Web Publications — LaTeX Style

Give your Web Publications LaTeX Style

Stefan Gössner[1]

[1]Dortmund University of Applied Sciences. Department of Mechanical Engineering

May 2021

Keywords: mdmath, static page, publication, Journal, LaTeX, math

Abstract

Now there is an easy way to have both, a possibly interactive, static web page containing math and a scientific print document for documentation and publication, each looking like a professional LaTeX document. All you need is the popular free, open source *Visual Studio Code* text editor with its lightweight extension *Markdown+Math* for managing LaTeX style and math syntax as well as using meanwhile ubiquitous Markdown language. The resulting *HTML* document already contains prerendered math formulas, so browsers won't have the burden of math rendering via scripting.

Content

1. Introduction

There has been an upward trend in using Markdown language not only for web content, but also for student notes, handouts and scientific papers of small to medium size. Markdown source files are beneficial for viewing, editing and archiving content. Additionally we can generate visually pleasing LaTeX style documents from it

- as HTML pages possibly containing interactive graphics.
- as static PDF print documents.

Markdown parsers usually export semantic HTML, i.e. meaningful HTML elements only without using `class` attributes. Styling those HTML documents is accomplished then by using *classless* CSS stylesheets [4,5]. Styling semantic HTML with classless CSS ...

- makes HTML and CSS easy to read and understand.
- results in overall small code size.
- installs no rigid class name dependencies between HTML and CSS.

Edward Tufte's conventions for styling papers and handouts are followed here as a guidline [6].

This paper – styled itself as a LaTeX document – first lists the necessary installation aspects in short. After a tiny example of a LaTeX styled markdown document, it gives a quick overview of Markdown for newcomers. Finally some useful features proposed for scientific publication are explained in more detail.

Source code of the HTML template and its accompanying class-less CSS file can be found on GitHub [7].

2. Editor, Math Extension and Configuration

If you are a programmer, chances are high, that you already have installed and use *Visual Studio Code*. Otherwise you can install it from here and easily use it as a general non-wysiwyg text editor with excellent Markdown support.

Getting scientific now we install the VSCode extension markdown+math. You can do it directly from within VSCode via

```
View > Extensions
```

Now in VSCode select the menu entry

```
File > Preferences > Settings
```

and switch to the *mdmath* extension Theme `'publication'`.

