Fabulous Formulas

MathML: Fundamentals and Practice for Formulas

The origins of the XML-based markup language for formulas - MathML - goes back to the early days when HTML was established. Due to a lack of browser support, productive use in the web context wasn't satisfactory for a long time. While Firefox and Safari have had usable implementations for several years, Chrome and its derivatives didn't follow suit until early 2023. Let's look at the basics and its current usage options.

by Dr. Thomas Meinike

In January, The MathML Association tweeted via the account @mathml3: "The time has come. Welcome to 2023: The Year of #MathML [...]". This is in reference to the Google Chrome browser's recently released version 109. See here [1] and Figure 1 for more info. Despite all its pathos, this announcement has a true core. Namely, it's the arrival of a web technology that the W3C has specified for 25 years to represent mathematical and scientific-technical formulas in commercially available browsers.

Further integration into the browser

The first ideas of a markup language for formulas emerged as early as 1994. During a working draft on HTML 3.0, proposals for a concept called HTML Math surfaced. The W3C Math Working Group was founded in 1997. Their work resulted in the first published MathML specification in 1998 [2]. Significant further developments occurred later and are seen in the still current second edition of version 3.0 from 2014 [3]. That same year, MathML (and SVG) were adopted by the HTML5 standard. Therefore, all HTML5-capable browsers should be able to implement formulas and make them available for use. Apple and Mozilla's development teams did their homework with their browsers Safari and Firefox reasonably quickly. But the absence in other popular browsers hasn't exactly promoted broad acceptance and usability. Because of Google's initiative, Chromium-based derivatives like Brave (Brave Software), Edge (Microsoft), Opera (Opera Software), and Vivaldi (Vivaldi Technologies) are now designed to render formula content directly, in addition to Chrome. Figure 2 shows the "Can I use" implementation status from early July. Interestingly, Chrome 24 once briefly had its sights on MathML [4].

Fig. 1: MathML Association Tweet from 10.01.2023.

MathML - REC

Special tags that allow mathematical formulas and notations to be written on web pages.

Usage: % of all users — Global: 90.45% + 0.06% = 90.5%

Current aligned | Usage relative | Date relative | Filtered | All

Chrome	Edge*	Safari	Firefox	Opera	IE	Chrome for Android	Safari on iOS*	Samsung Internet	Opera Mini*	Opera Mobile*	UC Browser for Android	Android Browser*	Firefox for Android	QQ Browser	Baidu Browser	KaiOS Browser
4-23																
24				[5] 10-12.1												
25-96	12-96			15-82												
[3] 97-108	[3] 97-108	[2] 3.1-9.1	[1] 2-3.6	[3] 83-94			3.2-4.3									
[4] 109-113	[4] 109-113	10-16.4	4-113	[4] 95-98	6-10		5-16.4	4-20		12-12.1		2.1-4.4.4				2.5
[4] 114	[4] 114	16.5	114	[4] 99	11	[4] 114	16.5	21	all	73	13.4	[4] 114	113	13.1	13.18	3.1
[4] 115-117		16.6-TP	115-116				16.6-17									

Fig. 2: MathML's browser support [6].

Incidentally, since 2011, the technology has also been part of the e-books ecosystem in EPUB 3.x format. At this point, I should mention that the possibility of making formulas accessible with the JavaScript library MathJax [5] has existed for some time. Essentially, this needs a one-liner in the HTML document's head element with the CDNlink.

Firefox and Safari exhaust MathML 3.0's capabilities. Meanwhile, Google and co. primarily address the new W3C initiative MathML Core [7] (the core area that can now be used generally without problems), which is nearing completion.

Work is also being done on version 4.0. Corresponding extensions are especially interesting for technical communication document formats like DocBook and DITA or subject-specific applications. Therefore, this article deals with the Core variant's "Presentation Markup" capabilities used for representations. The Content Markup addresses the meaning of formulas in terms of content and can be used for interpretation with mathematical software.

MathML Core

To integrate individual formulas (or parts of them), a *math* element is inserted in the HTML structure at the desired position. The display can be single-line within a text paragraph, or in block form. The attribute *display* with the values *inline* or *block* is used for this. Formatting can be done analogously in the CSS stylesheet with the *display* property. The following code shows the basic structure of a *math* container. The namespace specification is optional, at least in HTML5 documents. But the basic notation is recommended for compatibility with other standards.

