Options for MkDocs PDF Generate Plugin

Options

Duodu Randy

Table of Contents

1. Global Options 4
    1.1. for Cover 5
        1.1.1. `cover` 5
        1.1.2. `cover_title` 5
        1.1.3. `cover_subtitle` 5
        1.1.4. `author` 5
        1.1.5. `author_logo` 6
        1.1.6. `copyright` 6
    1.2. `disclaimer` 6
        1.2.1. `include_legal_terms` 6
        1.2.2. `cover_images` 6
    1.3. for Heading and TOC 7
        1.3.1. `toc` 7
        1.3.2. `toc_title` 7
        1.3.3. `toc_level` 7
        1.3.4. `toc_numbering` 7
    1.4. ... and more 7
        1.4.1. `enable_csv` 7
        1.4.2. `custom_template_path` 8
        1.4.3. `enabled_if_env` 9
        1.4.4. `verbose` 9
        1.4.5. `debug` (for development purposes only) 9
        1.4.6. `debug_target` (for development purposes only) 10
        1.4.7. `media_type` 10
        1.4.8. `theme_handler_path` 10
2. Local Options 11
    2.1. `build` 11
    2.2. `title` 12
    2.3. `subtitle` 12

2.4. `type` 12

2.5. `filename` 12

2.6. `revision` 12

2.7. `csv_name` 12

2.8. `toc_txt` (experimental) 13

2.9. `legal_terms` 13

2.10. `cover_image` 13

1. Global Options

The plugin allows users to pass in both global and local options.

> **Note**
>
> Local options have higher precedence than global options.

Some of these global options are used as default options when local options are not set. The global options are passed to the plugin through `mkdocs.yml`:

```
plugins:
    - pdf-generate:
        author: "Randy Duodu"
        author_logo: img/logo.svg
        copyright: "Copyright © 2022 - MkDocs PDF Generate"
        disclaimer: "Disclaimer: Content can change at anytime and best to refer to website
for latest information."
        include_legal_terms: true
        cover: true
        cover_title: TITLE TEXT
        cover_subtitle: SUBTITLE TEXT
        custom_template_path: TEMPLATES PATH
        toc: false
        toc_level: 3
        toc_title: TOC TITLE TEXT
        toc_numbering: true
        cover_images:
          default: img/default.svg
          type1: img/type1.png
          type2: https://example.com/cover.svg
        enabled_if_env: ENABLE_PDF_EXPORT
```

1.1. for Cover

1.1.1. `cover`

Set the value to `false` if you don't need a cover page.

default: `true`

1.1.2. `cover_title`

Set the title text in cover page.

> **Note**
>
> The following rule is applied when setting the value for cover title text in cover page.
>
> Cover Title precedence:
>
> 1. title (local pdf metadata option)
> 2. H1 heading for page
> 3. cover title (global)
> 4. `site_name` variable in your project's `mkdocs.yml`

default: use `site_name` in your project's `mkdocs.yml`

1.1.3. `cover_subtitle`

Set the subtitle text in cover page.

> **Note**
>
> The following rule is applied when setting the value for cover subtitle text in cover page.
>
> Cover Subtitle precedence:
>
> 1. subtitle (local pdf metadata option)
> 2. type (local pdf metadata option)
> 3. cover subtitle (global)

default: `None`

1.1.4. `author`

Set the author text.

default: use `site_author` in your project's `mkdocs.yml`

1.1.5. `author_logo`

Provide a logo image which you can use in the cover page.

> **Tip**
>
> Using an SVG image as the value for author logo is recommended.

default: use `theme.logo` in your project's `mkdocs.yml`

1.1.6. `copyright`

Set the copyright text.

default: use `copyright` in your project's `mkdocs.yml`

1.2. `disclaimer`

Set the disclaimer text.

1.2.1. `include_legal_terms`

Set the value to `true` if you want to include legal information sections at the end of your PDF document.

If you specify this option, you need to have an HTML file or Jinja2 template file named `legal_terms` with any of these file extensions (`.html.j2` , `.html.jinja2` , `.html` , `.htm`). Also, you can create a customised legal_terms template and assign its name to the `legal_terms` local option. Check the legal_terms local option for more information.

The disclaimer templates must be saved under the folder you specified as the custom_template_path.

default: `false`

1.2.2. `cover_images`

Set the cover image for specific document types.

The option takes a `key-value` pair where the `key` must be the same value you specified for the type local pdf metadata option.

The `value` for a `key` must be the path to the image.