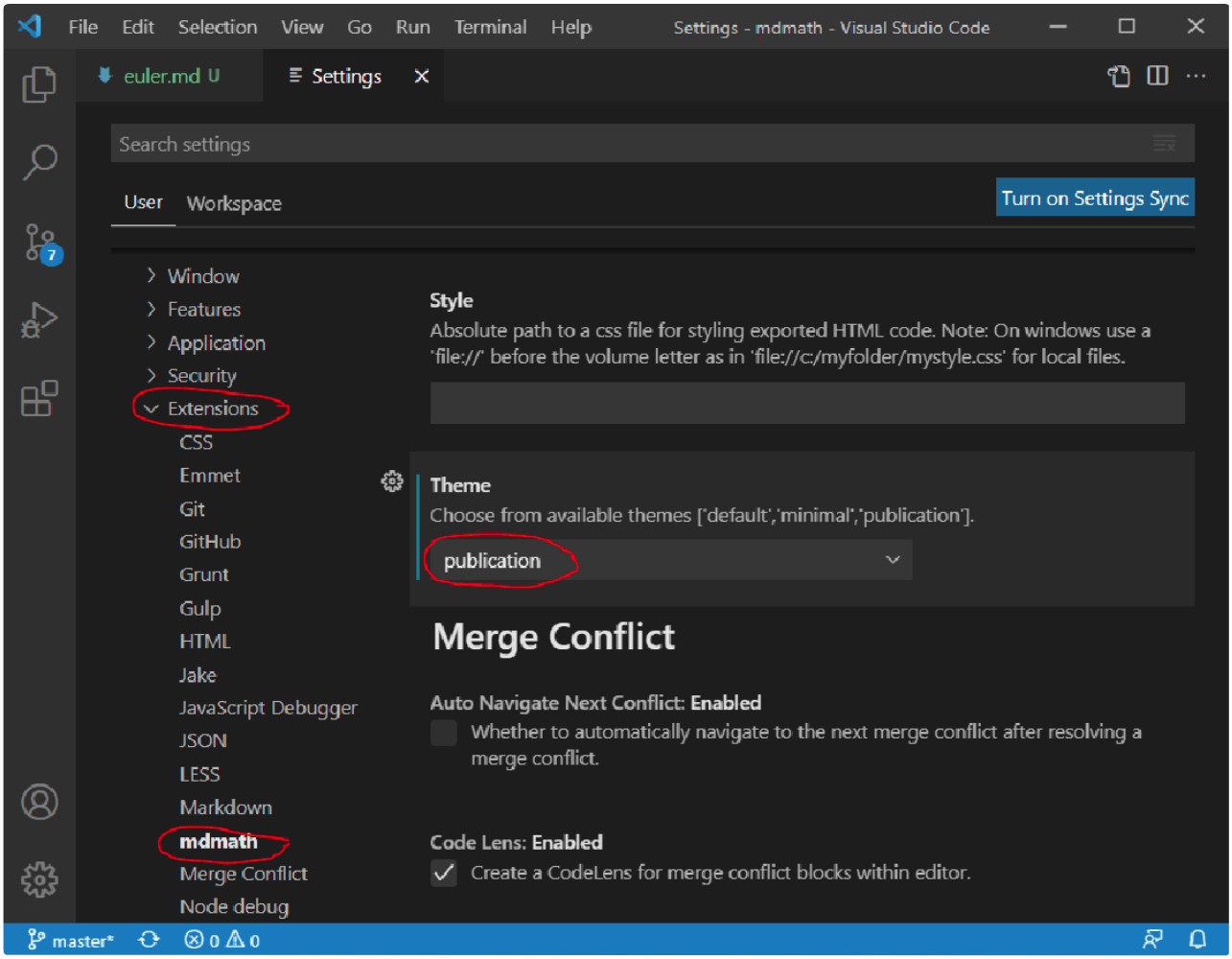

Fig 1: Setting theme for LaTeX output.

Now for verifying, that everything works as expected ...

- Open a new file in VSCode and save it as `euler.md`.
- Navigate your browser to the example file `https://raw.githubusercontent.com/goessner/mdmath/master/docs/euler.md` and copy / paste the markdown text into your open VSCode document.
- Open a preview window to the side (Ctrl + K V).

You should see something similar to the following yet:

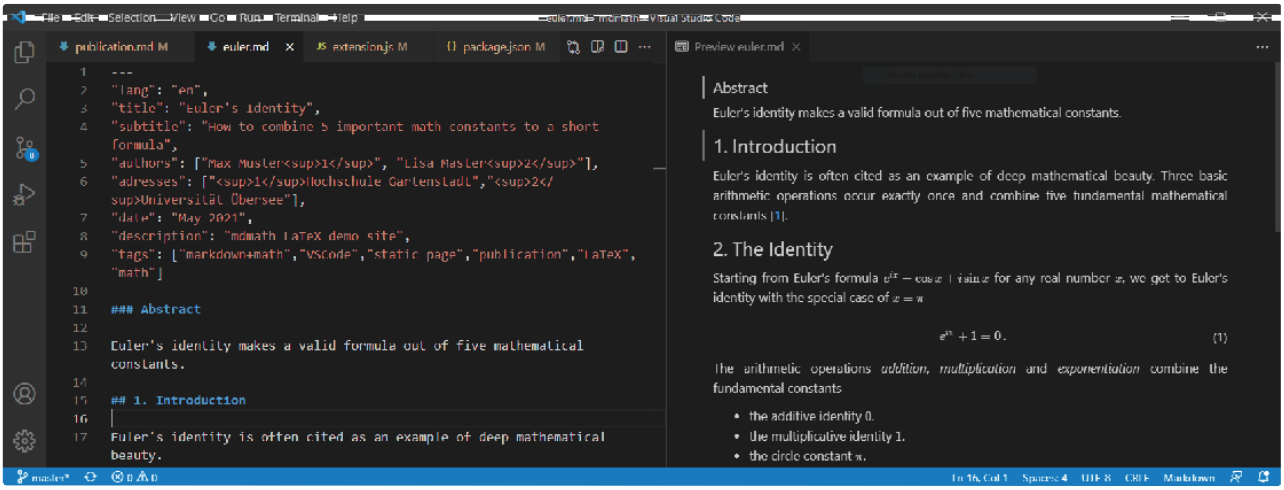

Fig 2: Markdown Example – Euler's Identity

Now from the VSCode Command Palette (Ctrl+Shift+P) run the command `Markdown: Save Markdown+Math to HTML` or simply press (Ctrl+K ,).

That way we have created an HTML version of our markdown file `euler.md` named `euler.html`, both located in the same folder. Viewing it in the browser of your choice gives:

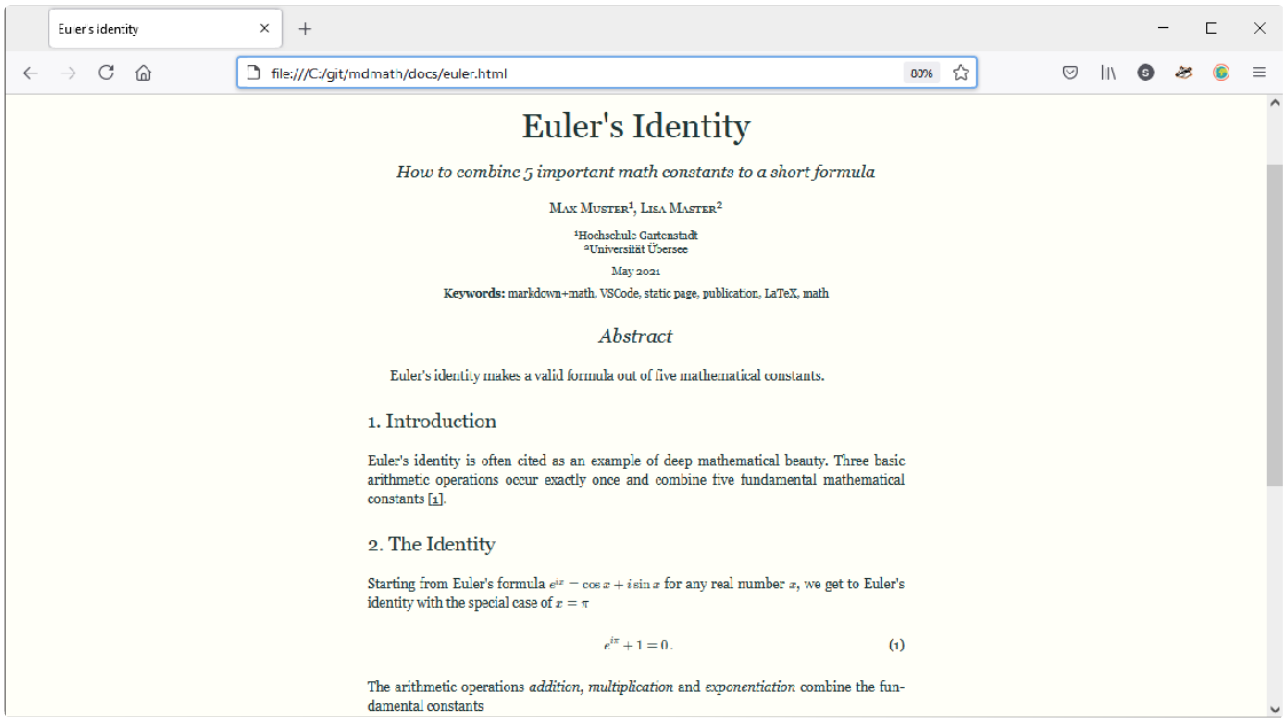

Fig 3: HTML Output – Euler's Identity

Having this we can generate a print document `euler.pdf` quite easily using our browser's *print to PDF* functionality. You can download a version from here for inspection.

