calls a context-sensitive help topic, it relies on the topic IDs and map numbers to identify the specific topic to display. Therefore, the topic IDs and map numbers must be embedded both in the application code and in the help system. If the topic IDs and map numbers do not match, the wrong topic (or no topic) displays when the user requests Help.

Following is a typical example of a Microsoft HTML Help map file:

```
#define IDH_WDWTYPE     1001
#define IDH_WDWENTER    1002
#define IDH_WDWCANCEL   1003
```

In this example, IDH_WDWTYPE is a topic ID, and 1001 is the corresponding map number. These topic IDs and map numbers must be embedded in the software application and in your source documents.

Following is a typical example of a Sun JavaHelp and Oracle Help map file:

```
<mapID target="ch1_htm_999374" url="ch1.htm#999374">
<mapID target="ch2_htm_999640" url="ch2.htm#999640">
<mapID target="ch9_htm_999786" url="ch9.htm#999786">
```

In this example, ch1_htm_99374 is a topic ID, and ch1.htm#99374 is the target URL for the topic ID. These topic IDs must be embedded in the software application and in your source documents.

Planning for Context-Sensitive Help

Creating context-sensitive help requires you to collaborate with application developers. Because topic IDs and map numbers must be embedded in both the software application and in your source documents, you and the application developers must agree in advance on the values to use.

Before you create context-sensitive help topics, complete the following steps:

1. Confirm with your application developers that the application supports context-sensitive help.
2. Meet with your application developers to identify each context-sensitive help topic you need to create.
3. Determine if you will use topic IDs or file names to create links to context-sensitive help topics.
4. Discuss the process for referencing context-sensitive help topics from the application with your application developers. Writers and application developers have the following options for creating context-sensitive help links:

    - The writer chooses the topic IDs or file names and embeds them in the source documents. If the generated output supports map files, the writer performs the following steps:
        - The writer uses topic IDs inserted into source documents and ePublisher to generate a map file, also known as a header file, that contains the topic IDs defined by the writer and automatically generated mapping IDs.
        - The writer supplies the generated map file to the application developers to implement.

    Note: The writer must supply the header file along with the help system to the developers each time the writer updates the help system. This ensures correctly identified context-sensitive help topics each time.

    - Application developers choose the topic IDs or file names and then give the topic IDs or filenames to the writer to embed in the source documents. If the generated output supports map files, the application developers perform the following steps:
        - Application developers create the map file, or header file.
        - Application developers give the writer a copy of the map file, or header file, and the writer embeds the topic IDs from the map file into the source documents.

Note: The group context must be unique so that if there are the same topic ID's in a help system, the context sensitive pointer will go to the correct place in the help.

Topic ID and File Name Requirements

If you are creating context-sensitive help topics using topic IDs, topic IDs must follow these guidelines:

- Must be unique
- Must begin with an alphabetical character
- May contain alphanumeric characters
- May not contain special characters or spaces, with the exception of underscores (_)

Dynamic HTML, Eclipse Help, Microsoft HTML Help, Oracle Help, Sun, WebWorks Help, and WebWorks Reverb support the use of topic IDs to create context-sensitive help.

If you are creating context-sensitive help topics using file names, file names must follow these guidelines:

- Must be unique
- Must begin with an alphabetical character
- May contain alphanumeric characters
- May not contain special characters or spaces

XML+XSL support the use of file names to create context-sensitive help.

Output Formats that support Creating Context-Sensitive Help Links In DITA Source Documents

The following table lists available for creating context-sensitive help links.

that Support Context-Sensitive Help Links
You can use ePublisher to create context-sensitive help links for the following : • Eclipse Help • Microsoft HTML Help • Oracle Help • Sun JavaHelp 2.0 • WebWorks Help • WebWorks Reverb • WebWorks Reverb 2.0

Specifying Context-Sensitive Help Links in DITA Source Documents

Before you specify context-sensitive help links, review context-sensitive help link requirements. For more information about context-sensitive help and context-sensitive help link requirements, see “Context-Sensitive Help” and “Output Formats that support Creating Context-Sensitive Help Links In DITA Source Documents”.

To specify a topic alias for a topic in a DITA source document:

1. In your DITA source document, locate your topic’s meta information container element.
2. For this element, you use the <othermeta> element to define the topic alias, for example:
   `<othermeta name="TopicAlias" content="helpid"/>`

   Note: You can also use the <data> element to define topic alias values, for example: `<data name="TopicAlias" value="helpid"/>`
3. Save the DITA document
4. Generate output for your target. For more information, see “Generating Output”

Steps for creating context-sensitive help links in DITA may be different in other versions of DITA.

