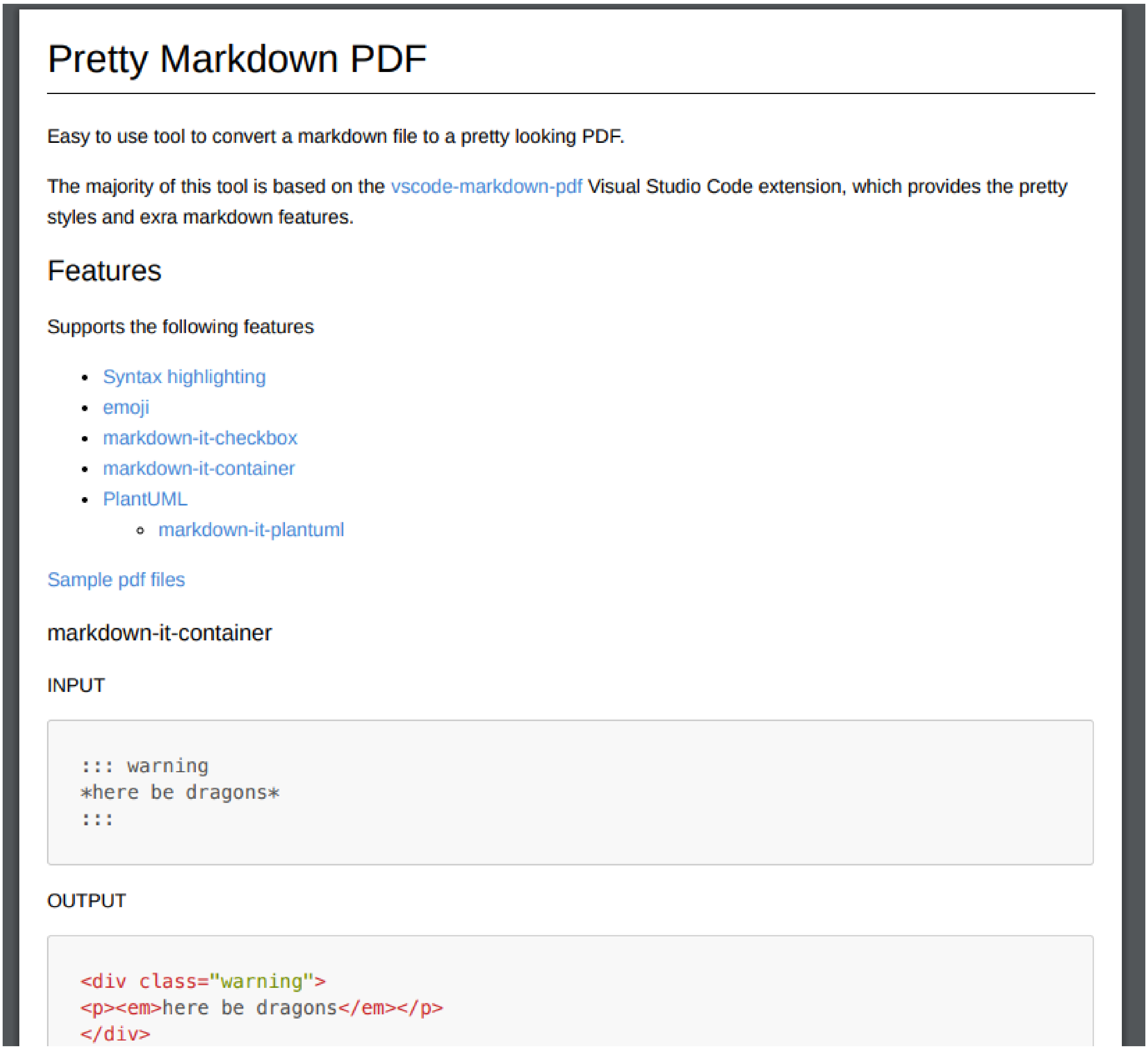
Pretty Markdown PDF

Easy to use tool to convert a markdown file to a pretty looking PDF.

The majority of the code is based on the vscode-markdown-pdf Visual Studio Code extension, which provides the pretty styles and extra markdown features.



Pretty Markdown PDF

Easy to use tool to convert a markdown file to a pretty looking PDF.

The majority of this tool is based on the vscode-markdown-pdf Visual Studio Code extension, which provides the pretty styles and exra markdown features.

