

mathym1

Convert your equations to MathML

v0.1.0 2025-05-29

<https://codeberg.org/akida/mathym1>

Akida

Contents

1. Overview	3
2. Quickstart	3
3. Missing/ non-working features	4
2. Reference	6
2.1. lib	6
2.1.1. include-mathfont	6
2.1.2. maybe-html	6
2.1.3. stylesheets	7
2.1.4. to-mathml	7
2.1.5. to-mathml-raw	8
2.1.6. try-to-mathml	9
2.2. prelude	10
2.2.1. bb	10
2.2.2. bold	10
2.2.3. cal	10
2.2.4. display	11
2.2.5. frac	11
2.2.6. inline	11
2.2.7. italic	12
2.2.8. mono	12
2.2.9. sans	12
2.2.10. script	12
2.2.11. serif	13
2.2.12. sscript	13
2.2.13. upright	13
2.2.14. dif	14
2.2.15. Dif	14

1. Overview

Mathyaml converts your typst equations to MathML Core so that they work well with HTML export. MathML Core is a language for describing mathematical notation and supported by major browsers (firefox and chrome). You can find an overview of MathML on [mdn](#) and in the [specification](#).


Note that MathML rendering is certainly not perfect, some features work better and some worse. In general the output tends to look much better in firefox than chrome. See the [section about missing/ non-working features](#).

MathML Core is not complete and can't render everything itself. Instead it relies on Web Platform features (such as CSS) (see the [explainer](#)). [Here](#) is a list of polyfills/ features used that do not come from MathML Core.

2. Quickstart


First, import mathyaml and include the prelude, which defines replacements for elements which mathyaml can't handle (e.g. bold or cal).

```
#import "@local/mathyaml:0.1.0"
#import mathyaml: to-mathml
#import mathyaml.prelude: *
```

 Typst

Include the required stylesheet (and the mathfont):


```
#mathyaml.stylesheets()
```

 Typst

Note that the mathfont is required, else the rendering looks really bad. The font is currently downloaded from [github](#). I would recommend changing the font family to your liking and downloading the css files yourself (so that it works without an internet connection).


Convert equations manually (but only for html):

```
The fraction #to-mathml($1/3$) is not a decimal number. And we know
#to-mathml($ a^2 + b^2 = c^2. $)
```

 Typst

You can also convert equations automatically. If this panics, try try-to-mathml instead, which will create a svg on error.

```
#show math.equation: to-mathml
```

 Typst

```
To solve the cubic equation $t^3 + p t + q = 0$ (where the real numbers
$p, q$ satisfy $4p^3 + 27q^2 > 0$) one can use Cardano's formula:
```

```
$
```

```
root(3, -q/2 + sqrt(q^2/4 + p^3/27)) + root(3, -q/2 - sqrt(q^2/4 +
p^3/27)).
```

```
$
```

3. Missing/ non-working features

The list below comes from own testing and typst's test suite. I've indicated the corresponding test names in parentheses. [Here](#) you can see the html output for the examples and tests.

Not working/ looking not perfect

set and show rules Examples:

- `#show math.equation: set align(start)`
- `#show math.equation: set text(font: "STIX Two Math")`

(E.g. issue-3973-math-equation-align, math-attach-show-limit, math-cases-delim, math-equation-show-rule, math-mat-delim-set, math-mat-augment-set, math-vec-delim-set)

non-math elements generally not supported

cancel (just not implemented)

multiline break in text (issue-1948-math-text-break)

weak spacing (math-lr-weak-spacing)

equation numbering and labels (math-equation-align-numbered)

rtl (issue-3696-equation-rtl)

semantics mathml allows adding svg annotations (<https://www.w3.org/TR/mathml-core/#semantics-and-presentation>). Typst html elements may not contain hyphens, so it is currently not possible to create an `annotation-xml` element.