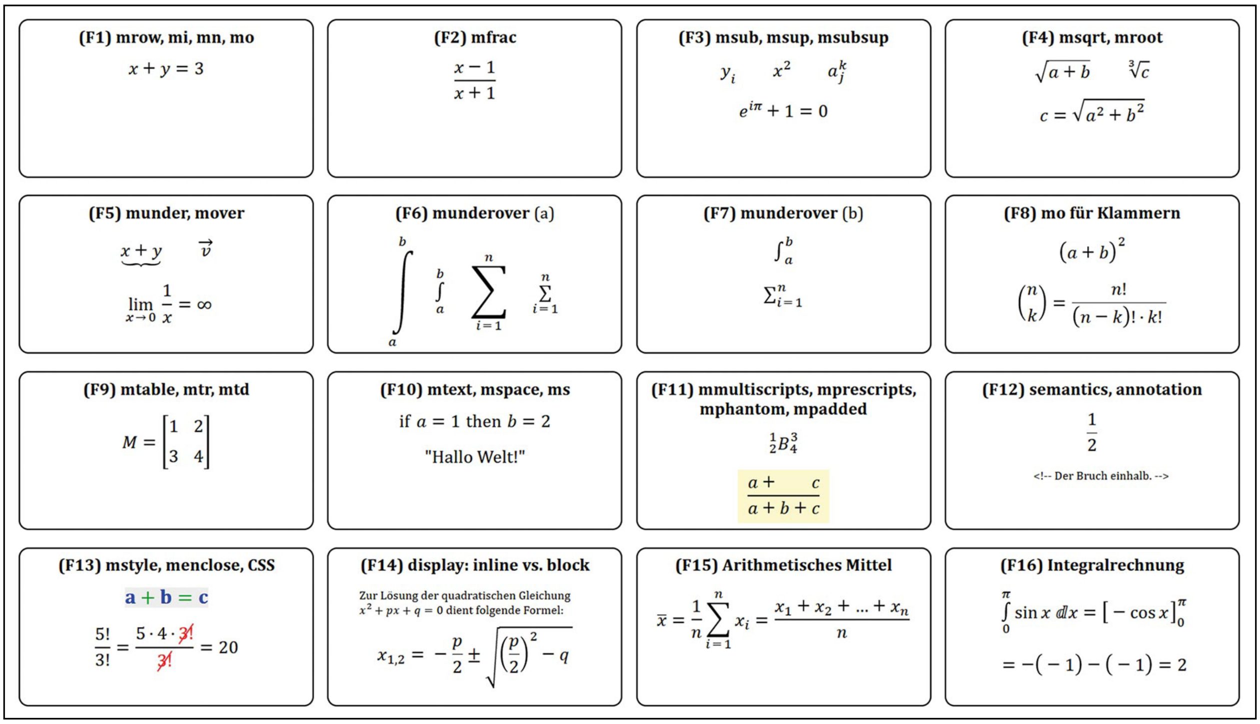

Fig. 3: MathML test formulas

```
<math xmlns="http://www.w3.org/1998/Math/MathML" display="...">
                                  <!-- weitere Inhalte ... --> </math>
```

In the specific individual examples the enclosing *math* element isn't present. With one exception, we assume the *block* variant. There are about 30 child elements below *math*, most of which start with the letter *m* and can also occur in additional nestings. The reference to the test formulas in Figure 3 is made with the respective number in round brackets, like (Fn). These can be traced online here.

mrow

The *mrow* element is used to delimit and group expressions, terms, or other related parts of a formula. Listing 1 describes an equation's simple linear arrangement, see (F1). Here, *mrowr* isn't absolutely necessary for representation, but it emphasizes the principle function of enclosing contents.

mi, mn, and mo

The *mi* elements are of utmost importance. They are already used in Listing 1 for variables, constants, and function names (identifiers), *mn* for integer or floating point values (number), and *mo* for basic operations like addition and the equals sign (operator).

mfrac

Fractions are marked with *mfrac* in the standard form of the numerator above the denominator. The parts are arranged one above the other according to Listing 2, where *mrow* is needed for their delimitation, cf. (F2).

msub, msup, and msubsup

The subscript or superscript of indices, variables, or terms is often required and realized using the elements *msub* and *msup*. Depending on the content being handled, *mrow* may be needed for splitting. The *msubsup* element is used for common subscripts and superscripts, as shown in the upper part of (F3) in Listing 3.