Features

Supports the following features

- Syntax highlighting
- emoji
- markdown-it-checkbox
- markdown-it-container
- PlantUML
  - markdown-it-plantuml

Sample pdf files

markdown-it-container

INPUT

```
::: warning
*here be dragons*
:::
```

OUTPUT

```
<div class="warning">
<p><em>here be dragons</em></p>
</div>
```

Features

Supports the following features

- Supports export from MD to PDF, HTML, PNG and JPEG

- Syntax highlighting

- emoji

- markdown-it-checkbox

- markdown-it-container
- PlantUML
    - markdown-it-plantuml
- Sample pdf file
- Sample html file
- Sample png file
- Sample jpeg file

markdown-it-container

INPUT

```
::: warning
*here be dragons*
:::
```

OUTPUT

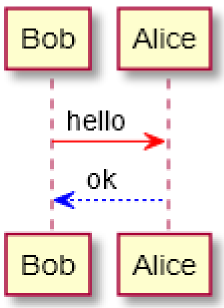
```
<div class="warning">
<p><em>here be dragons</em></p>
</div>
```

markdown-it-plantuml

INPUT

```
@startuml
Bob -[#red]> Alice : hello
Alice -[#0000FF]->Bob : ok
@enduml
```

OUTPUT



Usage

Install this project from the NPM package repository:

```
npm install -g pretty-markdown-pdf
```

Command Line

To convert a markdown file to PDF, simply run:

```
pretty-md-pdf -i my-doc.md
```

Run with `--help` to see all the options available.

This will output a file `my-doc.pdf` in the directory where `my-doc.md` resides.

To specify an output path as `my-exported-doc.pdf`, run:

```
pretty-md-pdf -i my-doc.md -o my-exported-doc.pdf
```

Other Export Types

To specify an output type other than PDF, run:

```
pretty-md-pdf -i my-doc.md -t png
```

Javascript

You can programmatically call this package, example:

```
const prettyMdPdf = require("pretty-markdown-pdf")

// output to `my-doc.pdf`
prettyMdPdf.convertMdToPdf({ markdownFilePath: "my-doc.md" })

// specify an output file path
prettyMdPdf.convertMdToPdf({ markdownFilePath: "my-doc.md", outputFilePath:
"output.pdf" })

// specify an output file type, outputs to `my-doc.html`
prettyMdPdf.convertMdToPdf({ markdownFilePath: "my-doc.md", outputFileType: "html"
})
```

Configuration

Advanced configuration is done using a JSON file. By default the tools uses the one shipped with this package, which has defaults set.

To specify a config file when running the tool:

```
pretty-md-pdf -i my-doc.md -c /tmp/config.json
```

From JavaScript you can do:

```
prettyMdPdf.convertMdToPdf({ markdownFilePath: "my-doc.md", configFilePath:
"/tmp/config.json")
```

Example

Below is the default config JSON file as an example:

```
{
    "type": [
        "pdf"
    ],
    "outputDirectory": "",
    "outputDirectoryRelativePathFile": false,
    "styles": [],
    "stylesRelativePathFile": false,
    "includeDefaultStyles": true,
    "highlight": true,
    "highlightStyle": "",
    "breaks": false,
    "emoji": true,
    "markdown-it-include": true,
    "executablePath": "",
    "scale": 1,
    "displayHeaderFooter": true,
    "headerTemplate": "<div style=\"font-size: 9px; margin-left: 1cm;\"> <span
class='title'></span></div> <div style=\"font-size: 9px; margin-left: auto; margin-
right: 1cm; \"> <span class='date'></span></div>",
    "footerTemplate": "<div style=\"font-size: 9px; margin: 0 auto;\"> <span
class='pageNumber'></span> / <span class='totalPages'></span></div>",
    "printBackground": true,
    "orientation": "portrait",
    "pageRanges": "",
    "format": "A4",
    "width": "",
    "height": "",
    "margin": "1cm",
    "quality": 100,
    "clip": {
        "height": null
    },
    "omitBackground": false
}
```

Options

Category	Option name
Save options	type

Category	Option name
	outputDirectory
	outputDirectoryRelativePathFile
Styles options	styles
	stylesRelativePathFile
	includeDefaultStyles
Syntax highlight options	highlight
	highlightStyle
Markdown options	breaks
Emoji options	emoji
Configuration options	executablePath
Common Options	scale
PDF options	displayHeaderFooter
	headerTemplate
	footerTemplate
	printBackground
	orientation
	pageRanges
	format
	width
	height
	margin.top
	margin.bottom
	margin.right
	margin.left
PNG JPEG options	quality
	clip.x
	clip.y
	clip.width
	clip.height
	omitBackground

Save options

`type`

- Output format: pdf, html, png, jpeg
- Multiple output formats support
- Default: pdf

```
"type": [
  "pdf",
  "html",
  "png",
  "jpeg"
],
```

`outputDirectory`

- Output Directory
- All `\` need to be written as `\\` (Windows)

```
"outputDirectory": "C:\\work\\output",
```

Relative Paths

```
"outputDirectory": "~/output",
```

- If you set a directory with a `relative path`, it will be created if the directory does not exist
- If you set a directory with an `absolute path`, an error occurs if the directory does not exist

`outputDirectoryRelativePathFile`

- If `outputDirectoryRelativePathFile` option is set to `true`, the relative path set with outputDirectory is interpreted as relative from the file
- boolean. Default: false