Creating Hyperlinks in DITA Source Documents

You will have to use DITA elements to create links in your documents so that ePublisher can create links to other topics or files in the online help output of your choice.

To create an a hyperlink to a topic in the same DITA file or another DITA file, complete the following steps

1. Identify your link type, as it will determine the markup that is required.
2. For a hyperlink that is going the same file create the following markup:

   ```
   <xref href="#yourtopicID">Your xref link text</xref>
   ```

3. For a hyperlink that is going to a different file create the following markup:

   ```
   <xref href="file.xml#yourtopicID">Your xref link text</xref>
   ```

4. For a hyperlink that is going to a PDF create the following markup:

   ```
   <xref format="PDF" href="sample.PDF">Your xref link text</xref>
   ```

Creating Popups in DITA Source Documents

This section explains how to create popups in DITA source documents.

Popups

A **popup window** is a window that is smaller than standard windows and typically does not contain some of the standard window features such as tool bars or status bars. Popup windows display when users hover over or click on a link. The popup topic closes automatically as soon as the users clicks somewhere else.

A typical use of popups is to display glossary terms. For example, in a printed document, terms and definitions are typically grouped in a separate glossary document. However, in a help system, you can display glossary definitions in popups. When you create glossary popups, users can choose whether they want to view the definition of an unfamiliar term. If they want additional information about the term, they can view the definition in a click.

You create popups by creating link between the word or phrase in a topic and the content you want to display in the popup, and then you use <othermeta> elements or paragraph styles to create popups.

Popup and Popup Append paragraph styles

Specifies that content displays both in popup windows and in standard help topics. You apply the Popup paragraph style to the first paragraph of content you want displayed in the popup window. If you have more than one paragraph of content you want to display, you apply the Popup Append style to the additional paragraphs.

For example, if you apply a glossary term and glossary definitions style for a glossary using the Popup and Popup Append styles, the terms and definitions in your output display in both a popup window and in a glossary topic that contains the definitions.

Popup Only and Popup Only Append paragraph styles

Specifies that content displays only in popup windows. You apply the Popup Only paragraph style to the first paragraph of content you want displayed in the popup window. If you have more than one paragraph of content you want to display, you apply the Popup Only Append style to the additional paragraphs.

For example, if you apply a glossary term and glossary definition style for a glossary using the Popup Only and Popup Only Append paragraph style, the terms and definitions in your output display in only popup windows. The content is not displayed in an additional glossary topic that contains the definitions.

Requirements for Creating Popups in DITA Source Documents

You prepare popups using <othermeta> elements or paragraph formats. Before you create popups in your source documents, verify that your , templates, and stationery meet popup requirements. The following table lists requirements for creating popups.

	Requirement
Output Format	You can use ePublisher to create popups for the following : • Microsoft HTML Help • Oracle Help • Sun JavaHelp • WebWorks Help

Creating Popup Links in DITA Source Documents

Steps for creating links in DITA may be different in other versions of DITA.

Using Paragraph Styles to Create Popups in DITA Source Documents

You can use Popup paragraph formats in your DITA source documents to create popups. To use Popup paragraph styles to create popup windows, your Stationery and DITA file must have the following items configured:

- Popup and Popup Append paragraph style behaviors if you want your content to display both in popup windows and in standard help topics.
- Popup Only and Popup Only Append paragraph style behaviors if you want your content to display only in popup windows.

The following procedure provides an example of how to use Popup paragraph formats to create popup windows in DITA source documents.

To create popup windows using Popup paragraph styles in DITA source document

1. In your DITA source document, create a link between a word or phrase in the topic and the content you want to display in the popup window and ensure that the link resolves in the document.
2. Save your DITA source document.
3. In the ePublisher **Style Designer**, configure the destination paragraph styles with the appropriate popup behavior via the **Options** panel.
4. Generate output for your project.
5. In Output Explorer, go to the page where you created the popup window and verify that ePublisher created the popup window and that the popup window displays the content you specified.

Creating Related Topics in DITA Source Documents

This section explains how to create related topics in DITA source documents.