In the lower part of (F3), you can see the Euler's identity formula. The product *iπ* bound in mrow is superscripted there. As the imaginary unit *i* and the circle number *π* represent constants, both are stored in *mi* (Listing 4).

msqrt and mroot

Square roots are described with *msqrt* and *nth* roots with *mroot*. The content under the root and, for *nth* roots, the extra root exponent must be clearly delimited. Listing 5 shows the procedure for both types of roots. Listing 6 describes the well-known Pythagoras formula in the transformation as a root expression; see (F4).

munder and mover

Setting formula parts or only single characters below or above terms is frequently a requirement. This is *munder* and *mover*'s responsibility; see Listing 7 and (F5). A horizontally aligned curly bracket is used in the *munder* part. Special characters can be displayed directly in code editors using the available fonts. In this case, the alternative entity reference to the hexadecimal Unicode point x23DF was used. The arrow selected in the *mover block* is inserted directly.

Listing 8 and (F5) show a limit's notation with the statement "limits *x* against *0*" declared by *munder* with reference to the result "infinity" – evident as a direct sign.

munderover

The parallel content setting below and above a construct is a typical task when creating integrals and sums. The element munderover is used for this. The basic structure of the variants shown in (F6) and (F7) is shown in Listing 9. There are also three attributes for the mo element for designing integral or sum characters: stretchy, largeop, and moveablelimits. They can each be assigned the Boolean values true or false. The first two elements can be used to change the extent and size, while the latter allows a more compact lateral arrangement of integral or sum limits.

Brackets with mo

MathML 3.0 provides a separate element for brackets called mfenced that has open and close attributes to

Listing 1

```
<mrow>
  <mi>x</mi>
  <mo>+</mo>
  <mi>y</mi>
  <mo>=</mo>
  <mn>3</mn>
</mrow>
```

Listing 2

```
<mfrac>
  <mrow>
    <mi>x</mi>
    <mo>–</mo>
    <mn>1</mn>
  </mrow>
  <mrow>
    <mi>x</mi>
    <mo>+</mo>
    <mn>1</mn>
  </mrow>
</mfrac>
```

Listing 3

```
<msub>
  <mi>y</mi>
  <mi>i</mi>
</msub>
<msup>
  <mi>x</mi>
  <mn>2</mn>
</msup>
<msubsup>
  <mi>a</mi>
  <mi>j</mi>
  <mi>k</mi>
</msubsup>
```

Listing 4

```
<msup>
  <mi>e</mi>
  <mrow>
    <mi>i</mi>
    <mi>π</mi>
  </mrow>
</msup>
<mo>+</mo>
<mn>1</mn>
<mo>=</mo>
<mn>0</mn>
```

Listing 5

```
<msqrt>
  <mi>a</mi>
  <mo>+</mo>
  <mi>b</mi>
</msqrt>
<mroot>
  <mi>c</mi>
  <mn>3</mn>
</mroot>
```

Listing 6

```
<mi>c</mi>
<mo>=</mo>
<msqrt>
  <msup>
    <mi>a</mi>
    <mn>2</mn>
  </msup>
  <mo>+</mo>
  <msup>
    <mi>b</mi>
    <mn>2</mn>
  </msup>
</msqrt>
```

assign the bracket type. Meanwhile, MathML Core relies on using mo. The specific brackets are noted in it. Therefore, two mo elements are needed for start and end brackets. Listing 10 describes the upper part of (F8). Its lower part is a formula from combinatorics. Since the code is a little more extensive, see the provided test page. What's interesting here is the use of the previously mentioned attribute stretchy="true" for the big brackets on the left side of the equation. This was generated as a fraction with mfrac, just like the right side. However, the fraction line is hidden because of the linethickness="0" attribute.

