MathJax TeX and LaTeX Support

The support for *TeX* and *LaTeX* in MathJax consists of two parts: the *tex2jax* preprocessor, and the *TeX* input processor. The first of these looks for mathematics within your web page (indicated by math delimiters like `$$...$$`) and marks the mathematics for later processing by MathJax. The TeX input processor is what converts the TeX notation into MathJax's internal format, where one of MathJax's output processors then displays it in the web page.

The *tex2jax* preprocessor can be configured to look for whatever markers you want to use for your math delimiters. See the *tex2jax configuration options* section for details on how to customize the action of *tex2jax*.

The TeX input processor handles conversion of your mathematical notation into MathJax's internal format (which is essentially MathML), and so acts as a TeX to MathML converter. The TeX input processor has few configuration options (see the *TeX options* section for details), but it can also be customized through the use of extensions that define additional functionality (see the *TeX and LaTeX extensions* below).

Note that the TeX input processor implements **only** the math-mode macros of TeX and LaTeX, not the text-mode macros. MathJax expects that you will use standard HTML tags to handle formatting the text of your page; it only handles the mathematics. So, for example, MathJax does not implement `\emph` or `\begin{enumerate}...\end{enumerate}` or other text-mode macros or environments. You must use HTML to handle such formatting tasks. If you need a LaTeX-to-HTML converter, you should consider other options.

TeX and LaTeX math delimiters

By default, the *tex2jax* preprocessor defines the LaTeX math delimiters, which are `\(...\)` for in-line math, and `\[...\]` for displayed equations. It also defines the TeX delimiters `$$...$$` for displayed equations, but it does **not** define `$...$` as in-line math delimiters. That is because dollar signs appear too often in non-mathematical settings, which could cause some text to be treated as mathematics unexpectedly. For example, with single-dollar delimiters, "... the cost is $2.50 for the first one, and $2.00 for each additional one ..." would

cause the phrase "2.50 for the first one, and" to be treated as mathematics since it falls between dollar signs. For this reason, if you want to use single-dollars for in-line math mode, you must enable that explicitly in your configuration:

```
MathJax.Hub.Config({
  tex2jax: {
    inlineMath: [['$','$'], ['\\(','\\)']],
    processEscapes: true
  }
});
```

Note that if you do this, you may want to also set `processEscapes` to `true`, as in the example above, so that you can use `\$` to prevent a dollar sign from being treated as a math delimiter within the text of your web page. (Note that within TeX mathematics, `\$` always has this meaning; `processEscapes` only affects the treatment of the *opening* math delimiter.)

See the `config/default.js` file, or the *tex2jax configuration options* page, for additional configuration parameters that you can specify for the *tex2jax* preprocessor, which is the component of MathJax that identifies TeX notation within the page.

TeX and LaTeX in HTML documents

Keep in mind that your mathematics is part of an HTML document, so you need to be aware of the special characters used by HTML as part of its markup. There cannot be HTML tags within the math delimiters (other than `<br>`) as TeX-formatted math does not include HTML tags. Also, since the mathematics is initially given as text on the page, you need to be careful that your mathematics doesn't look like HTML tags to the browser (which parses the page before MathJax gets to see it). In particular, that means that you have to be careful about things like less-than and greater-than signs (`<` and `>`), and ampersands (`&`), which have special meaning to the browsers. For example,

```
... when $x<y$ we have ...
```

will cause a problem, because the brower will think `<y` is the beginning of a tag named `y` (even though there is no such tag in HTML). When this happens, the browser will think the tag continues up to the next `>` in the document (typically the end of the next actual tag in the HTML file), and you may notice that you are missing part of the text of the document. In the example above, the "`we have ...`" will not be displayed because the browser thinks it is part of the tag starting at `<y`. This is one indication you can use to spot this problem; it is a common error and should be avoided.

Usually, it is sufficient to simply put spaces around these symbols to cause the browser to avoid them, so

```
... when $x < y$ we have ...
```

should work. Alternatively, you can use the HTML entities `&lt;`, `&gt;` and `&amp;` to encode these characters so that the browser will not interpret them, but MathJax will. E.g.,

```
... when $x &lt; y$ we have ...
```

Finally, there are `\lt` and `\gt` macros defined to make it easier to enter `<` and `>` using TeX-like syntax:

```
... when $x \lt y$ we have ...
```

Keep in mind that the browser interprets your text before MathJax does.

Another source of difficulty is when MathJax is used in content management systems that have their own document processing commands that are interpreted before the HTML page is created. For example, many blogs and wikis use formats like *Markdown* to allow you to create the content of you pages. In Markdown, the underscore is used to indicate italics, and this usage will conflict with MathJax's use of the underscore to indicate a subscript. Since Markdown is applied to the page first, it will convert your subscripts markers into italics (inserting `<i>` tags into your mathematics, which will cause MathJax to ignore the math).

Such systems need to be told not to modify the mathematics that appears between math delimiters. That usually involves modifying the content-management system itself, which is beyond the means of most page authors. If you are lucky, someone else will already have done this for you, and you can find a MathJax plugin for your system on the MathJax-In-Use page page.

If there is no plugin for your system, or if it doesn't handle the subtleties of issolating the mathematics from the other markup that it supports, then you may have to "trick" it into leaving your mathematics untouched. Most content-management systems provide some means of indicating text that should not be modified ("verbatim" text), often for giving code snippets for computer languages. You may be use that to enclose your mathematics so that the system leaves it unchanged and MathJax can process it. For example, in Markdown, the back-tick (`` ` ``) is used to mark verbatim text, so

```
... we have `\(x_1 = 132\)` and `\(x_2 = 370\)` and so ...
```

may be able to protect the underscores from being processed by Markdown.

Some content-management systems use the backslash (`\`) as a special character for "escaping" other characters, but TeX uses this character to indicate a macro name. In such systems, you may have to double the backslashes in order to obtain a single backslash in your HTML page. For example, you may have to do

```
\\begin{array}{cc}
  a & b \\\\
  c & c
\\end{array}
```

to get an array with the four entries *a*, *b*, *c*, and *d*. Note in particular that if you want `\\` you will have to double *both* backslashes, giving `\\\\`.

