XMLmind XML Editor - Support of MathML 2

Hussein Shafie, Pixware <xmleditor-support@xmlmind.com>

September 29, 2011

Requires XMLmind XML Editor Professional Edition.

1. Introduction

XMLmind XML Editor Professional Edition natively supports MathML 2 *presentation markup*[1] by the means of an add-on. If you are interested in adding equations to your XML documents, you need to use Options → Install Add-ons and select the add-on called "MathML support".

This MathML add-on:

- allows to create standalone MathML documents,
- tightly integrates MathML presentation markup with DocBook 5,
- tightly integrates MathML presentation markup with DITA topics.

How, in practice, to add equations to your XML documents is described in How to add equations to your XML documents [1]. The precise contents of the add-on is detailed in Appendix B, *Contents of the "MathML support" add-on* [8].

Native support means that MathML elements are treated like any other element. That is, XMLmind XML Editor does *not* embed a specialized math editor. On the contrary, all commands, all kinds of selection, work fine with MathML elements. However, using the generic commands requires you to learn MathML. If you don't want to do that, for example, because you just want to type a couple of equations, then simply use the MathML tool which is installed by the add-on just below the Edit tool, at the top/right of the main window. Using the MathML tool is explained in Section 3, “Using the MathML tool” [2].

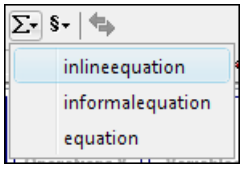
2. How to add equations to your XML documents

- If you are authoring DocBook 5 documents, insert an `equation`, `informalequation` or `inlineequation` containing a MathML `mml:math` child element into your document, then edit the contents of this `mml:math` child element normally, using the Edit tool and/or the MathML tool (see below [2]).

  In order to insert insert an `equation`, `informalequation` or `inlineequation` containing a MathML `mml:math` child element into your DocBook 5 document,

[1] The content markup is not supported at all. Content elements are rendered as tree views.

- Use the Edit tool and choose one of the `equation(mathml)`, `informalequation(mathml)` or `inlineequation(mathml)` element templates.
- OR use the "Add MathML Equation" found in the DocBook tool bar.



- Same approach if you are authoring DITA topics.
- For all document types other than DocBook 5 and DITA topics (DocBook 4, XHTML, etc), you must treat MathML as if it were a graphics format such as SVG or PNG.

  Unlike what happens for DocBook 5 documents and DITA topics, you'll not be able to directly edit the MathML elements, but if you install this MathML add-on (or the add-on called "JEuclid image toolkit plug-in" — see 4 [9]), the MathML elements will be properly rendered on screen and also properly converted to formats such as HTML, PDF, RTF, etc.

  In practice, this means:

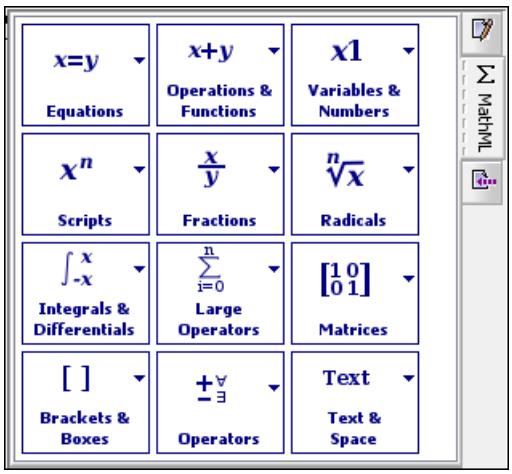
  a. Create a standalone MathML document using File → New, MathML, "Inline Math" or "Math Block".

  b. Reference this MathML document in the proper ``image element":

DocBook 4 example	`<imagedata fileref="equations/equation1.mml"/>`
XHTML example	`<img src="equations/equation1.mml"/>`