Since Firefox knows a large part of Math-ML 3.0, it will also interpret mfenced. When in doubt, it's advised that you use a Chromium browser for testing core compatibility when working with legacy code to reach the widest possible audience.

mtable, mtr and mtd

These element names are obviously connotative of the HTML elements used for tables, underlying rows, data cells, and act as constructors for matrices. The code for implementing the matrix constructed in (F9) is shown in Listing 11. Stretchy="true" also governs the expansion of the large rectangular brackets. Alternative brackets could be round or curly brackets or straight strokes.

mtext, mspace and ms

The element mtext lets you insert additional text fragments. In Listing 12, a type of condition is written down. This can be used in a proof or program logic, see (F10). mspace is used to increase distances without overloading the listings. It is also used in most of the formulas here.

The ms element is worth mentioning as it can only encapsulate string literals. The lower part of (F10)'s output comes from "Hello world!". MathML Core needs the quotes; in version 3.0 they are generated without by default.

mmultiscripts and mprescripts

With the mmultiscripts and mprescripts elements, objects can be obtained according to Listing 13 and (F11), as seen above. Starting from a base (called B), it can have indices on two sides above and below. For that reason, it's important to pay attention to the order in the code.

mphantom and mpadded

The element mphantom is used to hide fragments of a formula. For example, it can arrive at the impression seen in the lower part of (F11). The fraction's numerator, b and + are present, but aren't displayed. The remaining c is still above the one visible in the denominator. Here, mpadded is also present. This lets you highlight formula parts in color, block by block and position the attributes lspace, voffset, width, and height (Listing 14).

semantics and annotation

Generally, formulas can be enclosed in semantics elements. This is also done by some formula editors. This isn't absolutely necessary unless you want to include descriptive information. Listing 15 shows this with the annotation element intended for it. This text part isn't

Listing 7

```
<munder>
  <mrow>
    <mi>x</mi>
    <mo>+</mo>
    <mi>y</mi>
  </mrow>
  <mo>&#x23DF;
        </mo>
</munder>
<mover>
  <mi>v</mi>
  <mo>→</mo>
</mover>
```

Listing 8

```
<munder>
  <mo>lim</mo>
  <mrow>
    <mi>x</mi>
    <mo>→</mo>
    <mn>0</mn>
  </mrow>
</munder>
<mfrac>
  <mn>1</mn>
  <mi>x</mi>
</mfrac>
<mo>=</mo>
<mi>∞</mi>
```

Listing 9

```
<munderover>
  <mo>∫</mo>
  <mi>a</mi>
  <mi>b</mi>
</munderover>
<munderover>
  <mo>∑</mo>
  <mrow>
    <mi>i</mi>
    <mo>=</mo>
    <mn>1</mn>
  </mrow>
  <mi>n</mi>
</munderover>
```

Listing 10

```
<msup>
  <mrow>
    <mo>(</mo>
    <mi>a</mi>
    <mo>+</mo>
    <mi>b</mi>
    <mo>)</mo>
  </mrow>
  <mn>2</mn>
</msup>
```

Listing 11

```
<mi>M</mi>
<mo>=</mo>
<mrow>
  <mo stretchy="true">[</mo>
  <mtable>
    <mtr>
      <mtd><mn>1</mn></mtd>
      <mtd><mn>2</mn></mtd>
    </mtr>
    <mtr>
      <mtd><mn>3</mn></mtd>
      <mtd><mn>4</mn></mtd>
    </mtr>
  </mtable>
  <mo stretchy="true">]</mo>
</mrow>
```

Listing 12

```
<mtext>if</mtext>
<mspace width="0.3em"/>
<mi>a</mi>
<mo>=</mo>
<mn>1</mn>
<mspace width="0.3em"/>
<mtext>then</mtext>
<mspace width="0.3em"/>
<mi>b</mi>
<mo>=</mo>
<mn>2</mn>
```

Listing 13

```
<mmultiscripts>
  <mi>B</mi>
  <mn>4</mn>
  <mn>3</mn>
  <mprescripts/>
  <mn>2</mn>
  <mn>1</mn>
</mmultiscripts>
```

shown in the browser, so only the break itself and an added comment are visible in (F12).