Related Topics

Related topics provide a list of other topics that may be of interest to the user viewing the current topic. For example, you could have a section called Creating Web Pages in your help. You may also have many other topics, such as HTML Tags and Cascading Style Sheets, that related to creating Web pages. Identifying these related topics for users can help them find the information they need and identify additional topics to consider. However, providing these types of links as cross-references within the content itself may not be the most efficient way to present the information. By utilizing related topics links, you combine the capabilities of cross-references with the efficiency of a related topics button.

Related topics and See Also links provide similar capabilities, but there are several important differences:

- Related topics can link to headings in a Help system that do not start a new page.
- Relate topics links are static and defined in the source documents as links. You must have all the source documents to create the link and generate the output.
- If a related topics list contains a broken link in the source document, that link is broken in the generated output. In a See Also link list, the broken link is not included in the output.

The stationery designer can configure related topics to display in the following ways:

- Display in a popup window when the user clicks a button,
- Included in a list in the topic itself and then displayed in a popup window when the user clicks a button.

Note: If a related topic link is broken in the source document, in most cases that link is broken in the generated output. WebWorks Help provides an additional feature by removing broken links from related topics lists that are displayed in a popups window when a user clicks the Related Topics button.

Requirements for Creating Related Topics Links in DITA Source Documents

You can create related topics using a paragraph format. Before you create related topics links in your DITA source documents, verify that your , templates, and stationery meet related topics links requirements. The following table lists requirements for creating related topics links.

	Requirement
Output Format	You can use ePublisher to create related topics for the following : • Eclipse Help • Microsoft HTML Help • Oracle Help • Sun Java Help • WebWorks Help • WebWorks Reverb • WebWorks Reverb 2.0

Specifying Related Topics Links in DITA Source Documents

Create related topics links by applying the Related Topics paragraph format to cross-references you create in your DITA source documents. Before you create related topics in your source documents, review related topics links requirements. For more information about related topics and related topics links requirements, see “Related Topics” and “Requirements for Creating Related Topics Links in DITA Source Documents”.

The following procedure provides an example of how to create related topics links in DITA source documents using DITA 1.4. Steps for creating related topics links in DITA may be different in other versions of DITA.

To create a related topics list in a DITA source document

1. Identify the topic in which you would like to insert a related topics list.
2. Identify the different topics you want to link to from this topic.

   Note: Generally, you should only create one related topics list for each section of your source document that corresponds to a help topic. For example, if you have specified in your ePublisher project that there will be a page break at each Heading 1 section, then you should only create one related topics list for each Heading 1 section within your source document.

Creating See Also Links in DITA Source Documents

This section explains how to create See Also links in DITA source documents.

See Also Links

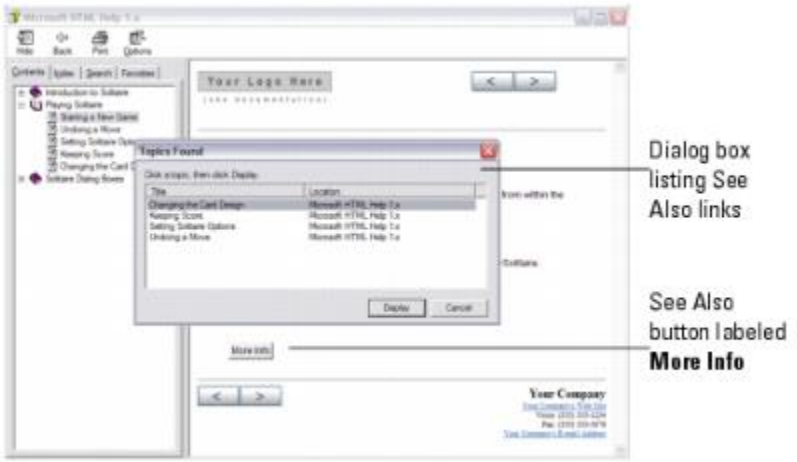
See Also links, also known as **ALinks**, or **associative links**, are links that may be of interest to the user viewing the current topic. These links use internal identifiers to specify the links and the link list is built dynamically based on the topics available when the user clicks to display the links. See Also links are important to use with larger help sets and merged help sets.

Related topics and See Also links provide similar capabilities, but there are several important differences:

- See Also links must link to styles that start a new topic, such as a heading.
- See Also links are dynamic and the lists of links are built at display time instead of during help generation.
- Since see Also link lists are dynamically built, they do not include links to topics that are not available when the user displays the links. If a er list contains a broken link in the source document, that link is broken in the generated output for most .