3. Using the MathML tool

Figure 1. The MathML tool

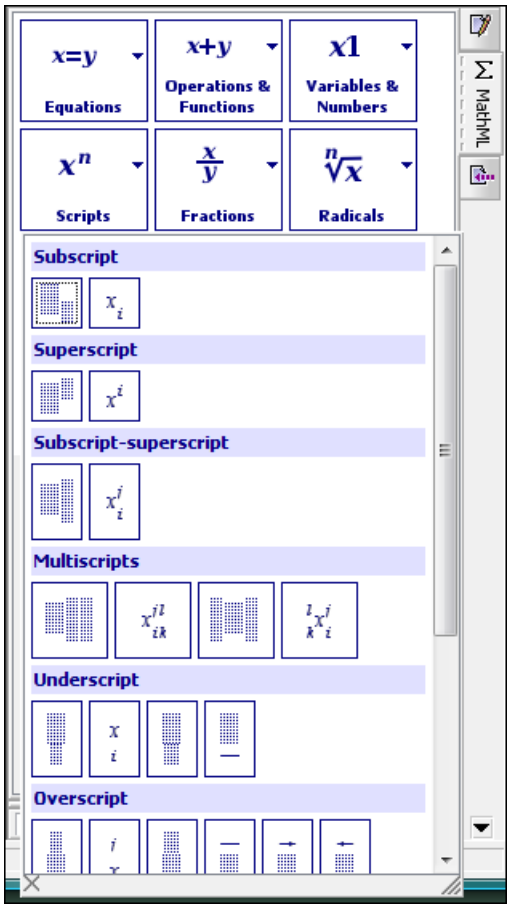


Using the MathML tool does not require you to learn MathML presentation markup. However there are three points you must learn, understand and remember before using this tool:

1. Clicking on a tool in a palette always *replaces the implicitly or explicitly selected element*[2] by the MathML template clicked upon.

[2]The explicitly selected nodes have a red box drawn around them. In absence of explicitly selected nodes, the implicitly selected element is the element having a text node child containing the caret.

Figure 2. Clicking on the "x^n Scripts" button in the MathML tool displays the Scripts palette



2. A section in a palette[3] always starts with a completely empty template followed by one or more samples of this template.

   These samples show you how, once filled, the empty template looks like. They are also useful by themselves: you may prefer clicking on them rather than the empty template and then, use the normal text selection to replace some of the text they contain.

3. As always with XMLmind XML Editor, you need to type text in the placeholders contained in the newly inserted math template. MathML has three main text containers:

   `mi`
   : represents identifiers: variable names, function names, constants, etc.

   `mn`
   : represents numbers.

   `mo`
   : represents operators (e.g. "+"), fences (e.g. "("), separators (e.g. ",").

   Most placeholders are `mi` elements[4]. Make sure not to type a number inside an `mi` element, not to type a variable name in an `mn` element, etc. If you need to specify a number, click inside the `mi` placeholder and use the "Variables & Numbers" palette to replace it by a number.

[3] In the above screenshot, the first section of the Scripts palette is Subscript, the second one is Superscript, etc
[4] The samples found after the empty template give you a hint on the nature of the placeholders.

Why all this trouble? After all, we are talking about presentation markup and not about content, semantic, markup. The reason is that *using the right presentation element also means using the right typesetting rules*.

Example 1: typing a single letter in an `mi` element renders this element using an italic font, while typing a longer identifier renders this `mi` element using an upright font.

Example 2: typing a curly brace "`{`" inside an `mo` element placed before a ``tall element'' (e.g. a fraction) will cause the curly brace to stretch vertically. Note that there is no way to force a curly brace to stretch vertically if you type this character in an `mn` or `mi` element.

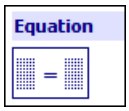
Example 3: typing a "+" sign inside an `mo` element placed between two elements automatically adds the right amount of space before and after the "+" sign.

3.1. Sample editing session

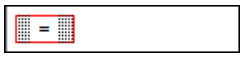
Suppose that you need to type the following, very simple, inline equation in a DocBook 5 document: $E = mc^2$.

1. Use the "Add MathML equation" button found in the DocBook toolbar and select inlineequation.

2. Click inside the newly inserted element and select the Equations palette in the MathML tool. Click on the Equation template:

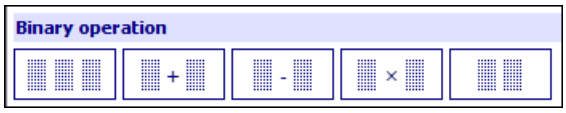
   This gives you:

3. Type "`E`" in the left placeholder:

4. Click in the right placeholder then select the ``Invisible Times'' binary operation (rightmost template in the screenshot below) from the "Operations & Functions" palette.

