# `mathblog` User's Manual

For `mathblog` version 0.6

Jonathan Daugherty (cygnus@foobox.com)

October 3, 2015

# Contents


**1 Introduction** **2**

1.1 Project Vision . . . . . 2

**2 Using `mathblog`** **4**

2.1 Step 1: Create . . . . . 4

2.2 Step 2: Configure . . . . . 5

2.3 Step 3: Edit . . . . . 6

2.3.1 Blog Post Format . . . . . 7

2.4 When Should I Run `mb`? . . . . . 8

2.5 Customization . . . . . 8

2.6 Blog Assets . . . . . 10

2.7 TEX Macros . . . . . 10

2.8 TikZ Embedding . . . . . 10

2.8.1 Styling TikZ Graphics CSS . . . . . 11

2.9 Disqus Integration . . . . . 12

2.10 Controlling Post Order . . . . . 12

# Chapter 1

# Introduction

`mathblog` is a program targeted at people who want to write statically-generated, mathematically-themed weblogs. It supports:

- Extended Markdown input syntax as supported by the Pandoc library
- Inline and block-level TEX math rendered by MathJax
- Function graphing and drawing with the TikZ LATEX package
- Support for integration of Javascript-based web services such as Disqus
- Template-based document rendering with support for layout and style customization
- A built-in web server with automatic page reloading for easy blog post authoring

## 1.1 Project Vision

I wrote `mathblog` with a very specific set of requirements in mind, motivated by the following principles:

- A blog should be easy to create, host, and update.
- A blog should be easy to maintain.

- I should be able to edit posts in my editor of choice and write them in an intelligent textual markup language.
- It should be easy to embed high-quality mathematical symbols and equations in the blog posts.

As a result, `mathblog` has the following features:

- The software is composed of a single executable which will automatically take care of creating your blog and regenerating pages when your blog post markup changes.
- All content is stored in plain text files and is generated statically. No database or web framework is used and no resident processes are required to serve your content.
- `mathblog`'s generated output files can be hosted by any web server.
- `mathblog` provides a built-in web server for easy blog post authoring.
- Blog posts are written in the Markdown format with extensions, as supported by the Pandoc document converter.
- Math is embedded with `$...$` (or `\(...\)`) for inline math and `$$...$$` (or `\[...\]`) for block-level math.

These features have some nice advantages; your blog content is cacheable and can be subjected to revision control. Posts are easy to edit and editing doesn't require a web browser. The static file representation model means you can compose a blog post on your laptop and get it just right using a local installation of `mathblog`, then push it up to your server to post it to your public blog.

I personally use this software package but I'll be pleased if others find it useful. In addition, I'm open to accepting contributions on the project if they're consistent with the goals I've outlined above.

Happy blogging!

# Chapter 2

# Using `mathblog`

The main program provided by the `mathblog` package is the program `mb`. This program takes care of initializing new blog data directories, detecting changes in your files, regenerating the right HTML output files into your output directory, and even serving them via HTTP while you work.

`mb` needs to know about two directories: the *data directory* where your post input files and templates will be kept, and the *output directory* where the HTML version of your blog will be generated. `mb` has two ways of knowing about both of these directories: you can set environment variables or you can pass command-line flags to `mb`:

- Data directory: set `MB_DATA_DIR` in the environment or pass the `-d` option to `mb`.
- Output directory: set `MB_OUTPUT_DIR` in the environment or pass the `-o` option to `mb`.

Any command line parameters given to `mb` will always take precedence over any environment variables.

## 2.1 Step 1: Create

Once you've chosen your data and output directories, run `mb -i`. It will take care of setting up a new blog data directory for you, complete with some default pages and a default first blog post:

```
$ mb -d .../blog -o .../html -i
Initializing data directory .../blog
Rendering post: "first-post"
Done.
```

The directory structure of the new blog data directory is as follows:

- `blog.cfg` - The blog configuration file.
- `posts/` - Blog post files ("`*.txt`").
- `posts/posts-index` - A special text file called the *posts index*. This file contains a list of posts in the order in which they should be listed on the blog. It gets updated by `mathblog` when new posts are created, but you can edit the file to change the ordering if you need to. For more information, see Section 2.10.
- `templates/` - Templates used to generate the blog post pages, post listing page, and RSS feed. These templates are StringTemplate templates.
- `assets/` - The directory where you can put arbitrary files to be copied to the HTML output directory. `mathblog` will copy `assets/foo` to `$MB_OUTPUT_DIR/foo`.