Formatting

Formula components can be initially formatted with the usual CSS techniques, including element selectors, context selectors, classes, and IDs. Inline styles can also be assigned with the style attribute. MathML provides direct approaches to individual formatting with the mstyle element and other math* attributes. Code for displaying the upper block of (F13) corresponds to Listing 16.

A gray background, a blue foreground color for formula contents and the font size with a bold font style are specified with mstyle The green color is achieved for the two plus operators by applying the mathcolor attribute.

Listing 14

```
<mpadded
mathbackground="#FFC" lspace=
"0.5em" voffset="0.75em" width=
        "5em" height="3em">
  <mfrac>
    <mrow>
      <mi>a</mi>
      <mo>+</mo>
      <mphantom>
        <mi>b</mi>
        <mo>+</mo>
      </mphantom>
        <mi>c</mi>
      </mrow>
      <mrow>
        <mi>a</mi>
        <mo>+</mo>
        <mi>b</mi>
        <mo>+</mo>
        <mi>c</mi>
      </mrow>
    </mfrac>
  </mpadded>
```

Listing 15

```
<semantics>
  <mfrac>
    <mn>1</mn>
    <mn>2</mn>
  </mfrac>
  <annotation>Der Bruch
   einhalb.</annotation>
</semantics>
```

Listing 16

```
<mstyle mathbackground="#EEE"
     mathcolor="#00F" mathsize=
      "1.1em" mathvariant="bold">
  <mi>a</mi>
  <mo mathcolor="#090">+</mo>
  <mi>b</mi>
  <mo mathcolor="#090">=</mo>
  <mi>c</mi>
</mstyle>
```

Listing 17

```
math
{
  font-family:
 "Cambria Math", math;
  font-size: 1.5em;
  text-align: left;
  width: fit-content;
  white-space: nowrap;
}
```

Listing 18

```
<menclose notation="updiagonalstrike"
  style="text-decoration: line-through;
                        color: #F00">
  <mn>3</mn>
  <mo>!</mo>
</menclose>
```

By default, browsers center display block type formulas. This is common practice in scientific publications. For the math element, the indentalign="left" / "right" attribute would change this behavior, but it hasn't made its way into the core standard yet. The solution is the CSS property text-align used in conjunction with an appropriate width assignment. Listing 17 contains this and some other declarations. You can also see the new generic font assignment math.

menclose

The useful menclose element is used to enclose parts of the formulas, but it isn't in the core standard yet. With its notation attribute, it allows certain outlines and strikethroughs. This is interesting for shortening fractions or other similar requirements. Listing 18 shows an excerpt of shortening of 3! (factorial of 3, i.e. 3 * 2 * 1) in the lower formula representation of (F13). If notation is recognized, say in Firefox for example, then a diagonal line appears from bottom left to top right. A Chromium browser wouldn't generate a line at all. With the additional style specification, then at least horizontal strikethrough is generated.

The other demonstrations can be followed in the more extensive code on the browser's test page. (F14) illustrates the interaction of inline and block formatting. The inline formula is in the introductory text paragraph, while the actual main formula is blocked below. The