3. Markdown Overview

According to the CommonMark specification [2], basic Markdown translates into a limited set of semantic HTML elements.

`<a>`, `<blockquote>`, `<code>`, `<em>`, `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`, `<h6>`, `<hr>`, `<img>`, `<li>`, `<ol>`, `<p>`, `<pre>`, `<strong>`, `<ul>`

It may be noticeable that not very semantic HTML elements `<div>` and `<span>` are missing here. For supporting these list of elements, a small CSS file suffices. Please also take note that Markdown allows to embed plain HTML code when needed.

3.1 Headings

Edward Tufte's conventions followed here recommend to use `<h1>` for the document title only, `<h2>` for section headings, and `<h3>` for lower level headings [6]. So for document structuring only two heading levels `'##'` (`<h2>`) and `'###'` (`<h3>`) are totally sufficient – with reference to famous *Feynman* lectures on physics [11].

```
# Reserved for Document Title `<h1>`
## Primary Section Headings `<h2>`
### Secondary Lower Level Headings `<h3>`
#### Not Recommended for Use `<h4> - <h6>`
```

3.2 Paragraph

Markdown interpretes a sequence of text lines, separated from others by a blank line, as a paragraph and translates it into an HTML `<p>` element. In the same way you can't control text wrap with HTML, you cannot do that with Markdown.

But you might – as an exception to that rule – force a *hard line break* by using HTML `<br>` or append at least two trailing spaces at the end of a line.

3.3 Images

Markdown images are inline elements with the syntax `![alt text](/path/to/img "title")`. These images are displayed side by side.

If we want to center-align an image and give it a caption, we use the semantic HTML `<figure>` and `<figcaption>` elements instead.

```
<figure>
  <img src="./img/triangle.png">
  <figcaption>Fig 1: The right triangle</figcaption>
</figure>
```

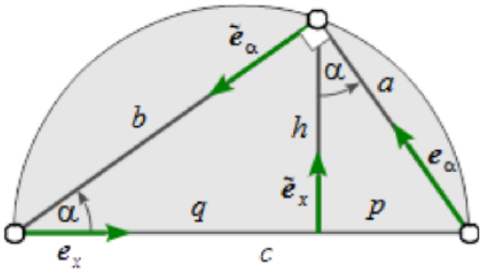

Fig 4: Right triangle

Images are responsive regarding their size, which we can verify by changing the size of the browser window.

3.4 Blockquote

Markdown uses email-style `>` characters for blockquoting.

```
> A human being is part of a whole called by us <q>Universe</q>.
> &mdash; Albert Einstein, *A Man of Many Parts*
```

> A human being is part of a whole called by us “Universe”.
> — Albert Einstein, *A Man of Many Parts*.

In scientific documents *Theorems*, *Lemmas* and *Proofs* are frequently written using blockquote syntax in Markdown.

3.5 Code Blocks

We can insert preformatted text in code blocks enclosed by three backticks ``` on its own single line each. Immediately following the begin-backticks we may specify the language for syntax highlighting.

````
```js
function print({x=0,y=0}) {
    return `(${x},${y}`;
};
```
<figcaption>Listing 1: JavaScript function</figcaption>
````

`<figcaption>` heading might be comfortably used as a code block caption.

```js
function print({x=0,y=0}) {
    return `(${x},${y}`;
};
console.log(print({x:3,y:4}));
```

Listing 1: JavaScript function

3.6 Inline Markdown

Besides regular Markdown inline rules there are some beneficial HTML inline elements.