   This gives you:

5. Type "`m`" in the left placeholder:

6. Click in the right placeholder then select the empty Superscript template in the Scripts palette:

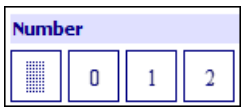
   

   This gives you:

7. Type "`c`" in the left placeholder:

8. Do not type "2" in the right placeholder. Instead click inside this placeholder then select 2 from the "Variables & Numbers" palette:

   This finally gives you:

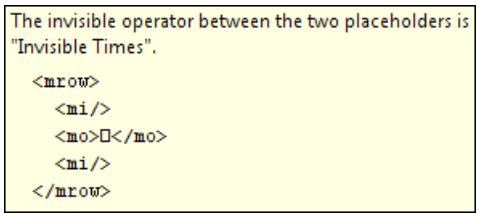
   

Notice how the equation was built using a *top to bottom* approach: equation, left hand, right hand, binary operation, left operand, right operand, etc. Using such top to bottom approach is absolutely required when you work with the MathML tool. If you prefer the more natural left to right approach, then you really have to learn MathML (`mrow`, etc) and work with the Edit tool (Insert After, Wrap, etc).

3.2. Useful tips

- When you don't understand a template, move your mouse over it and you'll see a useful ``balloon help".

Figure 3. Tool tip for the ``Invisible Times'' binary operation

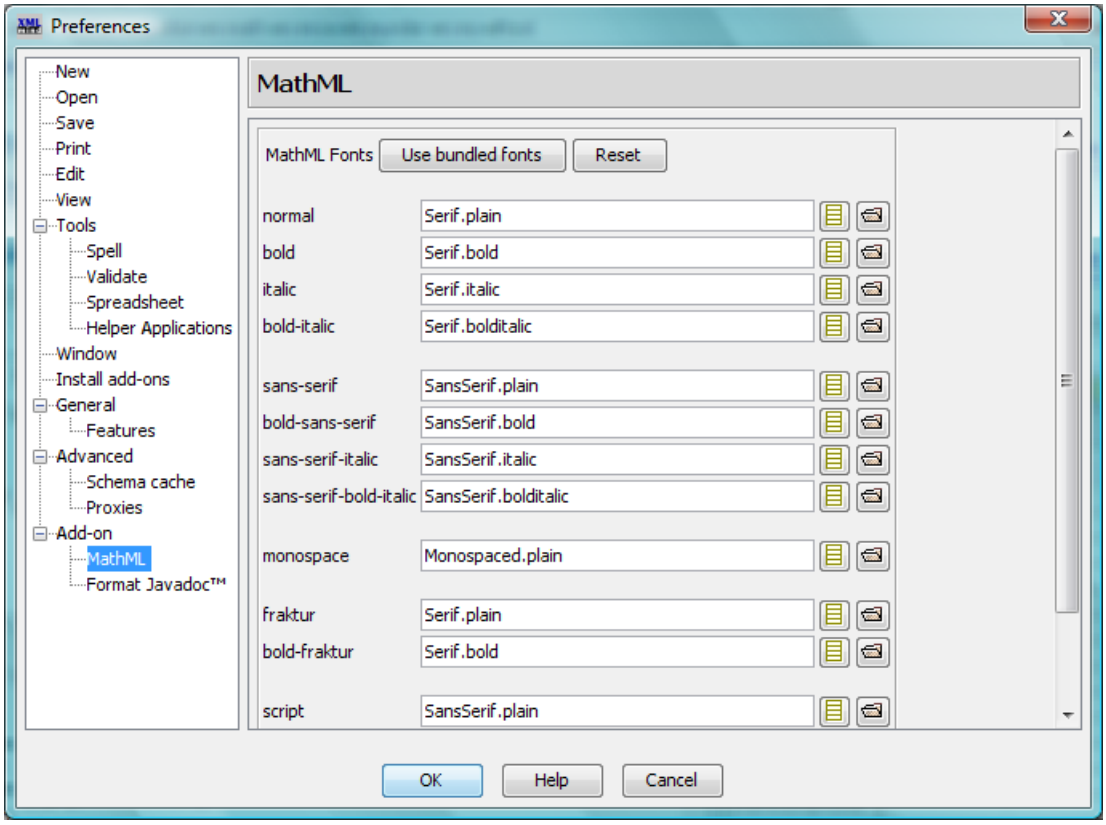


- The replacement operations performed using a MathML palette are repeatable. That is, suffice to click elsewhere and press **Ctrl**+**A** (**Cmd**+**A** on the mac) to repeat the same replacement operation.
- The replacement operations performed using a MathML palette can be recorded using the macro recorder (Tools → Record Macro → Start).
- The MathML tool has a "Text & Space" palette. Use it if you want to insert text which is not `mi`, `mn` or `mo`.
- If you want to edit an `mtable`, for example add a column to a matrix or add an equation to a set of equations, use the table editing commands found in the DocBook menu (e.g. DocBook → Column → Insert Before) and in the MathML menu (e.g. MathML → Row → Copy).
- If you want to insert special characters, use the Characters tool or use the "Insert MathML Character by Name" found in the DocBook menu and in the MathML menu.
- It is fairly easy to tweak the look of an equation created using the MathML tool. For this, you'll generally insert `mspace` elements (Edit → Insert After), wrap one or more elements into a `mphantom` or `mpadded` element (Edit → Convert [wrap]) or specify attribute values using the Attributes tool. Of course, doing this requires you to learn MathML presentation markup.