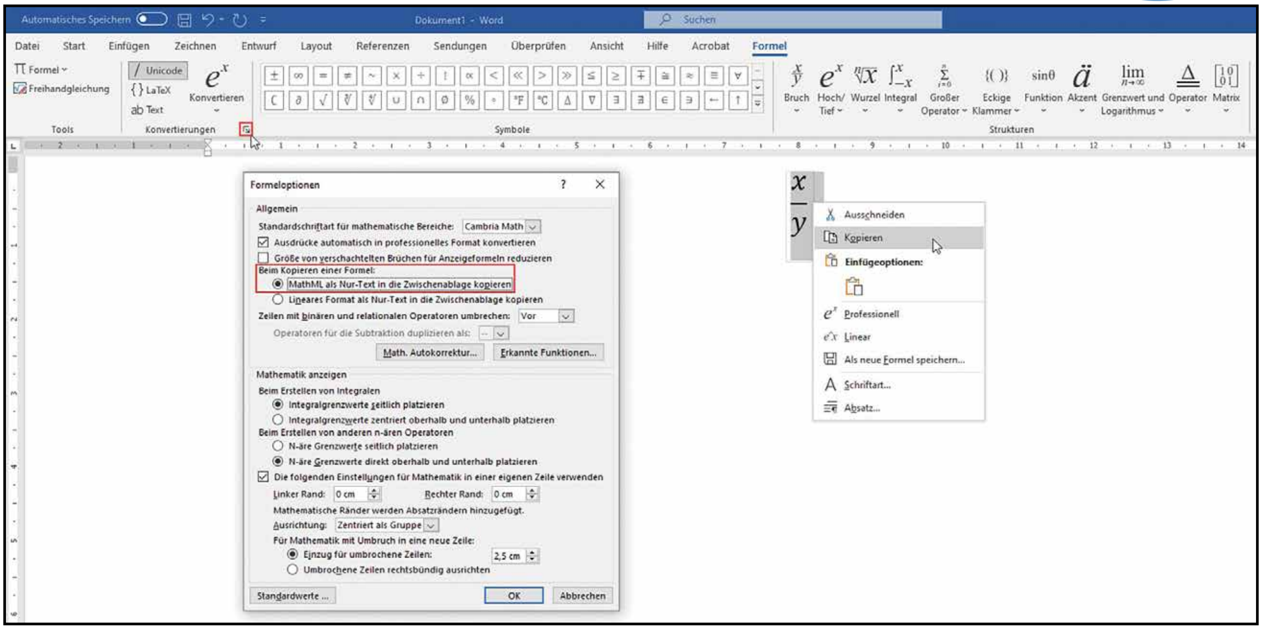

Fig. 4: Word formula editor with MathML export

arithmetic mean (F15) and integral calculus (F16) are other typical formulas you can generate using the basic techniques described here.

Formula tools

Formula code can be conveniently written in editors equipped for this, like the commercial XML editor and the freely available Visual Studio Code with MathML extension. MathType is a specialized commercial formula editor. The demo version is usable for MathML export. The powerful MATLAB software is well-known in the engineering field. It can also be used to convert formula expressions to MathML. The popular Wolfram platform bundles some useful online tools on this topic.

Popular word processors like Microsoft Word and OpenOffice/LibreOffice have formula editors that allow you to export code for graphically generated formulas. Figure 4 shows how to creating and exporting a formula with Word.

Listing 19 contains code for the fraction x / y that was exported from Word with the clipboard. You'll notice the unwanted mml: namespace prefixes for all elements.

Listing 19

```
<mml:math xmlns:mml=http://www.w3.org/1998/Math/MathML
    xmlns:m="http://schemas.openxmlformats.org/officeDocument/
                                                    2006/math">
    <mml:mfrac>
      <mml:mrow><mml:mi>x</mml:mi>
    </mml:mrow>
    <mml:mrow>
      <mml:mi>y</mml:mi>
    </mml:mrow>
  </mml:mfrac>
</mml:math>
```

You can eliminate these by simply searching and replacing them with an empty string. Additionally, xmlns:mml is shortened to xmlns and the namespace xmlns:m from the Office ecosystem is removed. As discussed at the beginning of this article, mrow can also be omitted here.

Conclusion and outlook

MathML is becoming more attractive in the context of using HTML5 due to improved support in common browsers. Direct integration into websites improves usability and accessibility compared to raster graphics with formula content. The code editor and graphical formula tool support makes practical use easier. After years of stagnation, the specification process has been revitalized, so expect further enhancements and improvements in the future.

Finally, I recommend the MDN [9] for in-depth learning and reference, which is popular in the web development field and focuses on MathML Core.

Dr. Thomas Meinike works as a teacher in Merseburg. His work is focused on XML applications in technical documentation, online-help and web development.

References

[1] https://twitter.com/mathml3/status/1612881623510388738
[2] https://www.w3.org/TR/1998/REC-MathML-19980407/
[3] https://www.w3.org/TR/MathML/
[4] https://caniuse.com/?search=MathML
[5] https://www.mathjax.org/
[6] https://caniuse.com/?search=MathML
[7] https://w3c.github.io/mathml-core/
[8] https://w3c.github.io/mathml/
[9] https://developer.mozilla.org/en-US/docs/Web/MathML