Table 2: Some Inline Markdown / Markup

Markdown / Markup	**Result**
`*italic text*`	*italic text*
`**strong text**`	**strong text**
`***strong italic text***`	***strong italic text***
`~~strike through~~`	~~strike through~~
`<!--stripped comment-->`	
`<u>underlining</u>`	underlining
`Super<sup>script</sup>`	Super$^{\text{script}}$
`Sub<sub>script</sub>`	Sub$_{\text{script}}$
`<small>Small text for fine print</small>`	Small text for fine print
`<kbd>Alt</kbd>+<kbd>Ctrl</kbd> <kbd>Del</kbd>`	Alt + Ctrl Del
`<abbr title="Cascading Style Sheets">CSS</abbr>`	CSS
`Hilite text with <mark>mark</mark>`	Hilite text with mark
`Hard line break` *	Hard line break
`<q>... quotation ...</q>`	“... quotation ...”
`<cite>... citation ...</cite>`	*... citation ...*

3.7 Table

```
<figcaption> Table 3: Column alignment </figcaption>

| Left | Center | Right |
|:---|:---:|--:|
|L|C|R|
```

We can use a simple syntax for tables, which is a popular extension to the CommonMark standard [3]. Inline rules as shown in Table 2 apply to the table cells. Tables as a whole are always center-aligned.

Table 3: Column alignment

Left	Center	Right
L	C	R

3.8 Wrapping Text around figures, listings, etc.

This is not a Markdown feature. Also text wrapping is controversial discussed regarding its use in scientific papers. But if you want that, HTML `<aside>` comes for help. HTML specification recommends to "use `<aside>` only for content that is indirectly related to the main article content."

> Note the general fact, that we can use Markdown text inside of HTML block elements, as long as it starts and ends with a blank line.

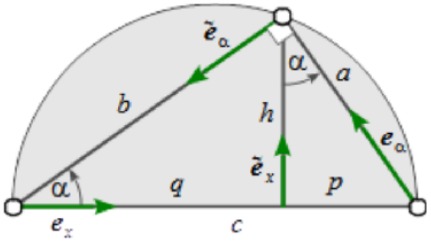

Fig. 6: Triangle aside

```
<aside>

   ![aside image](./img/triangle.png "triangle aside")
   <figcaption> Fig. 6: Triangle aside</figcaption>

</aside>
```

3.9 Math Support

Math support is the core functionality of *mdmath*. Inline math $r = \sqrt{x^2 + y^2}$ and display math expressions

$$e^{ix} = \cos x + i \sin x$$

are supported, due to `markdown-it` extension `markdown-it-texmath` [8] and the fast math renderer KaTeX [9]. Commutative diagrams can be used since *mdmath* version 2.6

$$\begin{CD} A @>a>> B \\ @VbVV @AAcA \\ C @= D \end{CD}$$

```
... Inline math $r = \sqrt{x^2 + y^2}$ and display math expressions $$e^{ix}=\cos x + i\sin
x$$ are supported ... Commutative diagrams can be used since *mdmath* version 2.6

$$\begin{CD} A @>a>> B \\ @VbVV @AAcA \\ C @= D \end{CD}$$
```

4. Document Structure

The visual part of exported HTML document is provided by *mdmath* template `themes/publication/theme.js`, which has a simple semantic structure:

```
<body>
  <header> ... </header>
  <main> ... </main>
</body>
```

Your Markdown content will be parsed by `markdown-it`, the Markdown parser used by VSCode [10]. Resulting HTML output goes then into the `<main>` section.

Markdown does not assist you to structure your textual content any further. Especially it does not use HTML `<article>` and `<section>` elements. But in fact it doesn't need to, as *headings* – the remaining structuring elements – are totally sufficient.