Styles options

`styles`

- A list of local paths to the stylesheets to use from pretty-markdown-pdf
- If the file does not exist, it will be skipped
- All `\` need to be written as `\\` (Windows)

```
"styles": [
  "C:\\Users\\<USERNAME>\\Documents\\css",
  "/home/<USERNAME>/settings/css",
],
```

```
"styles": [
  "css",
],
```

```
"styles": [
  "~/.config/Code/User/css"
],
```

- Remote CSS (https://xxx/xxx.css) is applied correctly for JPG and PNG, but problems occur with PDF

```
"styles": [
  "https://xxx/css"
],
```

`stylesRelativePathFile`

- If `stylesRelativePathFile` option is set to `true`, the relative path set with styles is interpreted as relative from the file
- boolean. Default: false

`includeDefaultStyles`

- Enable the inclusion of default Markdown styles
- boolean. Default: true

Syntax highlight options

`highlight`

- Enable Syntax highlighting
- boolean. Default: true

`highlightStyle`

- Set the style file name. for example: github.css, monokai.css ...
- file name list
- demo site : https://highlightjs.org/static/demo/

```
"highlightStyle": "github.css",
```

Markdown options

`breaks`

- Enable line breaks
- boolean. Default: false

Emoji options

`emoji`

- Enable emoji. EMOJI CHEAT SHEET
- boolean. Default: true

Configuration options

`executablePath`

- Path to a Chromium or Chrome executable to run instead of the bundled Chromium
- All `\` need to be written as `\\` (Windows)

```
"executablePath": "C:\\Program Files
(x86)\\Google\\Chrome\\Application\\chrome.exe"
```

Common Options

`scale`

- Scale of the page rendering
- number. default: 1

```
"scale": 1
```

PDF options

- pdf only. puppeteer page.pdf options

`displayHeaderFooter`

- Enable display header and footer
- boolean. Default: true

`headerTemplate`

`footerTemplate`

- HTML template for the print header and footer
- `<span class='date'></span>` : formatted print date
- `<span class='title'></span>` : markdown file name
- `<span class='url'></span>` : markdown full path name
- `<span class='pageNumber'></span>` : current page number
- `<span class='totalPages'></span>` : total pages in the document

```
"headerTemplate": "<div style=\"font-size: 9px; margin-left: 1cm;\"> <span
class='title'></span></div> <div style=\"font-size: 9px; margin-left: auto; margin-
right: 1cm; \"> <span class='date'></span></div>",
```

```
"footerTemplate": "<div style=\"font-size: 9px; margin: 0 auto;\"> <span
class='pageNumber'></span> / <span class='totalPages'></span></div>",
```

`printBackground`

- Print background graphics
- boolean. Default: true

`orientation`

- Paper orientation
- portrait or landscape
- Default: portrait

`pageRanges`

- Paper ranges to print, e.g., '1-5, 8, 11-13'
- Default: all pages

```
"pageRanges": "1,4-",
```

`format`

- Paper format
- Letter, Legal, Tabloid, Ledger, A0, A1, A2, A3, A4, A5, A6
- Default: A4

```
"format": "A4",
```

`width`

`height`

- Paper width / height, accepts values labeled with units(mm, cm, in, px)
- If it is set, it overrides the format option

```
"width": "10cm",
"height": "20cm",
```

`margin.top`

`margin.bottom`

`margin.right`

`margin.left`

- Paper margins.units(mm, cm, in, px)

```
"margin.top": "1.5cm",
"margin.bottom": "1cm",
"margin.right": "1cm",
"margin.left": "1cm",
```

PNG JPEG options

- png and jpeg only. puppeteer page.screenshot options

`quality`

- jpeg only. The quality of the image, between 0-100. Not applicable to png images

```
"quality": 100,
```

`clip.x`

`clip.y`

`clip.width`

`clip.height`

- An object which specifies clipping region of the page
- number

```
//  x-coordinate of top-left corner of clip area
"clip.x": 0,

// y-coordinate of top-left corner of clip area
"clip.y": 0,

// width of clipping area
"clip.width": 1000,

// height of clipping area
"clip.height": 1000,
```

`omitBackground`

- Hides default white background and allows capturing screenshots with transparency
- boolean. Default: false

FAQ

How can I change emoji size ?

1. Add the following to your stylesheet, which you can specified in the styles field

```
.emoji {
  height: 2em;
}
```

Output directory

If you want to always output to a directory path relative from the Markdown file.

For example, to output to the "output" directory in the same directory as the Markdown file, set it as follows.

```
"outputDirectory" : "output",
"outputDirectoryRelativePathFile": true,
```

Page Break

Please use the following to insert a page break.

```
<div class="page"/>
```

Note on Chromium

Chromium download starts automatically before the first conversion; this is a one time operation, only if your reinstall this package will it be downloaded again.

This isa time-consuming task depending on the environment because of its large size (~ 170Mb Mac, ~ 282Mb Linux, ~ 280Mb Win).

During the Chromuim download, the message `Installing Chromium` will be displayed in the console.