See Also links are useful if you plan to merge help systems. For example, if you have a multiple help systems that you merge into one main help system at run time and if your topics in the merged help systems contain See Also keywords that are also used in the main help system, links to those topics are included in the See Also lists in the main project.

You can create See Also links as buttons or as inline text links in Microsoft HTML Help and WebWorks Help. The following example shows how the two different types of See Also links display in a Microsoft HTML Help system.



To create See Also links in your generated output, use a See Also paragraph format or character format defined by the stationery designer and through <othermeta> elements.

Requirements for Creating See Also Links in DITA Source Documents

You create See Also links using a paragraph or character format and through <othermeta> elements. Before you create See Also links in your DITA source documents, verify that your , templates, and stationery meet See Also link requirements. The following table lists requirements for creating See Also links.

	Requirement
Output Format	You can use ePublisher to create See Also links for the following : • Microsoft HTML Help • WebWorks Help

Specifying See Also Links in DITA Source Documents

Create See Also links by applying the See Also paragraph format or character format to text in your DITA source documents and through <othermeta> elements into your DITA source documents. Before you create See Also links in your source documents, review See Also link requirements. For more information about See Also links and See Also link requirements, see “See Also Links” and “Requirements for Creating See Also Links in DITA Source Documents”.

Steps for creating See Also links in DITA may be different in other versions of DITA.

Using the data element

This section explains how to use the <data> element in DITA source documents. The data element allows insertion of arbitrary marker style data similar to the <othermeta> element. Unlike the <othermeta> element, the <data> element can be used throughout a topic, just like a marker.

Using the data element to insert markers

The <data> element represents a property within a DITA topic or map. You can use the <data> element to insert a marker type anywhere on your page. Here is an example of the <data> element: `<data name =”TOCIcon” value = “red.png”/>`. In this example, our marker name is TOCIcon, and is using the value of red.png.

For more information on the <data> element see https://docs.oasis-open.org/dita/v1.2/os/spec/langref/data.html

Assigning Custom Page Styles to Pages in DITA Source Documents

This section explains how to assign custom page styles to specific pages in DITA source documents.

Page Styles

By default, each page generated by ePublisher is associated with the default page style defined in the stationery used by your ePublisher project. This means that typically you do not need to specify a page style for pages when you generate output.

For example, you may want to use one page style in your help system for all concept and procedure topic pages, and another page style for all context-sensitive window description topic pages in your help system. In this example, you can use the default page style for all of your concept and procedure topic pages, and then you can use a second custom page style defined in your stationery for all context-sensitive window description topic pages in your help system.

Requirements for Specifying Custom Page Styles for Pages in DITA Source Documents

You specify page styles for pages using <othermeta> elements. Before you specify page styles for pages by using <othermeta> elements in your DITA source documents, verify that your , templates, and stationery meet image style requirements. The following table lists requirements for specifying page styles for pages.

	Requirement
Output Format	You can use ePublisher to specify page styles for specific images for the following : • Dynamic HTML • Eclipse Help • eBook - ePub 2.0 • Microsoft HTML Help • Oracle Help • Sun JavaHelp • WebWorks Help • WebWorks Reverb • WebWorks Reverb 2.0

Specifying Custom Page Styles for Pages in DITA Source Documents

For more information about page styles and page style requirements, see “Page Styles” and “Requirements for Specifying Custom Page Styles for Pages in DITA Source Documents”.

To specify a Page Style for a topic in a DITA source document:

1. In your DITA source document, locate your topic’s meta information container element.
2. For this element, you use the <othermeta> element to define the Page Style, for example: <othermeta name="PageStyle" content="stylename"/>
3. Save the DITA document
4. Generate output for your target. For more information, see “Generating Output”

Steps for specifying custom page styles for pages in DITA may be different in other versions of DITA.

Using Custom Graphic Styles for Images in DITA Source Documents

This section explains how to use custom graphic styles for images in DITA source documents. It also explains how graphic styles are assigned by default using implicit image properties in the source content.

Assigning Graphic Styles

By default, each image generated by ePublisher is associated with one of several possible graphic styles that all begin with **Default**. This means that typically you do not need to specify a graphic style for images when you generate output.

For example, you may want to create a special style called “thumbnail” for handling very large images. In this example, each graphic that you assign the “thumbnail” style would have those defined properties. All other graphics would use one of the **Default** graphic styles.