Finally, if you have enabled single dollar-signs as math delimiters, and you want to include a literal dollar sign in your web page (one that doesn't represent a math delimiter), you will need to prevent MathJax from using it as a math delimiter. If you also enable the

`processEscapes` configuration parameter, then you can use `\$` in the text of your page to get a dollar sign (without the backslash) in the end. Alternatively, you use something like `<span>$</span>` to isolate the dollar sign so that MathJax will not use it as a delimiter.

Defining TeX macros

You can use the `\def`, `\newcommand`, `\renewcommand`, `\newenvironment`, `\renewenvironment`, and `\let` commands to create your own macros and environments. Unlike actual TeX, however, in order for MathJax to process these, they must be enclosed in math delimiters (since MathJax only processes macros in math-mode). For example

```
\(
   \def\RR{\bf R}
   \def\bold#1{\bf #1}
\)
```

would define `\RR` to produce a bold-faced "R", and `\bold{...}` to put its argument into bold face. Both definitions would be available throughout the rest of the page.

You can include macro definitions in the *Macros* section of the *TeX* blocks of your configuration, but they must be represented as JavaScript objects. For example, the two macros above can be pre-defined in the configuration by

```
MathJax.Hub.Config({
  TeX: {
    Macros: {
      RR: "{\\bf R}",
      bold: ["{\\bf #1}",1]
    }
  }
});
```

Here you give the macro as a *name:value* pair, where the *name* is the name of the control sequence (without the backslash) that you are defining, and *value* is either the replacement string for the macro (when there are no arguments) or an array consisting of the replacement string followed by the number of arguments for the macro.

Note that the replacement string is given as a JavaScript string literal, and the backslash has special meaning in JavaScript strings. So to get an actual backslash in the string you must double it, as in the examples above.

If you have many such definitions that you want to use on more than one page, you could put them into a configuration file that you can load along with the main configuration file. For example, you could create a file in `MathJax/config/local` called `local.js` that contains your macro definitions:

```
MathJax.Hub.Config({
  TeX: {
    Macros: {
      RR: "{\\bf R}",
      bold: ["{\\bf #1}",1]
    }
  }
});

MathJax.Ajax.loadComplete("[MathJax]/config/local/local.js");
```

and then load it along with your main configuration file on the script that loads `MathJax.js`:

```
<script src="/MathJax/MathJax.js?config=TeX-AMS_HTML,local/local.js"></script>
```

If you are using the CDN, you can make a local configuration file on your own server, and load MathJax itself from the CDN and your configuration file from your server. See *Using a Local Configuration File with the CDN* for details.

Automatic Equation Numbering

New in MathJax v2.0 is the ability to have equations be numbered automatically. This functionality is turned off by default, so that pages don't change when you update from v1.1 to v2.0, but it is easy to configure MathJax to produce automatic equation numbers by adding:

```
<script type="text/x-mathjax-config">
MathJax.Hub.Config({
  TeX: { equationNumbers: { autoNumber: "AMS" } }
});
</script>
```

to your page just before the `<script>` tag that loads `MathJax.js` itself.

Equations can be numbered in two ways: either number the AMSmath environments as LaTeX would, or number all displayed equations (the example above uses AMS-style numbering). Set `autoNumber` to `"all"` if you want every displayed equation to be numbered. You can use `\notag` or `\nonumber` to prevent individual equations from being numbered, and `\tag{}` can be used to override the usual equation number with your own symbol instead.

Note that the AMS environments come in two forms: starred and unstarred. The unstarred versions produce equation numbers (when `autoNumber` is set to `"AMS"`) and the starred ones don't. For example

```
\begin{equation}
   E = mc^2
\end{equation}
```

will be numbered, while

```
\begin{equation*}
   e^{\pi i} + 1 = 0
\end{equation*}
```

won't be numbered (when `autoNumber` is `"AMS"`).

You can use `\label` to give an equation an identifier that you can use to refer to it later, and then use `\ref` or `\eqref` within your document to insert the actual equation number at that location, as a reference. For example,

```
In equation \eqref{eq:sample}, we find the value of an
interesting integral:

\begin{equation}
  \int_0^\infty \frac{x^3}{e^x-1}\,dx = \frac{\pi^4}{15}
  \label{eq:sample}
\end{equation}
```

includes a labeled equation and a reference to that equation. Note that references can come before the corresponding formula as well as after them. See the equation numbering links in the MathJax examples page for more examples.

You can configure the way that numbers are displayed and how the references to them are made using parameters in the `equationNumbers` block of your `TeX` configuration. See the *TeX configuration options* page for more details.

TeX and LaTeX extensions

While MathJax includes nearly all of the Plain TeX math macros, and many of the LaTeX macros and environments, not everything is implemented in the core TeX input processor. Some less-used commands are defined in extensions to the TeX processor. MathJax will load some extensions automatically when you first use the commands they implement (for example, the `\def` and `\newcommand` macros are implemented in the `newcommand.js` extension, but MathJax loads this extension itself when you use those macros). Not all extensions are set up to load automatically, however, so you may need to request some extensions explicitly yourself.

To enable any of the TeX extensions, simply add the appropriate string (e.g., `"AMSmath.js"`) to the *extensions* array in the `TeX` block of your configuration. If you use one of the combined configuration files, like `TeX-AMS_HTML`, this will already include several of the extensions automatically, but you can include others using a mathjax configuration script prior to loading MathJax. For example

```
<script type="text/x-mathjax-config">
  MathJax.Hub.Config({ TeX: { extensions: ["autobold.js"] }});
</script>
<script type="text/javascript"
    src="http://cdn.mathjax.org/mathjax/latest/MathJax.js?config=TeX-AMS_HTML">
</script>
```

will load the *autobold* TeX extension in addition to those already included in the `TeX-AMS_HTML` configuration file.

You can also load these extensions from within a math expresion using the non-standard `\require{extension}` macro. For example

```
\(\require{color}\)
```

would load the *color* extension into the page. This way you you can load extensions into pages that didn't load them in their configurations (and prevents you from having to load all the extensions into all pages even if they aren't used).

