Don't (XML) Mention It: Tagging XML Content in DITA 1.3

Contents

Part I

Topics:

Don't (XML) Mention It: Tagging XML Content in DITA 1.3

Introduction to the XML Mention Domain

A minor frustration with DITA was an inability to easily describe XML content. If you have ever found yourself using the angle bracket character entities and wishing that there was another, more elegant way to describe your XML code, then look no further than the XML Mention domain of elements available in DITA 1.3. This domain now allows content authors to mark up XML content for XML elements and attributes, numeric character references, text entities, parameter entities, processing instructions and namespaces. Two key advantages of this approach is that you no longer have to "escape" inline XML references and you have the means to more effectively search for content that is semantically described by the XML Mention elements. So no more looking through every angle bracket reference, percentage symbol, ampersand or other typical characters used in formatting XML content in the hope that you will eventually find what you are looking for. This article focuses on how the DITA 1.3 XML Mention elements can be used.

Tagging Tags with `<xmlelement>` and Attributes with `<xmlatt>`

If you have ever written an internal DITA style guide detailing how to use various DITA elements and attributes, you will definitely appreciate the new `<xmlelement>` and `<xmlatt>` elements.

For example, let's say you are writing an internal DITA style guide, and want to indicate that your writers should avoid using the non-semantic bold and italic tags in their DITA content. Instead of having to use angle bracket entity characters to describe what you mean, you can use `<xmlelement>` instead. For example:

```
<p>Please avoid using the <xmlelement>b</xmlelement> (bold) and
 <xmlelement>i</xmlelement> (italic) tags in your content. Here at ACME we use
 semantic markup whenever possible.</p>
```

Similarly, if you want to indicate in your DITA style guide which note attributes to use and not to use, you can indicate the names of the attributes using `<xmlatt>`:

```
<p>When using the <xmlelement>note</xmlelement> element in ACME documentation,
 use the following <xmlatt>type</xmlatt> values: <q>caution</q> to inform
 users when they need to take care before proceeding with a task, <q>warning</
q> to indicate a potentially hazardous situation for the user, and <q>danger</
q> for scenarios that might involve injury or death. ACME documentation also
 uses the <q>tip</q> and <q>important</q> attributes for <xmlelement>note</
xmlelement>.</p>
```

Referencing Numeric Characters and Text Entities with `<numcharref>` and `<textentity>`

Prior to DITA 1.3, if you wanted to insert a numeric character reference into your document, you had to "escape" the leading ampersand so that it didn't actually render the character itself. If, for example, you wanted to reference the Greek capital letter sigma (used as a summation operator in mathematics) inline in your content, you had to write it as &amp;#931; (decimal base) or &amp;#x3A3; (hexadecimal base) so that it would render as "&#931;" and "&#x3A3;" respectively, and not as the character ("Σ") itself. The new `<numcharref>` element in DITA 1.3 makes this process easier, eliminating the need for explicitly referencing the leading ampersand and octothorpe characters. The following sample code demonstrates how `<numcharref>` can be used:

```
<p>To include the summation operator symbol, simply type <numcharref>931</
numcharref> or <numcharref>x3A3</numcharref> within the text field.</p>
```

Similarly, the new `<textentity>` element in DITA 1.3 allows you to describe character entities the same way. By default, the DITA Open Toolkit formats a leading ampersand and a trailing semi-colon to the character entity it encloses, just as with `<numcharref>`. The following example shows how this element can be used:

```
<p>Thanks to DITA 1.3, you no longer have to wrap the lesser-than
 (<textentity>lt</textentity>) or greater-than (<textentity>gt</textentity>)
 character entities around elements to describe them in your DITA content, you
 can simply use <xmlelement>xmlelement</xmlelement> instead.</p>
```

Describing Parameter Entities with `<parameterentity>`

Parameter entities are used to define a collection of elements, attributes and their values in XML. If you have ever looked at a DTD, you have seen parameter entities. As an example, the parameter used to define the paragraph element in DITA is called "`%p.content;`". If you are trying to explain aspects of a DTD, you may have a need for the new `<parameterentity>` element. This element wraps a leading percentage symbol and trails with a semi-colon around the parameter entity. The following is an example of how it can be used in practice:

```
<p>For example, the parameter used to define the paragraph element in DITA is
 called <parameterentity>p.content</parameterentity>.</p>
```

Referencing Processing Instructions using the `<xmlpi>` element

In XML, processing instructions are used to carry information on how a processor should work with a defined instruction. An XML processing instruction typically contains a name followed by an optional value. While processing instructions are not part of the DITA standard, some authoring tools enable their use within DITA. Additionally, processing instructions are used in other types of XML, such as DocBook or XSL. An example processing instruction in XSL looks like this: `xml-stylesheet type="text/xsl" href="style.xsl"`, where "xml-stylesheet" is the name and the remainder is a value that describes how the processor should interpret and work with it. To describe processing instructions in DITA 1.3, use the `<xmlpi>` element, which wraps a "<?" to the beginning of the processing instruction it describes, and a "?>" at its end. The following example shows how it can be used:

```
<p>In DocBook processing instructions that begin with <q>dbfo</q> can be
 used for formatting operations. An example of how this can be used to
 set the width for a two-and-a-half inch sidebar would look like this:
 <xmlpi>dbfo_sidebar-width="2.5in"</xmlpi>.</p>
```

Using `<xmlnsname>` to Describe Namespaces

In XML a namespace is a mechanism designed to avoid clashes between elements or attributes from more than one XML vocabulary. For example, HTML and DocBook both have a "table" element just like DITA does, and if you wanted to be clear as to which table element you are referencing when you are using more than one XML instance of table, you would do so by declaring the namespace that this table belongs to. The following example shows how it might be used in practice:

```
<p>In MathML, the <xmlement>m:math</xmlelement> is the root element. The "m:"
 prefix is bound to the MathML 2.0 namespace: <xmlnsname>http://www.w3.org/
TR/2003/REC-MathML2-20031021/</xmlnsname>.</p>
```

Unlike all of the other elements discussed in this article, there is no expectation that processing will add any characters at the beginning and the end of the enclosed text. But, of course it can be formatted at output in order to make it appear distinct from other content in any way you might want.

When and How to Use `<markupname>`

While not part of the XML Mention domain, it’s worth including the `<markupname>` element here, as all of the other elements within the XML Mention are specialized from it. In general, it is recommended that the other, more specific XML Mention elements be used whenever possible, or that `<markupname>` be used for describing XML markup that is not covered by the other elements in that domain. In environments where the XML Mention domain is constrained away for whatever reason, `<markupname>` can act as a generic descriptor for XML markup. Here’s an example of it in practice:

```
<p>In commonElementsMod.rng, the <markupname>basic.ph</markupname> pattern
 provides a choice of several inline elements.</p>
```

There is no guidance in the DITA 1.3 specification as to how `<markupname>` should be formatted at output. It semantically describes XML markup but does not prescribe how it should appear.

Conclusion

The addition of the XML Mention domain makes it much easier for content creators to mark up XML within their documentation without having to resort to using awkward entities.

For more information on the XML Mention domain or anything else relating to DITA 1.3, head to the OASIS website *for the official specification*.

Thanks goes out to Jason Owen, Leigh White and Bob Thomas for their editorial assistance with this article.