2. MathML

Recall that MathML actually consists of two separate XML applications--Presentation MathML and Content MathML. In this section, we will study the two versions in more detail, primarily through a few examples.

There are some technical issues involved with compound XHTML + MathML documents, such as file formats, linking to stylesheets, name spaces, entities, and document types. We will not discuss these issues in thie article. For more information, see the MathML site at W3C. In addition, you can look at the source code of this page (unless you are viewing the PDF version) to see how some of the details are handled.

2.1 Presentation MathML

Presentation MathML is a fairly complete specification of modern mathematical notation. Almost any mathematical expression that the average math teacher or researcher would want can be expressed in Presentation MathML. The language works by marking up the two-dimensional representation of the expression, from left to right, in a logical way. A few of the essential Presentation MathML elements are

- `<mi></mi>` for enclosing a basic variable or other entity.
- `<mn></mn>` for enclosing a number.
- `<mo></mo>` for enclosing an operator.
- `<mrow></mrow>` for logically grouping other elements.
- `<msub></msub>` for creating a subscript.
- `<msup></msup>` for a superscript.
- `<msupsub></msupsub>` for a superscript and subscript.
- `<mfrac></mfrac>` for a fraction.

In each of the following examples, a mathematical expression is given followed by the corresponding Presentation MathML markup

Simple function notation

$$f(x,\ y)$$

```
<math>
  <mrow>
    <mi>f</mi>
    <mo>(</mo>
    <mrow>
      <mi>x</mi>
      <mo>,</mo>
      <mi>y</mi>
    </mrow>
    <mo>)</mo>
  </mrow>
</math>
```

A fraction

$$\frac{x^2+1}{\sin(y_1-3)}$$

```
<math>
    <mfrac>
        <mrow>
          <msup>
              <mi>x</mi>
              <mn>2</mn>
          </msup>
          <mo>+</mo>
          <mn>1</mn>
        </mrow>
        <mrow>
          <mi>sin</mi>
          <mo>(</mo>
          <mrow>
             <msub>
                 <mi>y</mi>
                 <mn>1</mn>
             </msub>
             <mo>−</mo>
             <mn>3</mn>
          </mrow>
          <mo>)</mo>
        </mrow>
    </mfrac>
</math>
```

A series

$$\sum_{n=1}^{\infty}\frac{1}{n^2}=\frac{\pi^2}{6}$$

```
<math>
   <mrow>
```

```
    <mrow>
      <msubsup>
        <mo>&sum;</mo>
        <mrow>
          <mi>n</mi>
          <mo>=</mo>
          <mn>1</mn>
        </mrow>
        <mi mathvariant="normal">&infin;</mi>
      </msubsup>
        <mrow>
          <mfrac>
            <mn>1</mn>
            <mrow>
              <msup>
                <mi>n</mi>
                <mn>2</mn>
              </msup>
            </mrow>
          </mfrac>
        </mrow>
    </mrow>
    <mo>=</mo>
    <mrow>
      <mfrac>
        <mrow>
          <msup>
            <mi mathvariant="normal">&pi;</mi>
            <mn>2</mn>
          </msup>
        </mrow>
        <mn>6</mn>
      </mfrac>
    </mrow>
 </mrow>
</math>
--------------------------------------------------------------------------------
```

If you have any experience with markup languages, you should have no trouble understanding the logic of the markup in these examples. On the other hand, you are almost certainly struck by the large amount of markup needed for relatively simple mathematical expression. Presentation MathML actually goes quite a long way in describing the underlying structure of mathematical expressions. But, as these examples show, Presentation MathML is verbose to the point of being unreadable and requires, as the name suggests, that all notational elements be specifically marked up. This notational markup is hardwired, of course, and can only be changed one element at a time..

2.2 Content MathML

Content MathML is a bit harder to understand at first than Presentation MathML. Once mastered, however, Content MathML has some distinct advantages, as we will see. We will revisit the examples given above in the section on Presentation MathML. As before, in each example a mathematical expression is given followed by the corresponding Content MathML markup.