It is also possible to create a macro that will autoload an extension when it is first used (under the assumption that the extension will redefine it to perform its true function). For example

```
<script type="text/x-mathjax-config">
MathJax.Hub.Register.StartupHook("TeX Jax Ready",function () {
  MathJax.Hub.Insert(MathJax.InputJax.TeX.Definitions.macros,{
    cancel: ["Extension","cancel"],
    bcancel: ["Extension","cancel"],
    xcancel: ["Extension","cancel"],
    cancelto: ["Extension","cancel"]
  });
});
</script>
```

would declare the `\cancel`, `\bcancel`, `\xcancel`, and `\cancelto` macros to load the *cancel* extension (where they are actually defined). Whichever is used first will cause the extension to be loaded, redefining all four to their proper values. Note that this may be better than loading the extension explicitly, since it avoids loading the extra file on pages where these macros are *not* used. The sample autoloading macros example page shows this in action. The *autoload-all* extension below defines such macros for *all* the extensions so that if you include it, MathJax will have access to all the macros it knows about.

The main extensions are described below.

Action

The *action* extension gives you access to the MathML `<maction>` element. It defines three new non-standard macros:

`\mathtip{math}{tip}`

Use `tip` (in math mode) as tooltip for `math`.

`\texttip{math}{tip}`

Use `tip` (in text mode) as tooltip for `math`.

`\toggle{math1}{math2}...\endtoggle`

Show `math1`, and when clicked, show `math2`, and so on. When the last one is clicked, go back to math1.

To use this extension in your own configurations, add it to the *extensions* array in the TeX block.

```
TeX: {
  extensions: ["action.js"]
}
```

This extension is **not** included in any of the combined configurations, and will not be loaded automatically, so you must include it explicitly in your configuration if you wish to use these commands.

AMSmath and AMSsymbols

The *AMSmath* extension implements AMS math environments and macros, and the *AMSsymbols* extension implements macros for accessing the AMS symbol fonts. These are already included in the combined configuration files that load the TeX input processor. To use these extensions in your own configurations, add them to the *extensions* array in the TeX block.

```
TeX: {
  extensions: ["AMSmath.js", "AMSsymbols.js", ...]
}
```

See the list of control sequences at the end of this document for details about what commands are implemented in these extensions.

If you are not using one of the combined configuration files, the *AMSmath* extension will be loaded automatically when you first use one of the math environments it defines, but you will have to load it explicitly if you want to use the other macros that it defines. The *AMSsymbols* extension is not loaded automatically, so you must include it explicitly if you want to use the macros it defines.

Both extensions are included in all the combined configuration files that load the TeX input processor.

AMScd

The *AMScd* extensions implements the *CD* environment for commutative diagrams. See the AMScd guide for more information on how to use the *CD* environment.

To use this extension in your own configurations, add it to the *extensions* array in the TeX block.

```
TeX: {
  extensions: ["AMScd.js"]
}
```

Alternatively, if the extension hasn't been loaded in the configuration, you can use `\require{AMScd}` to load it from within a TeX expression. Note that you only need to include this once on the page, not every time the *CD* environment is used.

This extension is **not** included in any of the combined configurations, and will not be loaded automatically, so you must include it explicitly in your configuration if you wish to use these commands.

Autobold

The *autobold* extension adds `\boldsymbol{...}` around mathematics that appears in a section of an HTML page that is in bold.

```
TeX: {
  extensions: ["autobold.js"]
}
```

This extension is **not** loaded by the combined configuration files.

BBox

The *bbox* extension defines a new macro for adding background colors, borders, and padding to your math expressions.

`\bbox[options]{math}`

puts a bounding box around **`math`** using the provided **`options`**. The options can be one of the following:

1. A color name used for the background color.
2. A dimension (e.g., **`2px`**) to be used as a padding around the mathematics (on all sides).
3. Style attributes to be applied to the mathematics (e.g., **`border:1px solid red`**).
4. A combination of these separated by commas.

Here are some examples:

```
\bbox[red]{x+y}      % a red box behind x+y
\bbox[2pt]{x+1}      % an invisible box around x+y with 2pt of extra space
\bbox[red,2pt]{x+1}  % a red box around x+y with 2pt of extra space
\bbox[5px,border:2px solid red]
                     % a 2px red border around the math 5px away
```

This extension is **not** included in any of the combined configurations, but it will be loaded automatically, so you do not need to include it in your *extensions* array.

Begingroup

The *begingroup* extension implements commands that provide a mechanism for localizing macro defintions so that they are not permanent. This is useful if you have a blog site, for example, and want to isolate changes that your readers make in their comments so that they don't affect later comments.

It defines two new non-standard macros, `\begingroup` and `\endgroup`, that are used to start and stop a local namespace for macros. Any macros that are defined between the `\begingroup` and `\endgroup` will be removed after the `\endgroup` is executed. For example, if you put `\(\begingroup\)` at the top of each reader's comments and `\(\endgroup\)` at the end, then any macros they define within their response will be removed after it is processed.

In addition to these two macros, the *begingroup* extension defines the standard `\global` and `\gdef` control sequences from TeX. (The `\let`, `\def`, `\newcommand`, and `\newenvironment` control sequences are already defined in the core TeX input jax.)

To use this extension in your own configurations, add it to the *extensions* array in the TeX block.

```
TeX: {
  extensions: ["begingroup.js"]
}
```

This extension is **not** included in any of the combined configurations, and will not be loaded automatically, so you must include it explicitly in your configuration if you wish to use these commands.

Cancel

The *cancel* extension defines the following macros:

`\cancel{math}`

Strikeout **`math`** from lower left to upper right.

`\bcancel{math}`

Strikeout **`math`** from upper left to lower right.

`\xcancel{math}`

Strikeout `math` with an "X".

`\cancelto{value}{math}`

Strikeout `math` with an arrow going to `value`.

To use this extension in your own configurations, add it to the *extensions* array in the TeX block.

```
TeX: {
  extensions: ["cancel.js"]
}
```

This extension is **not** included in any of the combined configurations, and will not be loaded automatically, so you must include it explicitly in your configuration if you wish to use these commands.