4. MathML preferences

Installing the MathML add-on adds a MathML preference sheet to the Preferences dialog box (Options → Preferences). This preference sheet allows you to specify which fonts are to be used when rendering MathML.

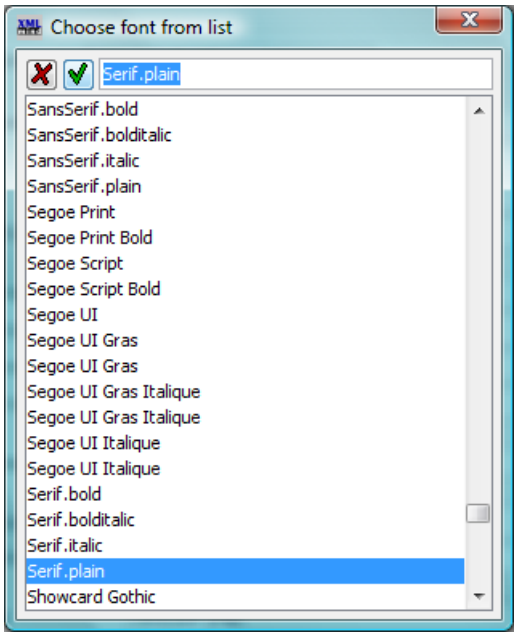
Figure 4. The MathML preference sheet



Notice that, by default, the standard Java™ logical fonts (Serif, SansSerif, Monospaced) are also used to render the MathML elements.

There are situations where you'll want to change the above mapping. For example, on the Mac, the Serif logical font is mapped to the Times physical font and this font is intrinsically too small. In such case, you may use the "Choose font from list" button found at the right of each font field. This button displays a list allowing you to choose any True Type font installed on your system.

Figure 5. Dialog box displayed by the "Choose font from list" button



Alternatively, you may use the "Choose font file" button found at the very right of each font field. This button displays the standard file chooser which allows you to specify any `.ttf` font file, whether this font has been installed or not on your system.

4.1. Bundled math fonts

The MathML add-on is bundled with the following math fonts:

Logical MathML Font	Bundled Font (found in *`mathml_addon_install_dir`*`/fonts/`)	Origin
normal	`DejaVuSerif.ttf`	DejaVu
bold	`DejaVuSerif-Bold.ttf`	
italic	`DejaVuSerif-Italic.ttf`	
bold-italic	`DejaVuSerif-BoldItalic.ttf`	
sans-serif	`DejaVuSans.ttf`	
bold-sans-serif	`DejaVuSans-Bold.ttf`	
sans-serif-italic	`DejaVuSans-Oblique.ttf`	
sans-serif-bold-italic	`DejaVuSans-BoldOblique.ttf`	
monospace	`DejaVuSansMono.ttf`	
fraktur	`eufm10.ttf`	AMS
bold-fraktur	`eufb10.ttf`	
script	`eusm10.ttf`	
bold-script	`eusb10.ttf`	
double-struck	`msbm10.ttf`	

If you want to use the all above bundled fonts, simply click on the "Use bundled fonts" button. If you don't like the bundled fonts and want to restore the default settings, click on the Reset button.

