

The use of Office Math ML

MathML is not an ISO standard, and Ecma 376 is not required to store mathematical definitions using this recommendation. Functional differences between MathML and the capability supported by Microsoft Office require a different solution to maintain compatibility for existing documents.

Microsoft Office Word allows users to embed arbitrary span-level material (basically anything you can put into a Word paragraph) in math zones and MathML is geared toward allowing only math in math zones. Since both MathML and OMML are XMLs, XSLTs can (and have) been created to convert one into the other.

The spirits of MathML and OMML are somewhat different. The proper comparison can be drawn by referring to the [MathML presentation tag set](#), rather than the [content tag set](#), since the main emphasis is on presentation. A quick summary of the difference in spirits is

1. MathML built-up objects may be described by an infix notation, while OMML's are described by a prefix notation
2. MathML built-up object arguments are defined positionally, while OMML's are tagged explicitly

With MathML when you find an `<mrow>`, you look at the next tag(s) to see what's inside. If you find an `<mo>` entry, you have an operator, which you look up in your operator table to figure what kind of a possibly built-up object is involved. It could be an open fence (parentheses, brackets, braces, etc.), an n-ary operator, a function-apply (for trigonometric and other math functions), or one of many operators that don't get built up. For fences MathML also has the `<mfenced>` tag, which is essentially the same as OMML's delimiter `<d>` tag.

Each way of representing fences has its advantages, the infix approach allowing embellished fences (such as underlined fences) and the `<mfenced>` approach allowing a sequence of separated arguments. OMML's `<d>` can't represent embellished fences, but fortunately for OMML, they aren't common. Built-up expressions like subscripts and superscripts are represented by prefix notations in both MathML and OMML.

The following table summarizes the built-up objects in the Office math model along with the OMML and target MathML tags

Built-up Office Math Object	OMML tag	MathML
Accent	acc	mover/munder
Bar	bar	mover/munder
Box	box	menclose (approx)
BoxedFormula	borderBox	menclose
Delimiters	d	mfenced
EquationArray	eqArr	mtable (with alignment groups)
Fraction	f	mfrac
FunctionApply	func	&FunctionApply; (binary operator)
LeftSubSup	sPre	mmultiscripts (special case of)
LowerLimit	limLow	munder

Matrix	m	mtable
Nary	nary	mrow followed by n-ary mo
Phantom	phant	mphantom and/or mpadded
Radical	rad	msqrt/mroot
GroupChar	groupChr	mover/munder
Subscript	sSub	msub
SubSup	sSubSup	msubsup
Superscript	sSup	msup
UpperLimit	limUpp	mover

OMML tags are always written with a math namespace prefix like "m:" and I recommend this convention for MathML as well. The reason is that these XMLs are useful in many contexts, not just in HTML(5) and using namespace prefixes allows the XML parser to delegate to the appropriate tag-set owner.

Comparing the two ways of representing the built-up fraction, one can see how OMML has explicit argument tags, whereas MathML determines arguments by position. The built up version of the fraction a/b in OMML is represented by (aside from possible properties)

```
<m:f>
  <m:num>
    <m:mr>a</m:mr>
  </m:num>
  <m:den>
    <m:mr>b</m:mr>
  </m:den>
</m:f>
```

where we see how the numerator and denominator are tagged explicitly. In MathML, these arguments are given by the next entity and the one after that, respectively:

```
<m:mfrac>
  <m:mi>a</m:mi>
  <m:mi>b</m:mi>
</m:mfrac>
```

This comparison reveals that OMML can be more verbose than MathML. A less verbose comparison results for the fraction $(a+b)/c$, since in OMML it's

```
<m:f>
  <m:num>
    <m:mr>a+b</m:mr>
  </m:num>
  <m:den>
    <m:mr>c</m:mr>
  </m:den>
</m:f>
```

whereas in MathML, it's

```
<m:mfrac>
  <m:mrow>
    <m:mi>a</m:mi>
    <m:mo>+</m:mo>
    <m:mi>b</m:mi>
  </m:mrow>
  <m:mi>c</m:mi>
</m:mfrac>
```

Here the `<m:mrow>` is needed for the numerator to make it the first entity following the `<m:mfrac>`. For both a/b and $(a+b)/c$, the linear format sure has the shortest representation!

Another difference between MathML and OMML is in the positioning of the radical (root) degree and prescript arguments relative to their respective bases. In OMML these arguments are positioned so that the left and right arrow keys traverse the objects unidirectionally. Specifically for the radical, the degree argument precedes the radicand, while for MathML it follows the radicand. By having it precede, a right arrow key at the start of the radical moves into the degree and then into the radicand, exactly the way one would expect geometrically. Similarly OMML's prescripts precede the base, whereas in MathML's multiscripts object they follow the base.