Color

The `\color` command in the core TeX input jax is not standard in that it takes the mathematics to be colored as one of its parameters, whereas the LaTeX `\color` command is a switch that changes the color of everything that follows it.

The *color* extension changes the `\color` command to be compatible with the LaTeX implementation, and also defines `\colorbox`, `\fcolorbox`, and `\definecolor`, as in the LaTeX color package. It defines the standard set of colors (Apricot, Aquamarine, Bittersweet, and so on), and provides the RGB and grey-scale color spaces in addition to named colors.

To use this extension in your own configurations, add it to the *extensions* array in the TeX block.

```
TeX: {
  extensions: ["color.js"]
}
```

This extension is **not** included in any of the combined configurations, and will not be loaded automatically, so you must include it explicitly in your configuration if you wish to use these commands, and have `\color` be compatible with LaTeX usage.

Enclose

The *enclose* extension gives you access to the MathML `<menclose>` element for adding boxes, ovals, strikethroughs, and other marks over your mathematics. It defines the following non-standard macro:

`\enclose{notation}[attributes]{math}`

Where **`notation`** is a comma-separated list of MathML **`<menclose>`** notations (e.g., **`circle`**, **`left`**, **`updiagonalstrike`**, **`longdiv`**, etc.), **`attributes`** are MathML attribute values allowed on the **`<menclose>`** element (e.g., **`mathcolor="red"`**, **`mathbackground="yellow"`**), and **`math`** is the mathematics to be enclosed.

For example

```
\enclose{circle}[mathcolor="red"]{x}
\enclose{circle}[mathcolor="red"]{\color{black}{x}}
\enclose{circle,box}{x}
\enclose{circle}{\enclose{box}{x}}
```

To use this extension in your own configurations, add it to the *extensions* array in the TeX block.

```
TeX: {
  extensions: ["enclose.js"]
}
```

This extension is **not** included in any of the combined configurations, and will not be loaded automatically, so you must include it explicitly in your configuration if you wish to use these commands.

Extpfeil

The *extpfeil* extension adds more macros for producing extensible arrows, including `\xtwoheadrightarrow`, `\xtwoheadleftarrow`, `\xmapsto`, `\xlongequal`, `\xtofrom`, and a non-standard `\Newextarrow` for creating your own extensible arrows. The latter has the form

`\Newextarrow{\cs}{lspace,rspace}{unicode-char}`

where **`\cs`** is the new control sequence name to be defined, **`lspace`** and **`rspace`** are integers representing the amount of space (in suitably small units) to use at the left and right of text that is placed above or below the arrow, and **`unicode-char`** is a number representing a unicode character position in either decimal or hexadecimal notation.

For example

```
\Newextarrow{\xrightharpoonup}{5,10}{0x21C0}
```

defines an extensible right harpoon with barb up. Note that MathJax knows how to stretch only a limited number of characters, so you may not actually get a stretchy character this way.

To use this extension in your own configurations, add it to the *extensions* array in the TeX block.

```
TeX: {
  extensions: ["extpfeil.js"]
}
```

This extension is **not** included in any of the combined configurations, and will not be loaded automatically, so you must include it explicitly in your configuration if you wish to use these commands.

HTML

The *HTML* extension gives you access to some HTML features like styles, classes, element ID's and clickable links. It defines the following non-standard macros:

`\href{url}{math}`

Makes `math` be a link to the page given by `url`.

`\class{name}{math}`

Attaches the CSS class `name` to the output associated with `math` when it is included in the HTML page. This allows your CSS to style the element.

`\cssId{id}{math}`

Attaches an id attribute with value `id` to the output associated with `math` when it is included in the HTML page. This allows your CSS to style the element, or your javascript to locate it on the page.

`\style{css}{math}`

Adds the give `css` declarations to the element associated with `math`.

For example:

```
x \href{why-equal.html}{=} y^2 + 1

(x+1)^2 = \class{hidden}{(x+1)(x+1)}

(x+1)^2 = \cssId{step1}{\style{visibility:hidden}{(x+1)(x+1)}}
```

This extension is **not** included in any of the combined configurations, but it will be loaded automatically when any of these macros is used, so you do not need to include it explicitly in your configuration.

mhchem

The *mhchem* extensions implements the `\ce`, `\cf`, and `\cee` chemical equation macros of the LaTeX *mhchem* package. See the mhchem CPAN page for more information and a link to the documentation for *mhchem*.

For example

```
\ce{C6H5-CHO}
\ce{$A$ ->[\ce{+H2O}] $B$}
\ce{SO4^2- + Ba^2+ -> BaSO4 v}
```

To use this extension in your own configurations, add it to the *extensions* array in the TeX block.

```
TeX: {
  extensions: ["mhchem.js"]
}
```

This extension is **not** included in any of the combined configurations, and will not be loaded automatically, so you must include it explicitly in your configuration if you wish to use these commands.

noErrors

The *noErrors* extension prevents TeX error messages from being displayed and shows the original TeX code instead. You can configure whether the dollar signs are shown or not for in-line math, and whether to put all the TeX on one line or use multiple lines (if the original text contained line breaks).

This extension is loaded by all the combined configuration files that include the TeX input processor. To enable the *noErrors* extension in your own configuration, or to modify its parameters, add something like the following to your `MathJax.Hub.Config()` call:

```
TeX: {
  extensions: ["noErrors.js"],
  noErrors: {
    inlineDelimiters: ["",""],   // or ["$","$"] or ["\\(","\\)"]
    multiLine: true,             // false for TeX on all one line
    style: {
      "font-size":   "90%",
      "text-align":  "left",
      "color":       "black",
      "padding":     "1px 3px",
      "border":      "1px solid"
      // add any additional CSS styles that you want
      //  (be sure there is no extra comma at the end of the last item)
    }
  }
}
```

Display-style math is always shown in multi-line format, and without delimiters, as it will already be set off in its own centered paragraph, like standard display mathematics.