Note that the DejaVu fonts are really good looking but lack some glyphs found in the default fonts. The AMS fonts contain very few glyphs, mainly "`a`" to "`z`", "`A`" to "`Z`" and "`0`" to "`9`".

A. Conformance with the MathML 2 W3C standard

MathML support in XMLmind XML Editor passes almost all the MathML 2 tests of the W3C (Presentation section only), modulo the following limitations and specificities. Note that all advanced features (stretching of embellished operators, alignment groups and marks, etc), expect one (see below) are generally well supported.

Implementation limitations:

- The content markup is not supported at all. Content elements are rendered as tree views.
- The rendering of accents is very poor. Each MathML element, including a `mo` used as an accent, is represented on screen by a gadget[1]. This gadget has the same width, ascent and descent as the characters it contains. In XMLmind XML Editor, there is no way to make a gadget (the accent element) overlap another one (the accented element). Therefore, there is always too much space between the accent element and the accented element.
- Negative dimensions and offsets in elements such as `mpadded` are not supported.
- Limitations related to `mtable`:
  - Vertical and horizontal stretching rules are not applied to `mtd` elements containing a stretchy operator.

[1] A very lightweight control.

  - Cells acting as labels in `mlabeledtr` elements are not flushed to the right or to the left. You need to explicitly specify a `minlabelspacing` attribute for the `mtable`.
  - A percent value for the `width` attribute of an `mtable` is not supported.
  - A `fit` or percent value for the `columnwidth` attribute of an `mtable` is not supported.
  - Value `leftoverlap` of attribute `side` of an `mtable` is considered to be equivalent to value `left`. Value `rightoverlap` of attribute `side` of an `mtable` is considered to be equivalent to value `right`.
- Limitations related to `maligngroup` and `malignmark`:
  - A cell containing alignment groups is always left-aligned. That is, in such case, the `columnalign` attribute is always ignored.
  - The `decimalpoint` attribute value is honored in most cases. However, in complex cases, you may have to insert a `malignmark` element to obtain the desired alignment.
  - Empty alignment groups are ignored by the alignment algorithm.

Implementation specificities:

- Line-breaking is not implemented.
- Invisible elements such as `mphantom`, `mspace`, etc, are rendered in light gray to allow editing them. Invisible characters such as `&InvisibleTimes;`, `&InvisibleComma;`, etc, are rendered using tiny vertically aligned glyphs to allow editing them.
- Comments and processing-instructions having a `math` element ancestor are not rendered at all.
- The `fontfamily` attribute of an `mglyph` element may reference the family of any font installed on the machine running XMLmind XML Editor. The `index` attribute of an `mglyph` element must specify the Unicode code point of the character represented by the `mglyph` (e.g. 97 for character 'a').

B. Contents of the "MathML support" add-on

The "MathML support" add-on contains:

1. A configuration which allows to create standalone documents conforming to the MathML 2 *schema*[1] (presentation markup only, content markup is not supported).

   Such standalone documents are typically used the way graphics files are. XHTML example: `<img src="equations/equation1.mml"/>`.

   Note that this configuration adds not only a MathML menu to the GUI, but also a MathML tool just below the Edit tool, at the top/right of the main window.

2. A customization of the DocBook 5 configuration. This customization allows to edit MathML elements embedded in DocBook 5 documents (by the means of elements such as `imagedata`, `equation`, `inlineequation`, etc).

   This customization also adds a "Insert MathML Character by Name" item to the DocBook menu and extends the table editing commands in order to support the `mml:mtable` element and its descendants.

   A sample DocBook 5 document containing math is found in *`mathml_addon_install_dir`*`/db5mml/sample.xml`.

[1] Opening a document starting with:

```
<!DOCTYPE math PUBLIC "-//W3C//DTD MathML 2.0//EN"
"http://www.w3.org/Math/DTD/mathml2/mathml2.dtd">
```

will of course cause the DTD to be used instead of the W3C XML Schema.

3. A customization of the DITA Topic configuration.

   This customization adds the same facilities as the above DocBook 5 customization.

   A sample DITA Topic document containing math is found in *mathml_addon_install_dir*/dit-amml/sample.dita.