accents

- chrome:
 - it generally seems the chrome does not recognize the width of the accents, so they also collide and are offset (math-accent-high-base, math-accent-sym-call)
 - dotless (math-accent-dotless)
- size does not work

alignment

- nested alignments are unsupported
- chrome: sometimes chrome does not align the columns correctly (math-align-toggle, math-align-wider-first-column) (does not respect `text-align: right;`)

attach

- `t` and `b` attachments are further away (math-attach-mixed, math-attach-subscript-multiline)
- scripts are limits sometimes (math-op-scripts-vs-limits) (I am not sure to improve this, as `displaystyle` should be enough)
- limits are scripts sometimes (math-accent-wide-base, math-attach-limit-long) (set `movablelimits="false"` for the next inner `mo` in `limits`)
- attached text is not rendered as stack above but at the top right (math-stretch-attach-nested-equation)
- nested attaches are not merged (math-attach-nested-deep-base)
- attaches are not stretched automatically

lr/ mid

- the size parameter does not work
- firefox: `mid` is sometimes not completely large enough (math-lr-mid)

- firefox: lr does not include subscript (math-lr-unparen)

mat

- chrome: gap does not work (math-mat-gaps)
- augment colors (math-mat-augment)
- linebreaks are not discarded (math-mat-linebreaks)
- manual alignment does not work (math-mat-align)

vec

- manual alignment does not work (math-vec-align-explicit-alternating)

primes chrome: they are really small (math-primes-attach)

sqrt chrome: small artifacts (math-root-large-body, math-root-large-index)

sizes (display, inline, script, sscript)

- only two levels (math-size)
- only available via prelude
- cramped is not supported

variants (serif, sans, frak, mono, bb, cal)

- only available via prelude

styles (upright, italic, bold)

- only available via prelude
- dotless styles don't really work (math-style-dotless)

spacing MathML adds extra spaces (math-spacing-kept-spaces) (not too bad)

stretch

- only supports ops
- only works (horizontally) with an element above/ below (see math-stretch-complex, math-stretch-horizontal, math-underover-brace)

overset multiline in overbrace looks weird, it has extra space

List of polyfills/ features used that do not come from MathML Core

- some CSS for inserting html frames inline or as a block
- some CSS for alignment points
- cases: uses CSS padding-bottom for gap
- vec: uses CSS padding-bottom for gap, text-align for align.
- mat:
 - uses CSS padding-bottom and padding-right for gap, text-align for align.
 - uses CSS (border-right and border-bottom) for augments
- primes: uses CSS (padding-left)

2. Reference

2.1. lib

2.1.1. include-mathfont

Include a math font in the html document.

See <https://github.com/fred-wang/MathFonts> for a list of supported fonts.

Parameters

```
include-mathfont(font: auto str)
```

font auto or str

The name of the font to include.

Default: auto

2.1.2. maybe-html

Transform the content if html-export is active.

Returns the content as-is, if html is not active.

Parameters

```
maybe-html(  
  transform: function,  
  inner: content any,  
  force: bool,  
  ..args: arguments  
) -> content
```

transform function

This function which will be called to transform the content.

The first parameter of the function will be `maybe-html.inner`, the other parameters are `maybe-html.args`.

inner content or any

The content to transform.

force bool

Controls if the transform function should be called even on paged targets (pdf, png etc.).

Default: false

..args arguments

Extra arguments to pass to [maybe-html.transform](#).

2.1.3. **stylesheets**

Include the required/ recommended stylesheets.

Parameters

`stylesheets(include-fonts: bool)`

include-fonts bool

Whether to include a math font in the html document. See [include-mathfont\(\)](#).

Default: `true`

2.1.4. **to-mathml**

Convert the inner body to MathML and panic on error.

If you want to embed a svg-frame on error instead, use [try-to-mathml\(\)](#). If you want to handle errors yourself, use [to-mathml-raw\(\)](#).

Note that for non-html targets, this function does nothing (you can change this with [to-mathml.force](#)).

Parameters

```
to-mathml(
  inner: content math.equation,
  block: bool auto,
  force: bool
) -> content
```

inner content or math.equation

The equation/ content to convert.

block bool or auto

Whether the equation is rendered as a separate block.

If block is auto, it will be inferred from the input.

Default: `auto`

force `bool`

Whether to build an html tree even on paged targets (pdf, png etc.).