The default settings place the invalid TeX in a multi-line box with a black border. If you want it to look as though the TeX is just part of the paragraph, use

```
TeX: {
  noErrors: {
    inlineDelimiters: ["$","$"],   // or ["","",""] or ["\\(","\\)"]
    multiLine: false,
    style: {
      "font-size": "normal",
      "border": ""
    }
  }
}
```

You may also wish to set the font family or other CSS values here.

If you are using a combined configuration file that loads the TeX input processor, it will also load the *noErrors* extension automatically. If you want to disable the *noErrors* extension so that you receive the normal TeX error messages, use the following configuration:

```
TeX: { noErrors: { disabled: true } }
```

Any math that includes errors will be replaced by an error message indicating what went wrong.

noUndefined

The *noUndefined* extension causes undefined control sequences to be shown as their macro names rather than generating error messages. So `$X_{\xxx}$` would display as an "X" with a subscript consisting of the text `\xxx` in red.

This extension is loaded by all the combined configuration files that include the TeX input processor. To enable the *noUndefined* extension in your own configuration, or to modify its parameters, add something like the following to your `MathJax.Hub.Config()` call:

```
TeX: {
  extensions: ["noUndefined.js"],
  noUndefined: {
    attributes: {
      mathcolor: "red",
      mathbackground: "#FFEEEE",
      mathsize: "90%"
    }
  }
}
```

The `attributes` setting specifies attributes to apply to the `mtext` element that encodes the name of the undefined macro. The default values set `mathcolor` to `"red"`, but do not set any other attributes. This example sets the background to a light pink, and reduces the font size slightly.

If you are using a combined configuration file that loads the TeX input processor, it will also load the *noUndefined* extension automatically. If you want to disable the *noUndefined* extension so that you receive the normal TeX error messages for undefined macros, use the following configuration:

```
TeX: { noUndefined: { disabled: true } }
```

Any math that includes an undefined control sequence name will be replaced by an error message indicating what name was undefined.

Unicode support

The *unicode* extension implements a `\unicode{}` extension to TeX that allows arbitrary unicode code points to be entered in your mathematics. You can specify the height and depth of the character (the width is determined by the browser), and the default font from which to take the character.

Examples:

```
\unicode{65}                          % the character 'A'
\unicode{x41}                         % the character 'A'
\unicode[.55,0.05]{x22D6}             % less-than with dot, with height .55em and depth
0.05em
\unicode[.55,0.05][Geramond]{x22D6}   % same taken from Geramond font
\unicode[Garamond]{x22D6}             % same, but with default height, depth of
.8em,.2em
```

Once a size and font are provided for a given unicode point, they need not be specified again in subsequent `\unicode{}` calls for that character.

The result of `\unicode{...}` will have TeX class *ORD* (i.e., it will act like a variable). Use `\mathbin{...}`, `\mathrel{...}`, etc., to specify a different class.

Note that a font list can be given in the `\unicode{}` macro, but Internet Explorer has a buggy implementation of the `font-family` CSS attribute where it only looks in the first font in the list that is actually installed on the system, and if the required glyph is not in that font, it does not look at later fonts, but goes directly to the default font as set in the *Internet-Options/Font* panel. For this reason, the default font list for the `\unicode{}` macro is `STIXGeneral, 'Arial Unicode MS'`, so if the user has *STIX* fonts, the symbol will be taken from that (almost all the symbols are in *STIXGeneral*), otherwise MathJax tries *Arial Unicode MS*.

The *unicode* extension is loaded automatically when you first use the `\unicode{}` macro, so you do not need to add it to the *extensions* array. You can configure the extension as follows:

```
TeX: {
  unicode: {
    fonts: "STIXGeneral, 'Arial Unicode MS'"
  }
}
```

Autoload-all

The *autoload-all* extension predefines all the macros from the extensions above so that they autoload the extensions when first used. A number of macros already do this, e.g., `\unicode`, but this extension defines the others to do the same. That way MathJax will have access to all the macros that it knows about.

To use this extension in your own configurations, add it to the *extensions* array in the TeX block.

```
TeX: {
  extensions: ["autoload-all.js"]
}
```

This extension is **not** included in any of the combined configurations, and will not be loaded automatically, so you must include it explicitly in your configuration if you wish to use these commands.

Note that *autoload-all* redefines `\color` to be the one from the *color* extension (the LaTeX-compatible one rather than the non-standard MathJax version). This is because `\colorbox` and `\fcolorbox` autoload the *color* extension, which will cause `\color` to be redefined, and so for consistency, `\color` is redefined immediately.

If you wish to retain the original definition of `\color`, then use the following

```
<script type="text/x-mathjax-config">
MathJax.Hub.Config({
  TeX: { extensions: ["autoload-all.js"] }
});
MathJax.Hub.Register.StartupHook("TeX autoload-all Ready", function () {
  var MACROS = MathJax.InputJax.TeX.Definitions.macros;
  MACROS.color = "Color";
  delete MACROS.colorbox;
  delete MACROS.fcolorbox;
});
</script>
```

Supported LaTeX commands

This is a long list of the TeX macros supported by MathJax. If the macro is defined in an extension, the name of the extension follows the macro name. If the extension is in brackets, the extension will be loaded automatically when the macro or environment is first used.

More complete details about how to use these macros, with examples and explanations, is available at Carol Fisher's TeX Commands Available in MathJax page.