4. An image toolkit similar to the one based on JEuclid.

   Note that installing this add-on also requires installing the add-on called "*JEuclid image toolkit plug-in*". That gives us *two* image toolkits which allow to convert MathML to a variety of graphics formats:

   - The ``internal" image toolkit is needed to render the contents of the MathML tool[2].
   - The JEuclid This image toolkit is used when XML documents embedding MathML or referencing MathML files are converted to formats such as HTML, PDF, RTF, etc.

Compatibility with other customizations of the DocBook 5 or DITA Topic configurations

Because this add-on customizes the stock DocBook 5 configuration bundled with XMLmind XML Editor, it is not possible to install it in conjunction with other add-ons which also customize the DocBook 5 configuration.

The same limitation applies to other customizations of the DITA Topic configuration.

C. Customizing the MathML add-on

Warning

Normal users are not supposed to do this. The intended audience for this appendix is consultants and local gurus.

1. Customizing the contents of the MathML tool

The contents of the MathML tool is specified by *mathml_addon_install_dir*/common/pane/mathml.pane. Customizing the contents of the MathML tool consists in editing this XML file in XMLmind XML Editor, saving the changes and restarting the application.

2. Bundling better math fonts

The MathML add-on is bundled with a set of fonts [7] which are not used by default. If you want to force the use of this font set or if you want to specify an alternate font set, you'll have to specify a proper value for the "mathml.fonts" preference key.

The syntax of the value of this preference key is:

```
key_value --> logical_font_name ';' font_spec
              [ ';' logical_font_name ';' font_spec ]*

logical_font_name --> 'normal'
                    | 'bold'
                    | 'italic'
                    | 'bold-italic'
                    | 'sans-serif'
                    | 'bold-sans-serif'
                    | 'sans-serif-italic'
```

[2] Yes, just the contents of the MathML tool.

```
                  | 'sans-serif-bold-italic'
                  | 'monospace'
                  | 'fraktur'
                  | 'bold-fraktur'
                  | 'script'
                  | 'bold-script'
                  | 'double-struck'

font_spec --> physical_font_name
          | TTF_file_URI
```

A *physical_font_name* example is "`Century Schoolbook L Roman`" and not just "`Century Schoolbook L`", which is a font family.

Example C.1. Force the use of the bundled font set using the command-line

(Remove the newline character found after each "`;`".)

```
xxe -putpref mathml.fonts "monospace;mathml-config:fonts/DejaVuSansMono.ttf;
sans-serif;mathml-config:fonts/DejaVuSans.ttf;
bold-sans-serif;mathml-config:fonts/DejaVuSans-Bold.ttf;
sans-serif-italic;mathml-config:fonts/DejaVuSans-Oblique.ttf;
sans-serif-bold-italic;mathml-config:fonts/DejaVuSans-BoldOblique.ttf;
normal;mathml-config:fonts/DejaVuSerif.ttf;
bold;mathml-config:fonts/DejaVuSerif-Bold.ttf;
italic;mathml-config:fonts/DejaVuSerif-Italic.ttf;
bold-italic;mathml-config:fonts/DejaVuSerif-BoldItalic.ttf;
fraktur;mathml-config:fonts/eufm10.ttf;
bold-fraktur;mathml-config:fonts/eufb10.ttf;
script;mathml-config:fonts/eusm10.ttf;
bold-script;mathml-config:fonts/eusb10.ttf;
double-struck;mathml-config:fonts/msbm10.ttf"
```

Note how the "`mathml-config:`" prefix may be used to refer to the directory where the MathML add-on has been installed. This works because the following rule has been added to *`mathml_addon_install_dir`*`/mathml_catalog.xml`:

```
<rewriteURI uriStartString="mathml-config:" rewritePrefix="." />
```

Example C.2. Force the use of the bundled font set when XMLmind XML Editor is deployed using Java™ Web Start