The `<header>` section contains the paper's title, authors, author addresses, date and keywords. These data are to be found in a *Frontmatter* metadata section at the beginning of the Markdown file and will also be processed via the template file. The *Frontmatter* section inside `euler.md` example has following structure

```
---
"lang": "en",
"title": "Euler's Identity",
"subtitle": "How to combine 5 important math constants to a short formula",
"authors": ["Max Muster<sup>1</sup>", "Lisa Master<sup>2</sup>"],
"adresses": ["<sup>1</sup>Hochschule Gartenstadt","<sup>2</sup>Universität Übersee"],
"date": "May 2021",
"description": "mdmath LaTeX demo site",
"tags": ["markdown+math","VSCode","static page","publication","LaTeX","math"]
---
```

> Note that Frontmatter syntax used here must strictly obey `JSON` syntax. So it is more restrictive than `YAML` based Frontmatter syntaxes.

mdmath offers a very basic but useful *Table Of Content* command. While editing Markdown ...

1. Invoke comand `Insert Table of Content` from Command Palette or press `Ctrl` + `K` `T`.
2. Manually remove unwanted ToC entries.
3. Done.

5. Styling

In order to approach LaTeX look and feel by Markdown authoring, a pure classless CSS stylesheet is needed. You might want to have a look into *mdmath*'s stylesheet.

There are some LaTeX stylesheets on the web.

- *Huy Nguyen*'s `latex-simple-css`, GitHub, stylesheet, Example
  - Classless CSS, HTML example file is heavily class-dependent.
  - No math.
  - No Latex fonts.
- *Andrew Belt*'s `WiTex`, GitHub, stylesheet, Example
  - LaTeX CSS file copied and modified from Wikipedia, not classless.
  - Math rendering via MathJax scripting.

    - `'Latin Modern Roman'` font files.
- *David Zollikofer*'s `latexcss`, GitHub, stylesheet, Example
    - Classless CSS + classes for theorem, lemma, proof, ...
    - Math rendering via MathJax scripting.
    - Data-url embedded `'Latin Modern Roman'` fonts.
- *Vincent Dörig*'s `latex-css`, GitHub, stylesheet, Example
    - Classless CSS + classes for theorem, lemma, proof, ...
    - Math rendering via MathJax scripting.
    - `'Latin Modern Roman'` font files.

All of these stylesheets seem to address handmade HTML exclusively. HTML as Markdown export is not mentioned.

Vincent Dörig's approach looks very promising. Acknowledgment goes to him for providing the *Latin Modern Roman* typeface font reused with *mdmath*.

6. Conclusion

HTML output provided by the theme "publication" of *mdmath* offers a styling with LaTeX look and feel. With it student notes, handouts and scientific papers can be comfortably authored using the Markdown language, while previewing the result in texteditor VSCode. During the Markdown → HTML conversion process math formulas are converted to browser understandable markup. So no script based math renderer is finally needed on the browser side. In fact no script at all is used within resulting HTML output.

mdmath addresses small to medium sized documents. However for large work as theses and books, available powerful LaTeX packages or – if you want to stick with Markdown – Pandoc is highly recommended.

References

[1] Daring Fireball (https://daringfireball.net/projects/markdown/)
[2] CommonMark (https://commonmark.org/)
[3] CommonMark - Deployed Extensions, (https://tinyurl.com/5bm568tn)
[4] No-Class CSS Frameworks, (https://css-tricks.com/no-class-css-frameworks/)
[5] The Next CSS Frontier — Classless, (https://tinyurl.com/vynsw3ew)
[6] Tufte-CSS, (https://edwardtufte.github.io/tufte-css/)
[7] Markdown+Math (https://github.com/goessner/mdmath)
[8] markdown-it-texmath, (https://github.com/goessner/markdown-it-texmath).
[9] KaTeX, (https://katex.org/)
[10] markdown-it, (https://github.com/markdown-it/markdown-it)
[11] The Feyman Lectures on Physics, (https://www.feynmanlectures.caltech.edu/)