Symbols

```
#
%
&
^
_
{
}
~
'

\   (backslash-space)
\!
\#
\$
\%
\&
\,
\:
\;
\>
\\
\_
\{
\|
\}
```

A

\above		
\abovewithdelims		
\acute		
\aleph		
\alpha		
\amalg		
\And		
\angle		
\approx		
\approxeq	AMSsymbols	
\arccos		
\arcsin		
\arctan		
\arg		
\array		
\Arrowvert		
\arrowvert		
\ast		
\asymp		
\atop		
\atopwithdelims		

B

\backepsilon	AMSsymbols	
\backprime	AMSsymbols	
\backsim	AMSsymbols	
\backsimeq	AMSsymbols	
\backslash		
\backslash		
\bar		
\barwedge	AMSsymbols	
\Bbb		
\Bbbk	AMSsymbols	
\bbox	[bbox]	
\bcancel	cancel	
\because	AMSsymbols	
\begin		
\begingroup	begingroup	non-standard
\beta		
\beth	AMSsymbols	
\between	AMSsymbols	
\bf		
\Big		
\big		
\bigcap		
\bigcirc		
\bigcup		

\Bigg
\bigg
\Biggl
\biggl
\Biggm
\biggm
\Biggr
\biggr
\Bigl
\bigl
\Bigm
\bigm
\bigodot
\bigoplus
\bigotimes
\Bigr
\bigr
\bigsqcup
\bigstar AMSsymbols
\bigtriangledown
\bigtriangleup
\biguplus
\bigvee
\bigwedge
\binom AMSmath
\blacklozenge AMSsymbols
\blacksquare AMSsymbols
\blacktriangle AMSsymbols
\blacktriangledown AMSsymbols
\blacktriangleleft AMSsymbols
\blacktriangleright AMSsymbols
\bmod
\boldsymbol [boldsymbol]
\bot
\bowtie
\Box AMSsymbols
\boxdot AMSsymbols
\boxed AMSmath
\boxminus AMSsymbols
\boxplus AMSsymbols
\boxtimes AMSsymbols
\brace
\bracevert
\brack
\breve
\buildrel
\bullet
\Bumpeq AMSsymbols
\bumpeq AMSsymbols

C

\cal		
\cancel	cancel	
\cancelto	cancel	
\cap		
\Cap	AMSsymbols	
\cases		
\cdot		
\cdotp		
\cdots		
\ce	mhchem	
\cee	mhchem	
\centerdot	AMSsymbols	
\cf	mhchem	
\cfrac	AMSmath	
\check		
\checkmark	AMSsymbols	
\chi		
\choose		
\circ		
\circeq	AMSsymbols	
\circlearrowleft	AMSsymbols	
\circlearrowright	AMSsymbols	
\circledast	AMSsymbols	
\circledcirc	AMSsymbols	
\circleddash	AMSsymbols	
\circledR	AMSsymbols	
\circledS	AMSsymbols	
\class	[HTML]	non-standard
\clubsuit		
\colon		
\color	color	
\colorbox	color	
\complement	AMSsymbols	
\cong		
\coprod		
\cos		
\cosh		
\cot		
\coth		
\cr		
\csc		
\cssId	[HTML]	non-standard
\cup		
\Cup	AMSsymbols	
\curlyeqprec	AMSsymbols	
\curlyeqsucc	AMSsymbols	
\curlyvee	AMSsymbols	
\curlywedge	AMSsymbols	
\curvearrowleft	AMSsymbols	
\curvearrowright	AMSsymbols	

D

\dagger	
\daleth	AMSsymbols
\dashleftarrow	AMSsymbols
\dashrightarrow	AMSsymbols
\dashv	
\dbinom	AMSmath
\ddagger	
\ddddot	AMSmath
\dddot	AMSmath
\ddot	
\ddots	
\DeclareMathOperator	AMSmath
\definecolor	color
\def	[newcommand]
\deg	
\Delta	
\delta	
\det	
\dfrac	AMSmath
\diagdown	AMSsymbols
\diagup	AMSsymbols
\diamond	
\Diamond	AMSsymbols
\diamondsuit	
\digamma	AMSsymbols
\dim	
\displaylines	
\displaystyle	
\div	
\divideontimes	AMSsymbols
\dot	
\doteq	
\Doteq	AMSsymbols
\doteqdot	AMSsymbols
\dotplus	AMSsymbols
\dots	
\dotsb	
\dotsc	
\dotsi	
\dotsm	
\dotso	
\doublebarwedge	AMSsymbols
\doublecap	AMSsymbols
\doublecup	AMSsymbols
\Downarrow	
\downarrow	
\downdownarrows	AMSsymbols
\downharpoonleft	AMSsymbols
\downharpoonright	AMSsymbols

E

\ell		
\emptyset		
\enclose	enclose	non-standard
\end		
\endgroup	begingroup	non-standard
\enspace		
\epsilon		
\eqalign		
\eqalignno		
\eqcirc	AMSsymbols	
\eqref	[AMSmath]	
\eqsim	AMSsymbols	
\eqslantgtr	AMSsymbols	
\eqslantless	AMSsymbols	
\equiv		
\eta		
\eth	AMSsymbols	
\exists		
\exp		

F

\fallingdotseq	AMSsymbols	
\fbox		
\fcolorbox	color	
\Finv	AMSsymbols	
\flat		
\forall		
\frac		
\frac	AMSmath	
\frak		
\frown		

G

\Game	AMSsymbols
\Gamma	
\gamma	
\gcd	
\gdef	begingroup
\ge	
\genfrac	AMSmath
\geq	
\geqq	AMSsymbols
\geqslant	AMSsymbols
\gets	
\gg	
\ggg	AMSsymbols
\gggtr	AMSsymbols
\gimel	AMSsymbols
\global	begingroup
\gnapprox	AMSsymbols
\gneq	AMSsymbols
\gneqq	AMSsymbols
\gnsim	AMSsymbols
\grave	
\gt	
\gt	
\gtrapprox	AMSsymbols
\gtrdot	AMSsymbols
\gtreqless	AMSsymbols
\gtreqqless	AMSsymbols
\gtrless	AMSsymbols
\gtrsim	AMSsymbols
\gvertneqq	AMSsymbols

H

\hat	
\hbar	
\hbox	
\hdashline	
\heartsuit	
\hline	
\hom	
\hookleftarrow	
\hookrightarrow	
\hphantom	
\href	[HTML]
\hskip	
\hslash	AMSsymbols
\hspace	
\Huge	
\huge	
\idotsint	AMSmath

I

\iff	
\iiiint	AMSmath
\iiint	
\iint	
\Im	
\imath	
\impliedby	AMSsymbols
\implies	AMSsymbols
\in	
\inf	
\infty	
\injlim	AMSmath
\int	
\intercal	AMSsymbols
\intop	
\iota	
\it	