To use a custom graphic style for images in your DITA source documents:

1. In your DITA source document, locate the image/s that will be assigned the custom graphic style.

   Example: `<image href="myimage.png" />`

2. Within this image element, insert the `outputclass` attribute to assign the custom graphic style to the image. Assign the `outputclass` value as the name of the custom graphic style.

   Example:

   ```
   <image href="myimage.png" outputclass="CustomGraphicStyle" />
   ```

3. Define the graphic style in ePublisher Designer.

Default Graphic Styles for DITA

For DITA, ePublisher will automatically use predefined graphic styles based on the properties of the graphic that is being generated. Below are the list of predefined graphic styles that ePublisher uses for DITA images when no custom graphic style has been assigned.

Graphic Style Name	Example DITA Code to Produce

Default	`<image href="file.png"/>`
	
	`<image placement="break" href="file.png"/>`

DefaultLeft	`<image placement="break" align="left" href="file.png"/>`
DefaultCenter	`<image placement="break" align="center" href="file.png"/>`
DefaultRight	`<image placement="break" align="right" href="file.png"/>`
Default Scalefit	`<image scalefit="yes" href="file.png"/>`

Customizing TOC Entry in DITA

Use these steps to customize a TOC entry in your **Reverb 2.0** output. Your DITA file must have a nested heading structure for TOC Icons to appear.

1. Locate the `<prolog>` element in your DITA document. If your document does not have a `<prolog>` element, you can insert one near the top of the document under the opening `<concept>`, `<task>`, or `<topic>` element. A `<metadata>` element must be nested within the `<prolog>` element. An `<othermeta>` element must be nested withing the `<metadata>` element. The end result should look like this:

```
<prolog>
    <metadata>
      <othermeta name="TOCEntryClass" content="folder_icon" />
    </metadata>
  </prolog>
```

2. The <othermeta> tag requires a name and content attribute. The value of the name attribute will be "TOCEntryClass". The value of the content attribute can be whatever value you would like to use. The value of the content attribute will become the name off the CSS class that you will customize in an override of the `*.scss` files.
3. Save your DITA document.
4. Scan the document in ePublisher Designer.
5. Open the **Style Designer**.
6. Open **Marker Styles**.
7. Locate the **Marker Type Option** from the **Options** tab and set its value to `TOC Entry Class`.
8. In this example, the assigned class for the Menu TOC entry will be the value of the marker: `folder_icon`.
9. Add the following to a target override of `_icons.scss`. Notice how the CSS class is the name of the value given in the Marker Text Window. In this example we change the icon color and the icon of the TOC entry. You are able to make other customizations such as adding a border, or changing the background color.

```
.folder_icon {

 > div > span > i {

color: black;

&:before {

      content: $folder_icon;

   }
```

```
    }

        }
```

10. Save your project and generate the output.

Customizing Table of Contents Icons for Topics in DITA Source Documents Using Legacy Outputs

This section explains how to customize the appearance of table of contents icons for topics in Microsoft HTML Help, Sun JavaHelp, Oracle Help, and WebWorks help systems. For information on how to customize Menu TOC entries in Reverb 2.0, see “Customizing TOC Entry in DITA”.

Requirements for Specifying Custom Table of Contents Icons in DITA Source Documents

	Requirement
Output Format	You can use ePublisher to specify custom table of contents icons in the following : • Microsoft HTML Help • Oracle Help • Sun JavaHelp • WebWorks Help

Specifying Custom Table of Contents Icons in DITA Source Documents

For more information about custom table of contents icons, see “Requirements for Specifying Custom Table of Contents Icons in DITA Source Documents”.

To specify a Custom Table of Contents Icon for a topic in a DITA source document:

1. In your DITA source document, locate your topic’s meta information container element.
2. For this element, you use the <othermeta> element to define the Page Style, for example: `<othermeta name="TOCIcon" content="blue.png"/>`.

   Note: You can also use the <data> element, for example: `<data name="TOCIcon" value="blue.png"/>`.
3. Save the DITA document
4. Generate output for your target. For more information, see “Generating Output”

Steps for customizing table of contents icons for topics in DITA may be different in other versions of DITA.