## 2.2 Step 2: Configure

`mathblog` creates a default "INI-style" blog configuration file in your data directory called `blog.cfg`. It contains information which may be included in the generated pages. The configuration file must have the following fields set:

- `baseUrl` - The base URL of your blog; this URL will be used to generate some links in the blog's pages where absolute URLs matter. URL generation assumes that your blog is hosted at the root of a domain. The base URL corresponds to the output directory.
- `title` - The title of your blog, such as "My math blog".
- `authorName` - Your name (for the RSS feed metadata and the page footer).
- `authorEmail` - Your e-mail address (for the RSS feed metadata only).
- `mathBackend` - The backend used to render TEX math expressions. Right now the only possible value is `mathjax`. No special configuration is necessary for MathJax, as `mathblog`'s default templates use CDN resources for MathJax.

- `tikz` - Whether to permit the use of the Tikz LaTeX package to create diagrams and function plots. Set to `yes`, `on`, or `1` to enable. Disabled by default.

All of the above fields can be accessed in templates using the syntax described in Section 2.5 below.

## 2.3 Step 3: Edit

Now you might want to edit or create a new post or remove one. Start by editing the appropriate file in `posts/` and then run `mb`. To create a new blog post, just create a new file ending in "`.txt`" in the `posts/` subdirectory. Here's an example of running `mb` after modifying an existing post:

```
$ mb
Rendering post: "first-post"
Done.
```

Ordinarily, you might run `mb` once in a while to update your output directory. But if you're in the middle of authoring a new post and you want to see what it looks like as you edit it rather than run `mb` by hand repeatedly, you can run `mb` in "listen" mode; in listen mode, `mb` will start a lightweight web server and serve your blog locally while you edit it. If you open a web browser to one of your blog posts, it will automatically reload whenever you make a change to one of your blog files!

Here's a listen mode example:

```
$ mb -l
Starting up in listen mode...
Done.

Blog generation complete.
Web server listening on http://localhost:8000/
File created: .../blog/posts/posts-index
Rendering post (post-index): "first-post"
Done.

File modified: .../blog/blog.cfg
Rendering post (config): "first-post"
Done.

File modified: .../blog/posts/first-post.txt
Rendering post: "first-post"
Done.
```

Above I ran `mb` in listen mode and then modified the posts index, then the blog config file, then one of the posts. In each case `mb` detected the change and ran its usual regeneration routine.

You can set the hostname and port that `mathblog` uses to serve the blog; see the `mb -h` output.

### 2.3.1 Blog Post Format

Posts are formatted in Markdown and support the extended Markdown syntax as implemented by Pandoc. The only important convention to note is that the post title, author, and date values go on the first three lines of the file as follows:

```
% My First Post
% Author Name
% August 4, 1976

First paragraph starts here.
```

Dates in the blog post header formatted as "`<MONTH> <DAY>, <YEAR>`" will be parsed and used to generate the RSS feed publication date.

## 2.4 When Should I Run mb?

mb looks at the modification times of the post files in posts/, the config file, template files, asset files, and the posts index (see Section 2.10) when determining when to regenerate content. It also looks for new posts that haven't been rendered in the past. The rule of thumb is: re-run mb whenever you make any changes to anything in your blog data directory.

Modifications a post file will cause that post to be re-rendered, but modifications the configuration file or templates will cause ALL posts to be re-rendered since those changes impact how all pages are generated.

## 2.5 Customization

It's likely that you'll want to customize the look and feel of your blog. To this end, mathblog generates the pages of your blog by assembling various pieces of the page to create the final result. The biggest piece of a generated page is the blog post itself, but the surrounding elements are read from various files that are created by mathblog when it creates your blog data directory. These files are stored in the templates/ subdirectory of your blog data directory and are as follows:

- templates/rssTemplate.xml - This is the template used to generate your RSS feed.
- templates/pageTemplate.html - This file makes up the overall structure of every page on the blog.
- templates/postTemplate.html - This file makes up the structure of the post portion of the page, for pages which show posts (i.e., not the posts index).
- templates/listTemplate.html - This file is the template used to generate the "all posts" page. Once generated, the all-posts listing is then embedded in the page template.

The templates mentioned above are "StringTemplate" templates and are processed with the HStringTemplate package. The following template placeholders are supported in each template:

- `$title$`, `$baseUrl$`, `$authorName$`, `$authorEmail$` - These placeholders all correspond directly to fields on the 'blog.cfg' configuration file.
- `$extraPageHead$` - Content to be placed in the `<HEAD>` tag of the page, such as javascript tags, stylesheets, etc. You'll need to ensure that this is somewhere in your `<HEAD>` tag if you want to use `mathblog` features which may need to load extra resources.

These placeholders are provided in the list template:

- `$posts$` - The list of all posts in the blog, starting with the most recent. Each has the same structure as a post in the post template (i.e. post.title, post.date, post.url).