J

\jmath	
\Join	AMSsymbols

K

```
\kappa
\ker
\kern
```

L

```
\label                [AMSmath]
\Lambda
\lambda
\land
\langle
\LARGE
\Large
\large
\LaTeX
\lbrace
\lbrack
\lceil
\ldotp
\ldots
\le
\leadsto               AMSsymbols
\left
\Leftarrow
\leftarrow
\leftarrowtail         AMSsymbols
\leftharpoondown
\leftharpoonup
\leftleftarrows        AMSsymbols
\Leftrightarrow
\leftrightarrow
\leftrightarrows       AMSsymbols
\leftrightharpoons     AMSsymbols
\leftrightsquigarrow   AMSsymbols
\leftroot
\leftthreetimes        AMSsymbols
\leq
\leqalignno
\leqq                  AMSsymbols
\leqslant              AMSsymbols
\lessapprox            AMSsymbols
\lessdot               AMSsymbols
\lesseqgtr             AMSsymbols
\lesseqqgtr            AMSsymbols
\lessgtr               AMSsymbols
\lesssim               AMSsymbols
```

\let	[newcommand]
\lfloor	
\lg	
\lgroup	
\lhd	AMSsymbols
\lim	
\liminf	
\limits	
\limsup	
\ll	
\llap	
\llcorner	AMSsymbols
\Lleftarrow	AMSsymbols
\lll	AMSsymbols
\llless	AMSsymbols
\lmoustache	
\ln	
\lnapprox	AMSsymbols
\lneq	AMSsymbols
\lneqq	AMSsymbols
\lnot	
\lnsim	AMSsymbols
\log	
\Longleftarrow	
\longleftarrow	
\Longleftrightarrow	
\longleftrightarrow	
\longmapsto	
\Longrightarrow	
\longrightarrow	
\looparrowleft	AMSsymbols
\looparrowright	AMSsymbols
\lor	
\lower	
\lozenge	AMSsymbols
\lrcorner	AMSsymbols
\Lsh	AMSsymbols
\lt	
\lt	
\ltimes	AMSsymbols
\lVert	AMSmath
\lvert	AMSmath
\lvertneqq	AMSsymbols

M

\maltese	AMSsymbols	
\mapsto		
\mathbb		
\mathbf		
\mathbin		
\mathcal		
\mathchoice	[mathchoice]	
\mathclose		
\mathfrak		
\mathinner		
\mathit		
\mathop		
\mathopen		
\mathord		
\mathpunct		
\mathrel		
\mathring	AMSmath	
\mathrm		
\mathscr		
\mathsf		
\mathstrut		
\mathtip	action	non-standard
\mathtt		
\matrix		
\max		
\mbox		
\measuredangle	AMSsymbols	
\mho	AMSsymbols	
\mid		
\middle		
\min		
\mit		
\mkern		
\mmlToken		non-standard
\mod		
\models		
\moveleft		
\moveright		
\mp		
\mskip		
\mspace		
\mu		
\multimap	AMSsymbols	

N

\nabla		
\natural		

\ncong AMSsymbols
\ne
\nearrow
\neg
\negmedspace AMSmath
\negthickspace AMSmath
\negthinspace
\neq
\newcommand [newcommand]
\newenvironment [newcommand]
\Newextarrow extpfeil
\newline
\nexists AMSsymbols
\ngeq AMSsymbols
\ngeqq AMSsymbols
\ngeqslant AMSsymbols
\ngtr AMSsymbols
\ni
\nLeftarrow AMSsymbols
\nleftarrow AMSsymbols
\nLeftrightarrow AMSsymbols
\nleftrightarrow AMSsymbols
\nleq AMSsymbols
\nleqq AMSsymbols
\nleqslant AMSsymbols
\nless AMSsymbols
\nmid AMSsymbols
\nobreakspace AMSmath
\nolimits
\normalsize
\not
\notag [AMSmath]
\notin
\nparallel AMSsymbols
\nprec AMSsymbols
\npreceq AMSsymbols
\nRightarrow AMSsymbols
\nrightarrow AMSsymbols
\nshortmid AMSsymbols
\nshortparallel AMSsymbols
\nsim AMSsymbols
\nsubseteq AMSsymbols
\nsubseteqq AMSsymbols
\nsucc AMSsymbols
\nsucceq AMSsymbols
\nsupseteq AMSsymbols
\nsupseteqq AMSsymbols
\ntriangleleft AMSsymbols
\ntrianglelefteq AMSsymbols
\ntriangleright AMSsymbols
\ntrianglerighteq AMSsymbols
\nu
\nVDash AMSsymbols

\nVdash	AMSsymbols
\nvDash	AMSsymbols
\nvdash	AMSsymbols
\nwarrow	

O

\odot	
\oint	
\oldstyle	
\Omega	
\omega	
\omicron	
\ominus	
\operatorname	AMSmath
\oplus	
\oslash	
\otimes	
\over	
\overbrace	
\overleftarrow	
\overleftrightarrow	
\overline	
\overrightarrow	
\overset	
\overwithdelims	
\owns	

P

```
\parallel
\partial
\perp
\phantom
\Phi
\phi
\Pi
\pi
\pitchfork              AMSsymbols
\pm
\pmatrix
\pmb
\pmod
\pod
\Pr
\prec
\precapprox             AMSsymbols
\preccurlyeq            AMSsymbols
\preceq
\precnapprox            AMSsymbols
\precneqq               AMSsymbols
\precnsim               AMSsymbols
\precsim                AMSsymbols
\prime
\prod
\projlim                AMSmath
\propto
\Psi
\psi
```

Q

```
\qquad
\quad
```

R

\raise		
\rangle		
\rbrace		
\rbrack		
\rceil		
\Re		
\ref	[AMSmath]	
\renewcommand	[newcommand]	
\renewenvironment	[newcommand]	
\require		non-standard
\restriction	AMSsymbols	
\rfloor		
\rgroup		
\rhd	AMSsymbols	
\rho		
\right		
\Rightarrow		
\rightarrow		
\rightarrowtail	AMSsymbols	
\rightharpoondown		
\rightharpoonup		
\rightleftarrows	AMSsymbols	
\rightleftharpoons		
\rightleftharpoons	AMSsymbols	
\rightrightarrows	AMSsymbols	
\rightsquigarrow	AMSsymbols	
\rightthreetimes	AMSsymbols	
\risingdotseq	AMSsymbols	
\rlap		
\rm		
\rmoustache		
\root		
\Rrightarrow	AMSsymbols	
\Rsh	AMSsymbols	
\rtimes	AMSsymbols	
\Rule		non-standard
\rVert	AMSmath	
\rvert	AMSmath	