Default: `false`

2.1.5. to-mathml-raw

Convert content to MathML.

Parameters

```
to-mathml-raw(
  inner: content math.equation ,
  block: bool auto ,
  on-error: function ,
  on-warn: function ,
  is-error: function
) -> content
```

inner `content` or `math.equation`

The equation/ content to convert.

block `bool` or `auto`

Whether the equation is rendered as a separate block.

If block is auto, it will be inferred from the input.

Default: `auto`

on-error `function`

This callback will be called with every error.

The function should take a single argument sink as parameter. If you overwrite this parameter, don't forget to overwrite `to-mathml-raw.is-error`. is-error should return true if and only if on-error was called.

For example you could return a custom dictionary on each error and check in is-error for that.

Default: `panic`

on-warn **function**

This function will be called with every warning.

The function should take a single argument sink as parameter. If you overwrite this parameter, don't forget to overwrite [to-mathml-raw.is-error](#). In combination with `is-error`, you can panic on the warning, silence it or propagate it.

Default: `(..args) => ()`

is-error **function**

This callback will be called to determine if a result is an error.

Errors will be propagated directly.

Default: `res => false`

2.1.6. try-to-mathml

Try to convert the inner body to MathML, but fallback to a svg frame on error.

Note that for non-html targets, this function does nothing (you can change this with [try-to-mathml.force](#)).

Parameters

```
try-to-mathml(
  inner: content math.equation,
  block: bool auto,
  strict: bool,
  force: bool
) -> content
```

inner **content** or `math.equation`

The equation/ content to convert.

block **bool** or **auto**

Whether the equation is rendered as a separate block.

If block is auto, it will be inferred from the input.

Default: `auto`

strict `bool`

Whether to consider warnings as errors.

Default: `false`

force `bool`

Whether to build an html tree even on paged targets (pdf, png etc.).

Default: `false`

2.2. prelude

2.2.1. **bb**

Blackboard bold (double-struck) font style in math.

For uppercase latin letters, blackboard bold is additionally available through symbols of the form \mathbb{N} and \mathbb{R} .

Parameters

`bb`(`body`: `content`)

body `content`

The content to style.

2.2.2. **bold**

Bold font style in math.

Parameters

`bold`(`body`: `content`)

body `content`

The content to style.

2.2.3. **cal**

Calligraphic font style in math.

Parameters

`cal`(`body`: `content`)

body `content`

The content to style.

2.2.4. **display**

Forced display style in math.

This is the normal size for block equations.

Parameters

`display`(`body`: `content`)

body `content`

The content to size

2.2.5. **frak**

Fraktur font style in math.

Parameters

`frak`(`body`: `content`)

body `content`

The content to style.

2.2.6. **inline**

Forced inline (text) style in math.

This is the normal size for inline equations.

Parameters

`inline`(`body`: `content`)

body `content`

The content to size

2.2.7. **italic**

Italic font style in math.

For roman letters and greek lowercase letters, this is already the default.

Parameters

`italic(body: content)`

body `content`

The content to style.

2.2.8. **mono**

Monospace font style in math.

Parameters

`mono(body: content)`

body `content`

The content to style.

2.2.9. **sans**

Sans-serif font style in math.

Parameters

`sans(body: content)`

body `content`

The content to style.

2.2.10. **script**

Forced script style in math.

This is the smaller size used in powers or sub- or superscripts.

Parameters

`script(body: content)`

body `content`

The content to size

2.2.11. serif

Serif (roman) font style in math. This is already the default.

Parameters

`serif`(`body`: `content`)

body `content`

The content to style.

2.2.12. sscript

Forced second script style in math.

This is the smallest size, used in second-level sub- and superscripts (script of the script).

Note that this currently is the same as `script`.

Parameters

`sscript`(`body`: `content`)

body `content`

The content to size

2.2.13. upright

Upright (non-italic) font style in math.

Parameters

`upright`(`body`: `content`)

body `content`

The content to style.

2.2.14. dif

dif symbol.

2.2.15. Dif

Dif symbol.