To specify a custom table of contents icon in a DITA source document

5. ***If you want to specify a custom table of contents icon for Microsoft HTML Help***, identify the number of the image you want to use for the table of contents image for the topic in the `.hhp` file for your Microsoft HTML Help project by completing the following steps:
   a. On the **View** menu, click **Output Directory**.
   b. Open the `ProjectName` folder, where *`ProjectName`* is the name of your project.
   c. Open the `ProjectName.hhp` file where *`ProjectName`* is the name of your project.
   d. On the **Contents** tab, select a table of contents entry, and then click the **Pencil** icon.
   e. On the **Advanced** tab, in the **Image index** field, use the up and down arrows to identify the table of contents image you want to use for the topic.
   f. Note the number of the image you want to use for the table of contents image for the topic.

      For example, if you want to use a question mark icon with a red star for the table of contents icon for new topics, note that the number for this icon is 10.
   g. Close HTML Help Workshop.
6. ***If you want to specify a custom table of contents icon for Oracle Help or Sun JavaHelp***, create the graphic file for the custom table of contents icon in `.gif` format. The default graphics used as Sun JavaHelp or Oracle Help table of contents icons are 17 x 17 pixels. The custom graphics you create for Sun JavaHelp or Oracle Help table of contents icons should also be 17 x 17 pixels. You can assign any name to the graphic files.

7. ***If you want to specify a custom table of content icon for WebWorks help***, create graphics files containing the collapsed and expanded versions of the icons you want to use, then save the graphic files in `.gif` format. The default graphics used as WebWorks Help table of contents icons are 17 x 17 pixels. The custom graphics you create for WebWorks Help table of contents icons should also be 17 x 17 pixels. You can assign any name to the graphic files.

8. Copy the graphic files you want to use as icons in the table of contents into the following folder:

   Note: If the folder does not exist, first create the folder using the specified folder structure and then copy the graphic files you want to use as icons into the folder. You do not need to perform this step when specifying custom table of contents icons for Microsoft HTML Help.

   - ***If you are generating Oracle Help***, copy the graphic files you want to use into the following folder:

     ProjectName`\Formats\Oracle Help\Files\images` folder, where *ProjectName* is the name of your project.

   - ***If you are generating Sun JavaHelp 1.1.3***, copy the graphic files you want to use into the following folder:

     ProjectName`\Formats\Sun Java Help 1.1.3\Files\images` folder, where *ProjectName* is the name of your project.

   - ***If you are generating Sun JavaHelp 2.0***, copy the graphic files you want to use into the following folder:

     ProjectName`\Formats\Sun Java Help 2.0\Files\images` folder, where *ProjectName* is the name of your project.

   - ***If you are generating WebWorks Help***, in your `ProjectName\Files` folder, where *ProjectName* is the name of your project, create a `wwhelp\images` subfolder and copy the graphic files you want to use into this folder. Your project file structure should be similar to the following structure:

     ProjectName`\Files\wwhelp\images`

Using markopen and markclose

This `default.wwconfig` modification specifies the start and a stop of an element. This is especially useful for the implementation of the dropdown elements in WebWorks Help 5 or WebWorks Reverb output. ePublisher has preconfigured items such as the Defintion Term Elements, below is an example of the sample DITA markup:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE concept PUBLIC "-//OASIS//DTD DITA Concept//EN" "concept.dtd">
<concept id="simplelist" xml:lang="en-us">
  <title>Definition List</title>
  <conbody>
    <p>Definition list</p>
    <dl>
     <dlhead>
      <dthd>Image File View Selection</dthd>
      <ddhd>Resulting Information</ddhd>
     </dlhead>
     <dlentry>
      <dt>File Type</dt>
      <dd>Image's file extension</dd>
     </dlentry>
     <dlentry>
      <dt>Image Class</dt>
      <dd>Image is raster, vector, metafile or 3D</dd>
     </dlentry>
     <dlentry>
      <dt>Number of pages</dt>
      <dd>Number of pages in the image</dd>
     </dlentry>
     <dlentry>
      <dt>Fonts</dt>
      <dd>Names of the fonts contained within a vector image</dd>
     </dlentry>
    </dl>
    <p>Content after the definition list.</p>
  </conbody>
</concept>
```

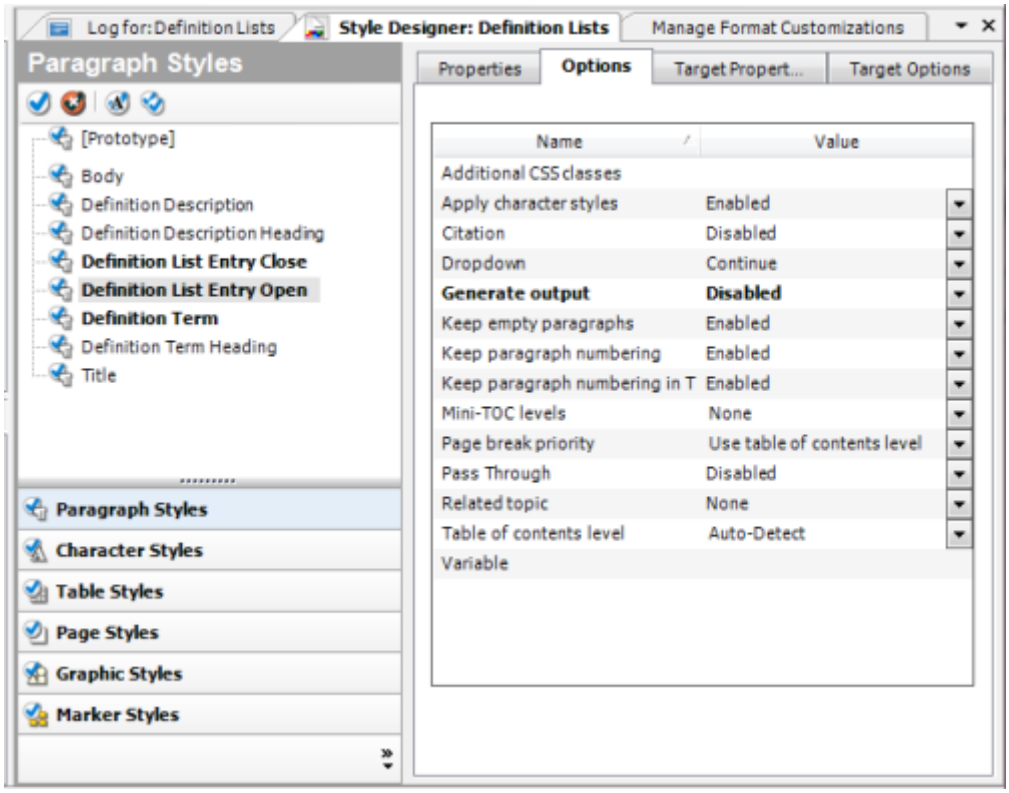
Without the use of the `markclose` in the default.wwconfig, the last paragraph would be underneath the last Definition Term element. To make sure that no other elements are inside the dropdown, you can assign markclosed to it by creating an override to `default.wwconfig` located in `[project`

`directory]\Formats\Adapters\xml\scripts\dita`. For more information, refer to “Creating Format Overrides” Below is an example of the modification needed to get the desired output:

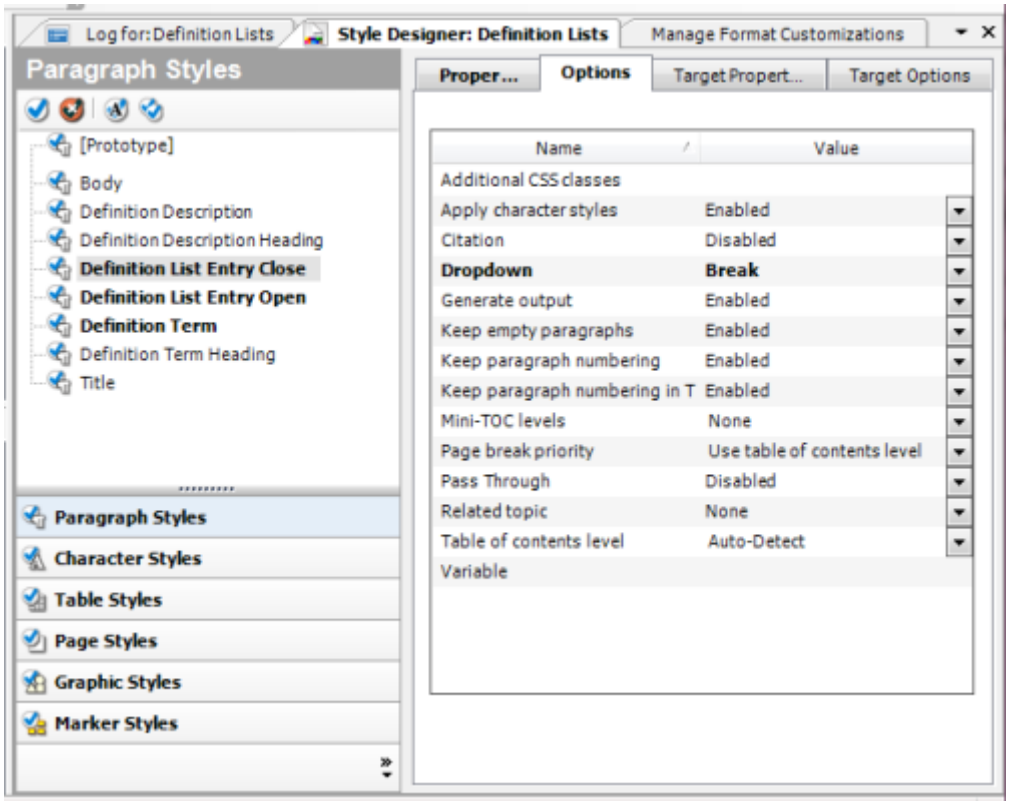
```
  <!-- Mark open/close on definition entries -->
  <!--                                        -->
  <Style match="//*[contains(@class, ' topic/dlentry ')]">
   <xsl:variable name="VarName" select="'Definition List Entry'" />
   <wwditaconfig:Head name="{$VarName}" markopen="{$VarName} Open"
 markclose="{$VarName} Close" />
 </Style>
```

Configuring markopen and markclose entries for dropdowns in ePublisher

After you have created the override, you will still need to scan for the newly created paragraph styles in ePublisher. Using our created override, you would see something like this in the Style Designer:



The output is set to disabled because the markopen just serves as a placeholder text. In the closed entry, we set the Options to have the Dropdown to Break:



We do this because that is how the content will emit, below is an example of the sample output's HTML that shows the paragraph styles as div classes:

```
20       <div class="Definition_List_Entry_Close"><a name="3_2_2_4_8c"> </a></div>
21       <div class="Definition_Term" onclick="WebWorks_ToggleDIV(WebWorksRootPath,
&quot;wwdd3_2_2_4_10_2&quot;);"><a name="3_2_2_4_10_2">Fonts</a><script
type="text/javascript" language="JavaScript1.2">WebWorks_WriteArrow(WebWorksRootPath,
"wwdd3_2_2_4_10_2", false);</script> <a
href="javascript:WebWorks_ToggleDIV('wwdd3_2_2_4_10_2');"><img
id="wwdd3_2_2_4_10_2_arrow" src="images/expanded.gif" alt="" border="0"></a></div>
22       <script type="text/javascript"
language="JavaScript1.2">WebWorks_WriteDIVOpen("wwdd3_2_2_4_10_2", false);
</script><div id="wwdd3_2_2_4_10_2" style="visibility: visible; display: block;">
23       <div class="Definition_Description"><a name="3_2_2_4_10_4">Names of the fonts
contained within a vector image</a></div>
24       <script type="text/javascript" language="JavaScript1.2">WebWorks_WriteDIVClose();
</script></div>
25       <div class="Definition_List_Entry_Close"><a name="3_2_2_4_10c"> </a></div>
26       <div class="Body"><a name="3_2_2_6">Content after the definition list.</a></div>
27     </blockquote>
```

Note that the Content after the definition list is now in a Body paragraph, so that it does not become included in the dropdown.

Troubleshooting DITA issues

Occasionally there might be issues with the source documents you are using. Below is a list linking to the wiki solutions website that will help you troubleshoot each one:

Issue	Solution
If you are having issues with whitespace when authoring with FrameMaker using DITA	White space in FrameMaker
If you are having issues with cross references when authoring with FrameMaker using DITA. In particular, if you are using the fm-xref element, generated output will not properly update the generated numbering. There is a solution to fix this behavior in FrameMaker.	DITA Cross References in FrameMaker
If you are receiving a Cannot Duplicate Document	Cannot duplicate document error message
If you are having issues outputting cross references from DITA	Cross Reference to same topic not working
If you are having issues using a custom DTD	DTD Issues
If you are using an older or newer version of the OTK	Different versions of the OTK
If you are trying to use conditional text	DITA conditions
If you are trying to localize figure/ table paragraphs	Localization of the word "table" or "figure"
If you are getting a Java error when generating	Java Error
If you are trying to put a list in a table	List in a Table
If you are getting a Resample WIFerror upon generation	ResampleWIF Pipeline Error Blocks Generation of Output

Issue	Solution
If your elements are not being populated in the style designer	DITA styles are not being read in the Style Designer
If your Overview topics are showing up on the same level as their children	DITA Overview Topics Showing on Same Level as Children Topics