Yamp

by @angrykoala



Yet Another Markdown Parser

The aim of this package is to provide an easy-to-use toolbox for markdown-related task including Html & Pdf conversion.

• GitHub: https://github.com/angrykoala/yamp

Npm: https://www.npmjs.com/package/yamp

Features

- HTML conversion
- PDF conversion
- Code highlight support
- Github-style output
- API to use *yamp* programmatically
- Custom styles
- CSS-embedded HTML (just open it offline in any browser)
- HTML tags support (for PDF output too)
- Include other files in your markdown
- HTML presentations
- Koalafied

Upcoming features

- Custom templates
- Client-side web support (browserify)

- Metadata on markdown file
- Check the project roadmap and our cute kanban board
- Live example Here

Installation

To use yamp cli, install it globally using npm:

```
npm install -g yamp
```

If you want to use the API instead, install it locally:

```
npm install --save yamp
```

then, include yamp in your javascript:

```
var yamp = require('yamp');
```

Usage

To create a .pdf file from your *markdown* file, simply type:

```
yamp <file.md>
```

For example:

```
yamp README.md
```

Will generate readme.pdf.

Options

- -h, --help to display a basic usage information
- -v, --version to display yamp version installed
- -o, --output <file> output filename (without extension)
- --pdf to generate a pdf (default)
- --html to generate html
- --remark to generate a html presentation using remark
- -t, --title [value] to add a custom title
- --style <file> to set a custom CSS stylesheet
 - Option not suported along with --no-style
- --no-style to disable CSS styling
 - Options not supported along with --style <file>
- --minify to minify Html output
- --no-tags to disable makdown tags
- --no-highlight to disable code highlight
- -k, --koala to koalify your outputs

To generate pdf and html with default styling and options:

```
yamp myFile.md --pdf --html
```

The --no-highlight and --no-style options will greatly reduce your Html and Pdf outputs

Yamp tags

Yamp supports extra tags in your markdown files. Currently using xejs templates. All tags are written between tags $\{\{\ldots\}\}$

- include [file.md]: Includes the given text file (markdown or not), the tags on the included file will also be parsed, allowing nested file structure
- date: Will write the current date (at the moment of rendering)
- page break: Will force a page break in pdf output

API

Include yamp in your javascript with:

```
var yamp = require('yamp');
```

You'll have access to different renderers to process your files:

- yamp.renderers.html to process a markdown file into an full Html page
- yamp.renderers.pdf to process a markdown into a pdf

To use a renderer:

```
var myRenderer = new renderers.pdf(options);
renderer.renderFile(myFile, function(err){
    if (err) return console.log("Error while rendering: "+err);
    else console.log("Rendering was successful");
});
```

Options

The options accepted by the default renderers are:

- **outputFilename**: name of the output filename (without extension), will default to the input filename
- highlight: (true) indicates if code blocks should be highlighted
- **style**: (*true*) indicates if default style should be used or no style at all. If a filename is passed, it will use it as custom css style

- minify: (false) whether the Html output should be minified or not
- title: Custom title for the Html page
- tags: (true) whether to parse yamp tags or not ({{ . . . }})
- koala: (false) true to koalify your outputs

Creating new renderers

If you need a custom renderer, instead of using one of the defaults you can extend directly from **Renderer** class or any of the default renderers:

```
class MyCustomRenderer extends yamp.Renderer {
    constructor(options) {
        super(options, "default.ejs", yamp.parsers.md2Html);
        this.output="html"; //desired output extension
    }
    beforeLoad(filename){
        //Modify filename or this.fileLoader before loading it
    }
    beforeRender(templateOptions) {
        // Modify the data passed to the template before rendering, inc
luding title, content and options
    afterRender(content) {
        // Modify template result (Html)
    }
   fileOutput(content, done) {
        // Write file (preferably to this.options.outputFilename) in th
e desired format using a parser
}
```

Custom parser: It is possible to use a custom parser from markdown to Html instead of the built-in *yamp.parsers.md2html*, the parser must be a function of the type function(originalString, options, callback) that will translate from

originalString (markdown) to html, calling the callback(err, res) afterwards.

Currently only **default.ejs** is supported as template.

If, instead of extending from <code>yamp.Renderer</code> you are extending from one of the default renderers, you should only re-implement the methods you need, and usually you should call <code>super().methodName</code> to maintain its basic functionality.

Development Instructions

To contribute to **yamp** you should clone the official repository https://github.com/angrykoala/yamp or your own *fork* with git .

You can also download it from GitHub clicking here

- To install execute npm install in the downloaded/cloned folder
- To test, execute npm test
- To execute the CLI, execute npm start -- <file> [options]
- To install your local version globally, execute npm install -g . on the project folder

It is strongly recommended to install the npm repository version instead of your local copy

Acknowledgments

- Markdown-it as markdown parser
- Github-markdown.css as default parse style
- Highlight.js for code highlighting
- html-pdf for pdf generation
- remark for html slides output

YAMP is developed under GNU GPL-3 license by @angrykoala