These placeholders are supported in the post template:

- `$next_post_url$`, `$prev_post_url$` - URLs to the next (newer) and previous (older) posts relative to the current post. One or both may be null.
- `$post_html$` - The rendered body of the post itself.
- `$post.basename$` - The base name of the post for use in Javascript (see Section 2.9 for an example).
- `$post.title$` - The title of the post as found in the Pandoc header.
- `$post.date$` - The publication date string of the post as found in the Pandoc header.
- `$post_authors$` - The list of post author names as found in the Pandoc header. This can be rendered as follows:

  ```
  Posted by $post_authors; separator=", "$
  ```

- `$post.tex_macros$` - The combined TEX macros string as found in all of the `#tex-macros` code blocks in the post. See the section on TEX macros for how to use this properly.

These placeholders are supported in the page template:

- `$content$` - The content of the page to be rendered.

## 2.6 Blog Assets

The default blog directory created by `mb` also includes an `assets/` subdirectory. Whenever `mb` detects changes in your blog post or asset files (or if you run "`mb -f`"), everything in `assets/` will be copied recursively into the output directory. This way, if you have custom stylesheets or other files which need to be available, they can be kept in your data directory and published with the rest of the generated content.

## 2.7 TEX Macros

You can define post-wide TEX macros and then reference them in both Mathjax expressions and TikZ pictures. To do this, define TEXmacros inside a "`tex-macros`" block as follows:

```
~~~ {#tex-macros}
\newcommand{\stuff}{...}
~~~
```

The block will be removed from the document during preprocessing, but the macros must be re-embedded in the final output HTML for Mathjax by updating your post template to include this *before* the post body:

```
<div style="display: none;">
\\(
$tex_macros$
\\)
</div>
```

The TEX macros will automatically be included in generated TEX source when TikZ processing is performed. This way, you can write macros and use them everywhere in the document without having to redefine them inside picture environments.

## 2.8 TikZ Embedding

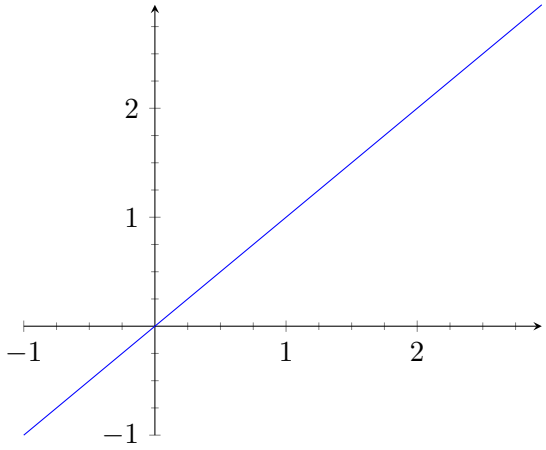
`mathblog` supports inline scripts for rendering function graphs and diagrams. Right now, only the TikZ LATEX package is supported. Set the appropriate configuration option (see Section 2.2) to enable it.

To specify a TikZ diagram in a blog post, we overload the Pandoc code block syntax. Here's an example of a TikZ figure:

```
~~~ {#tikz}
\begin{axis}[
    minor tick num=3,
    axis y line=center,
    axis x line=middle,
    ]

\addplot[smooth,mark=none,blue] plot coordinates {
    (-1,-1)
    (2.95,2.95)
};
\end{axis}
~~~
```

This is a LaTeX fragment which will automatically be embedded in a `tikzpicture` environment and rendered to an image within the blog post. The fragment above will produce this picture:




### 2.8.1 Styling TikZ Graphics CSS

Note that the Pandoc syntax also allows us to assign CSS class names to the code block, and `mathblog` passes these through to the generated image. So if you wanted to wrap

your text around the generated image, you could create a CSS class like this:

```
.eq-right {
  float: right;
}
```

and then assign it to your equation graph like this:

```
~~~ {#eq-basic .eq-right}
...
~~~
```

For more information on the code block syntax, please see:

http://johnmacfarlane.net/pandoc/README.html#fenced-code-blocks

## 2.9 Disqus Integration

Since `mathblog` doesn't provide many moving parts, it's up to you to outsource various web site features, such as comments. I've successfully integrated `mathblog` with the Disqus comments service. To do this, some javascript needs to be embedded in the blog pages. Disqus works best when you supply it with a page identifier so it can guarantee that comments are post-specific rather than URL-specific. The way `mathblog` makes this possible is by exposing a page basename string so you can configure Disqus properly:

```
var disqus_identifier = "$post.basename$";
```

## 2.10 Controlling Post Order

Whenever `mb` detects that you have added a new post, it updates the "posts index":

```
$MB_DATA_DIR/posts-index
```

This file lists the filenames of all posts from newest to oldest. By default, new posts get added to the beginning of the list, as you would expect. Any new posts added to the list

are sorted by modification time so that the newest post on disk appears earlier in the index. This feature exists to make it possible for older posts to be updated without changing their ordering in the overall sequence of posts.

You can edit the index at any time to reorder the posts as you see fit. `mb` will preserve ordering of posts already in the index when it runs. `mb` will also take care of removing posts from the index if they've been removed from the post source directory.