Example:

```
- pdf-generate:
    cover_images:
      default: img/home-banner.svg
      home: img/manual-banner.svg
      options: img/project-banner.svg
      customize: img/tutorial-banner.svg
```

> **Note**
>
> Apart from the `default` key-value pair under cover_images, the other key-value pairs can have user-defined values.
>
> Recommended: You must specify an image path for the `default` key-value pair. We will use the image as the cover image for any document that does not specify the type local pdf metadata option.

default: `None`

1.3. for Heading and TOC

1.3.1. `toc`

Set the value to `false` if you don't need a table of content section in the PDF document.

default: `true`

1.3.2. `toc_title`

Set the title text for *Table of Contents*.

default: `Table of Contents`

1.3.3. `toc_level`

Set the level of *Table of Contents*. This value is enabled in the range of from `1` to `6`.

default: `4`

1.3.4. `toc_numbering`

Set the value to `false` if you don't want your table of contents to be numbered in the PDF document.

default: `true`

1.4. ... and more

1.4.1. `enable_csv`

Set the value to `true` if you want to create a CSV file containing data about each valid document based on the format below:

```
title, type, revision, , , pdf_url, pdf_checksum, txt_checksum, txt_url
```

default: `false`

> **Note**
>
> The CSV file will contain data about documents with the toc_txt local option set to `true`.

1.4.2. `custom_template_path`

A relative path inside your projects' directory. This folder is where you put both the custom cover template and custom plugin CSS file (`cover.html and custom.css`).

> **Info**
>
> The custom template's filename can either be `cover` or the document type with any of these file extensions `.html.j2` , `.html.jinja2` , `.html` , or `.htm` . Example: `cover.html.j2` , `cover.html.jinja2` , `cover.html` , `cover.htm` OR Example: if document type is `manual` then you can create a template file called `manual.html` or `manual.html.j2` or `manual.html.jinja2` or `manual.htm` .
>
> You can refer to this example about how to use a custom cover template and custom CSS.

default: `use default plugin template`

1.4.3. `enabled_if_env`

Setting this option will generate PDF files only if there is an environment variable set to 1. The environment variable must match the value of `enabled_if_env`.

This is useful to disable building the PDF files during development, since it can take a long time to export all files.

default: `None`

PDF generation can take significantly longer than HTML generation which can slow down MkDocs built-in dev-server.

Adding `enabled_if_env: ENABLE_PDF_EXPORT` disables PDF generation during development and runs the dev-server normally:

```
$ mkdocs serve
INFO     -  Building documentation...
INFO     -  PDF export is disabled (set environment variable ENABLE_PDF_EXPORT to 1 to enable)
INFO     -  Cleaning site directory
INFO     -  Documentation built in 0.54 seconds
INFO     -  [08:51:24] Watching paths for changes: 'docs', 'mkdocs.yml'
INFO     -  [08:51:24] Serving on http://127.0.0.1:8000/
```

and to build PDF files, set the `ENABLE_PDF_EXPORT=1` environment variable:

```
$ ENABLE_PDF_EXPORT=1 mkdocs build
...
INFO    -  Converting 2 files to PDF took 1.82s
INFO    -  Documentation built in 2.29 seconds
```

1.4.4. `verbose`

Setting this to `true` will show all WeasyPrint debug messages during the build.
default: `false`

1.4.5. `debug` (for development purposes only)

Setting this to `true` enables the debug mode which saves the HTML content used in generating the PDF files into a folder called **`pdf_html_debug`**. The **`pdf_html_debug`** folder is relative to the documentation source directory.

This option is intended to help users in writing appropriate CSS styles for the HTML content used to generate the PDF documents.

default: `false`

> **Note**
>
> - The `debug` option only works with the `mkdocs build` command.
> - It is recommended to add the **`pdf_html_debug`** folder to your ignored files when using a version control system.

1.4.6. `debug_target` (for development purposes only)

This option helps you to generate a PDF file for a single target document. The value for `debug_target` should be the relative path to the target document. Example: `debug_target: customise/customisation.md`.

This option is intended to help reduce the time spent by users in debugging a single document used to generate a PDF file.

default: `null`

> **Note**
>
> - The `debug_target` option only works with the `mkdocs build` command.
> - You must set the debug option to `true`, if you want to use the `debug_target` option.

1.4.7. `media_type`

Allows you to use a different CSS media type (or a custom one like `pdf-generate`) for the PDF export.
default: `print`

1.4.8. `theme_handler_path`

Allows you to specify a custom theme handler module. This path must be ***relative to your project root*** (See example below).
default: `None`

`mkdocs.yml`:

```
plugins:
  - pdf-generate:
      theme_handler_path: theme-handler
```

```
project-root
├── theme-handler.py
├── docs
├── mkdocs.yml
├── site
.
.
```

2. Local Options

The plugin allows you to set document specific options using the Markdown page metadata. If a page metadata is specified, it has higher precedence than the global options.