The `<apply></apply>` tag is one of the most basic and important tags in Content MathML. In its simple form, it is used to denote a function; the first child is the function name and subsequent children are the arguments. For the simple function example, compare the Content MathML makrup below with the corresponding Presentation MathML markup.

The function revisited

$$f(x,\ y)$$

```
<math>
  <apply>
    <ci>f</ci>
    <ci>x</ci>
    <ci>y</ci>
  </apply>
</math>
```

The `<apply></apply>` tag is also used in conjunction with a number of unary tags to denote special operations. Some of the simple ones are

- `<plus />` used with two or more siblings for addition
- `<minus />` used with one sibling for unary minus, or with two siblings for subtraction
- `<power />` used with two siblings for raising a base to a power
- `<times />` used with two or more siblings for multiplication
- `<divide />` used with two siblings for division
- `<union />` used with two or more siblings for set union
- `<intersect />` used with two or more siblings for set intersection
- `<selector />` used with two or more siblings, is similar to the basic function construction, but with the arguments as subscripts. Thus, the first sibling is the sequence name while the remaining siblings are the arguments.

For the fraction example, compare the Content MathML markup below with the corresponding the Presentation MathML markup.

The fracion revisited

$$\frac{x^2+1}{\sin(y_1-3)}$$

```
<math>
```

```
  <apply><divide />
    <apply><plus />
      <apply><power />
        <ci>x</ci> <cn>2</cn>
      </apply>
      <cn>1</cn>
    </apply>
    <apply><sin />
      <apply><minus />
        <apply><selector />
          <ci>y</ci> <cn>1</cn>
        </apply>
        <cn>3</cn>
      </apply>
    </apply>
  </apply>
</math>
```

The <apply></apply> tag is also used for more complicated constructions, in conjunction with some other specialized binary tags:

- `<bvar></bvar>` for a bound variable
- `<lowlimit></lowlimit>` for a lower limit
- `<uplimit></uplimit>` for an upper limit
- `<condition></condition>` for a condition

The more complicated <apply></apply> constructions include

- `<sum />` for a sum indexed by a bound variable between limits or over a set
- `<int />` for an integral with respect to a bound variable between limits or over a set
- `<diff />` for a derivative with respect to a bound variable.

For the series example, compare the Content MathML markup below with the corresponding Presentation MathML markup.

The series revisited

$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}$$

```
<math>
  <apply><eq />
    <apply><sum />
      <bvar><ci>n</ci></bvar>
      <lowlimit> <cn>1</cn></lowlimit>
      <uplimit><infinity /></uplimit>
      <apply><divide />
        <cn>1</cn>
        <apply><power />
          <ci>n</ci> <cn>2</cn>
        </apply>
      </apply>
    </apply>
    <apply><divide />
      <apply><power />
        <pi /> <cn>2</cn>
      </apply>
      <cn>6</cn>
    </apply>
  </apply>
</math>
---------------------------------------------------------------------------
```

In addition to the <apply></apply> tag, other important binary tags whose meanings are fairly obvious are

- <set></set> for a set
- <vector></vector> for a vector
- <list></list> for a list

Unlike Presentation MathML, Content MathML is far from complete--indeed it barely covers the mathematics in the first two college years. Moreover, what is included and what is missing is a bit mystifying. Here are a few examples:

- Markup for factorials in included but not for binomial coefficients.
- Markup for set difference is included but not for complement.
- Markup for scalar product is included but not for norm.

Recall also that XML applications are supposed to be self-documenting. In that respect, the naming conventions in Content MathML are poorly done, in my opinion. For example, `<exponentiale />` (for the exponential *e*) and `<cartesianproduct />` (for Cartesian product) are completely written out, while `<int />` (for integral) and `<diff >` (for derivative) are ambiguously abbreviated. (Those tags could equally well apply to integer and difference, respetively.)

But on the plus side, Content MathML, while still rather verbose, is significantly less so than Presentation MathML. More importantly, Content MathML is much easier to read as mathematics than Presentation MathML. Finally, and perhaps most importantly, Content MathML allows the separation of the mathematical content of an expression from its presentation in terms of notation. How this is done is the topic of Section 3.