S

\S		
\scr		
\scriptscriptstyle		
\scriptsize		
\scriptstyle		
\searrow		
\sec		

\setminus		
\sf		
\sharp		
\shortmid	AMSsymbols	
\shortparallel	AMSsymbols	
\shoveleft	AMSmath	
\shoveright	AMSmath	
\sideset	AMSmath	
\Sigma		
\sigma		
\sim		
\simeq		
\sin		
\sinh		
\skew		
\small		
\smallfrown	AMSsymbols	
\smallint		
\smallsetminus	AMSsymbols	
\smallsmile	AMSsymbols	
\smash		
\smile		
\Space		
\space		
\spadesuit		
\sphericalangle	AMSsymbols	
\sqcap		
\sqcup		
\sqrt		
\sqsubset	AMSsymbols	
\sqsubseteq		
\sqsupset	AMSsymbols	
\sqsupseteq		
\square	AMSsymbols	
\stackrel		
\star		
\strut		
\style	[HTML]	non-stanard
\subset		
\Subset	AMSsymbols	
\subseteq		
\subseteqq	AMSsymbols	
\subsetneq	AMSsymbols	
\subsetneqq	AMSsymbols	
\substack	AMSmath	
\succ		
\succapprox	AMSsymbols	
\succcurlyeq	AMSsymbols	
\succeq		
\succnapprox	AMSsymbols	
\succneqq	AMSsymbols	
\succnsim	AMSsymbols	
\succsim	AMSsymbols	

\sum		
\sup		
\supset		
\Supset	AMSsymbols	
\supseteq		
\supseteqq	AMSsymbols	
\supsetneq	AMSsymbols	
\supsetneqq	AMSsymbols	
\surd		
\swarrow		

T

\tag	[AMSmath]	
\tan		
\tanh		
\tau		
\tbinom	AMSmath	
\TeX		
\text		
\textbf		
\textit		
\textrm		
\textstyle		
\texttip	action	non-standard
\tfrac	AMSmath	
\therefore	AMSsymbols	
\Theta		
\theta		
\thickapprox	AMSsymbols	
\thicksim	AMSsymbols	
\thinspace		
\tilde		
\times		
\tiny		
\Tiny		non-standard
\to		
\toggle	action	non-standard
\top		
\triangle		
\triangledown	AMSsymbols	
\triangleleft		
\trianglelefteq	AMSsymbols	
\triangleq	AMSsymbols	
\triangleright		
\trianglerighteq	AMSsymbols	
\tt		
\twoheadleftarrow	AMSsymbols	
\twoheadrightarrow	AMSsymbols	

U

\ulcorner	AMSsymbols	
\underbrace		
\underleftarrow		
\underleftrightarrow		
\underline		
\underrightarrow		
\underset		
\unicode	[unicode]	non-standard
\unlhd	AMSsymbols	
\unrhd	AMSsymbols	
\Uparrow		
\uparrow		
\Updownarrow		
\updownarrow		
\upharpoonleft	AMSsymbols	
\upharpoonright	AMSsymbols	
\uplus		
\uproot		
\Upsilon		
\upsilon		
\upuparrows	AMSsymbols	
\urcorner	AMSsymbols	

V

\varDelta	AMSsymbols
\varepsilon	
\varGamma	AMSsymbols
\varinjlim	AMSmath
\varkappa	AMSsymbols
\varLambda	AMSsymbols
\varliminf	AMSmath
\varlimsup	AMSmath
\varnothing	AMSsymbols
\varOmega	AMSsymbols
\varphi	
\varPhi	AMSsymbols
\varpi	
\varPi	AMSsymbols
\varprojlim	AMSmath
\varpropto	AMSsymbols
\varPsi	AMSsymbols
\varrho	
\varsigma	
\varSigma	AMSsymbols
\varsubsetneq	AMSsymbols
\varsubsetneqq	AMSsymbols
\varsupsetneq	AMSsymbols
\varsupsetneqq	AMSsymbols
\vartheta	
\varTheta	AMSsymbols
\vartriangle	AMSsymbols
\vartriangleleft	AMSsymbols
\vartriangleright	AMSsymbols
\varUpsilon	AMSsymbols
\varXi	AMSsymbols
\vcenter	
\vdash	
\Vdash	AMSsymbols
\vDash	AMSsymbols
\vdots	
\vec	
\vee	
\veebar	AMSsymbols
\verb	[verb]
\Vert	
\vert	
\vphantom	
\Vvdash	AMSsymbols

W

```
\wedge
\widehat
\widetilde
\wp
\wr
```

X

```
\Xi
\xi
\xcancel                cancel
\xleftarrow             AMSmath
\xlongequal             extpfeil
\xmapsto                extpfeil
\xrightarrow            AMSmath
\xtofrom                extpfeil
\xtwoheadleftarrow      extpfeil
\xtwoheadrightarrow     extpfeil
```

Y

```
\yen                    AMSsymbols
```

Z

```
\zeta
```

Environments

LaTeX environments of the form `\begin{XXX} ... \end{XXX}` are provided where `XXX` is one of the following:

```
align                   [AMSmath]
align*                  [AMSmath]
alignat                 [AMSmath]
alignat*                [AMSmath]
aligned                 [AMSmath]
alignedat               [AMSmath]
array

Bmatrix
bmatrix

cases
CD                       AMSmath

eqnarray
eqnarray*
equation
equation*

gather                  [AMSmath]
gather*                 [AMSmath]
gathered                [AMSmath]

matrix
multline                [AMSmath]
multline*               [AMSmath]

pmatrix

smallmatrix              AMSmath
split                   [AMSmath]
subarray                 AMSmath

Vmatrix
vmatrix
```