The local options are specified in the specific Markdown document you want to use:

```
---
pdf:
  build: false
  filename: Plugin Options
  title: Options for MkDocs PDF Generate Plugin
  type: Manual
  revision: 0.2
  toc_txt: true
  cover_image:
    id: cover_product_image
    source: "img/product_img.png" # relative to the MD file itself
    inline_css: "position: absolute; opacity: 0.8; height: 10mm; width: 15mm; left: 50mm; top:
50mm;"
---
```

The following options are available:

- build
- title
- subtitle
- type
- filename
- revision
- csv_name
- toc_txt
- legal_terms
- cover_image

2.1. `build`

Allows you to specify whether to generate a PDF file for a Markdown file. Value is `true` or `false`.

> **Note**
>
> The function of the build option is different from that of the enabled_if_env. The `build` option disables PDF generation for a single Markdown file while `enabled_if_env` disables PDF generation for the entire project.

default: `true`

2.2. `title`

Set the title text in cover page.

2.3. `subtitle`

Set the subtitle text in cover page.

2.4. `type`

Set the document type.

> **Note**
>
> - We use the value, of this option, in selecting the document's cover image from the cover_images option.
> - Also, we use the value, of this option, in selecting the custom cover template.

2.5. `filename`

Set the filename to use for a specific page when downloading the PDF document.

> **Note**
>
> Filename precedence:
>
> 1. filename (local pdf metadata) - formatted such that all are valid characters
> 2. title (local pdf metadata) - formatted such that all are valid characters
> 3. title (local metadata) - formatted such that all are valid characters
> 4. H1 - formatted such that all are valid characters

2.6. `revision`

Set the revision text in cover page.

2.7. `csv_name`

Set the product name for a row in the CSV file. The value for this option is used as the title for a particular row in the CSV file.

2.8. `toc_txt` (experimental)

Set to `true` if you want to build a TXT file that contains the Table of Contents of the Markdown file. Value is `true` or `false`.

default: `false`

> **Note**
>
> The TXT file acts as the Table of Contents lookup table for a PDF document.

> **Warning**
>
> You must set both the toc and toc_numbering global options to `true` before using this option.

2.9. `legal_terms`

Set the name of the custom template file, without the file extensions (e.g. *legal_terms* and not *legal_terms.html.j2* or *legal_terms.html*), you want to use that contains the legal_terms information. If the custom template file is not found, we use the global `legal_terms.html.j2` template, if it exists or the legal_terms information is not added.

The custom template must be an HTML file or Jinja2 template file.

> **Note**
>
> - You must set the include_legal_terms global option to `true` before using this option.
> - The legal_terms template's filename can either be `legal_terms` or any accepted filename with one of these file extensions `.html.j2`, `.html.jinja2`, `.html`, or `.htm`.
>
> **Example**: if `legal_terms` option is set to `privacy` then you can create a template file called `privacy.html` or `privacy.html.j2` or `privacy.html.jinja2` or `privacy.htm`.

2.10. `cover_image`

The `cover_image` option allows users to add an image to the cover page.
default: `None`

The option takes 3 required `key-value` pairs which consist of:

- **id**: Must be equal to the ID attribute of the HTML element you would like to apply the image on. E.g. if the HTML element looks like this: `<div id="product_img_area"></div>` then the `id` is equal to `product_img_area`.
- **source**: Image path relative to the Markdown file itself. E.g. `img/product_img.png`
- **inline_css**: CSS properties you would like to apply to the HTML element. E.g. `height: 200px; width: 200px; top: 10mm;`

Example:

```
cover_image:
    id: product_img_area
    source: "img/product_img.png" # relative to the MD file itself
    inline_css: "height: 200px; width: 200px; top: 10mm"
```

Note

- The plugin adds the image by:
  - using the value you set as `id` to select the element you want to apply the image to. **NB:** You must have an HTML element with an ID attribute equal to `id` in your cover template.
  - applying the CSS styles you set as `inline_css` to the HTML element using the inline CSS style attribute (i.e. `style='...'`).
  - then applying the value you set as `source` to the HTML element's style attribute using the CSS `background-image` property. E.g. `background-image: url('source')`.
- Also, the plugin adds some default CSS properties such as `position: absolute; background-size: contain; background-repeat: no-repeat;` to the HTML element's style attribute.