(Remove the newline character found after each "`;`".)

```
<application-desc main-class="com.xmlmind.xmleditapp.start.WebStart">
  <argument>-putpref</argument>
  <argument>mathml.fonts</argument>
  <argument>monospace;mathml-config:fonts/DejaVuSansMono.ttf;
sans-serif;mathml-config:fonts/DejaVuSans.ttf;
bold-sans-serif;mathml-config:fonts/DejaVuSans-Bold.ttf;
sans-serif-italic;mathml-config:fonts/DejaVuSans-Oblique.ttf;
sans-serif-bold-italic;mathml-config:fonts/DejaVuSans-BoldOblique.ttf;
normal;mathml-config:fonts/DejaVuSerif.ttf;
bold;mathml-config:fonts/DejaVuSerif-Bold.ttf;
italic;mathml-config:fonts/DejaVuSerif-Italic.ttf;
bold-italic;mathml-config:fonts/DejaVuSerif-BoldItalic.ttf;
fraktur;mathml-config:fonts/eufm10.ttf;
bold-fraktur;mathml-config:fonts/eufb10.ttf;
script;mathml-config:fonts/eusm10.ttf;
bold-script;mathml-config:fonts/eusb10.ttf;
double-struck;mathml-config:fonts/msbm10.ttf</argument>
</application-desc>
```

The "`mathml-config:`" prefix works fine even when XMLmind XML Editor is deployed using Java™ Web Start.

D. Integrating MathML with document types other than DocBook 5 and DITA Topic

Warning

Normal users are not supposed to do this. The intended audience for this appendix is consultants and local gurus.

What follows is a description of what has been done to integrate MathML with DocBook 5. Integrating MathML with other document types should be very similar.

1. Customize the configuration by first including the original configuration, as explained in *XMLmind XML Editor - Configuration and Deployment*.

```
<configuration name="DocBook v5+"
  xmlns="http://www.xmlmind.com/xmleditor/schema/configuration"
  xmlns:cfg="http://www.xmlmind.com/xmleditor/schema/configuration"
  xmlns:db="http://docbook.org/ns/docbook"
  xmlns:mml="http://www.w3.org/1998/Math/MathML">

  <include location="xxe-config:docbook5/docbook5.xxe" />

  <validate namespace="http://www.w3.org/1998/Math/MathML">1
    <relaxng location="../common_rng/mathml2.rng" />
  </validate>

  <css name="DocBook" location="css/docbook5.css" />2
  ...

  <include location="../common/mathml.incl" />3
  ...

  <elementTemplate name="mathml">4
    <equation xmlns="http://docbook.org/ns/docbook">
      <title></title>
      <mml:math xmlns:mml="http://www.w3.org/1998/Math/MathML"
                display="block">
        <mml:mi></mml:mi>
      </mml:math>
    </equation>
  </elementTemplate>
```

1. Dynamically merge the RELAX NG schema of MathML presentation markup (`mathml2.rng`) with the stock RELAX NG schema of DocBook 5 (`docbook.rng`). See below.
2. Specify a CSS style sheet merging the original CSS style sheet with the MathML one. See below.
3. Include the MathML configuration.
4. Specify one or more templates for elements embedding MathML elements.

2. Here's how the MathML schema has been dynamically merged with the DocBook 5 schema.

   a. The RELAX NG schema of DocBook 5 is declared in the stock DocBook 5 configuration `xxe-config:docbook5/docbook5.xxe` ("`xxe-config:`" resolves to *`XXE_install_dir`*`/addon/config/`).

   ```
   <relaxng location="rng/V5.0/docbook.rng" />
   ```

   b. The `validate` configuration element allows to merge one or more auxiliary DTDs or schemas to the main DTD or schema.

   ```
   <validate namespace="http://www.w3.org/1998/Math/MathML">
     <relaxng location="../common_rng/mathml2.rng" />
   </validate>
   ```

You'll find:

- the W3C XML Schema for MathML (both presentation and content markup) in *`mathml_addon_install_dir`*`/standalone/xsd/mathml2.xsd`,
- the RELAX NG schema for MathML (presentation markup only) in *`mathml_addon_install_dir`*`/common_rng/mathml2.rng`,
- the RELAX NG schema Compact Syntax for MathML (presentation markup only) in *`mathml_addon_install_dir`*`/common/pane/mathml2.rnc`.

  Both `mathml2.rng` and `mathml2.rnc` have been translated from the MathML DTD using trang.

However nothing forces you to merge schemas of the same kind. More information in *XMLmind XML Editor - Configuration and Deployment*.

3. Here's how the MathML CSS style sheet has been merged to the DocBook 5 CSS style sheet:

```
@import url(xxe-config:docbook5/css/docbook5.css);
@import url(../../common/mathml.css);

db5|imagedata:contains-element(mml|math),
inlineequation > mml|math,
informalequation > mml|math,
equation > mml|math {
    content: "";
}
```