migrating-to-v7

NAME

migrating-to-v7 - Migrating to v7

DESCRIPTION

Version 7 modernizes MkDocs Exporter: pages are rendered by a new engine, documents get bookmarks, books (combined documents) can have a table of contents and working links, and many long-standing issues are fixed. This page lists what you might need to change when upgrading from v6.

SECTIONS

01 Requirements
02 Builds
03 Paged.js
04 Sass
05 Cover pages
06 Books (combined documents)
07 Front matter
08 Content of documents only
09 While serving
10 Links

SEE ALSO

continuous-integration, performance, zensical, troubleshooting

Migrating to v7

Version 7 modernizes *MkDocs Exporter*: pages are rendered by a new engine, documents get bookmarks, books (combined documents) can have a table of contents and working links, and many long-standing issues are fixed. This page lists what you might need to change when upgrading from v6.

Requirements

- Python `>= 3.11.5` (3.9 and 3.10 have reached their end of life).
- MkDocs `>= 1.6, < 2` (or ProperDocs).
- Playwright `>= 1.53`, with Chromium 138 or later (also with `formats.pdf.browser.channel`): older versions can't measure books, whose page numbers and bookmarks would be wrong.
- v6's instructions installed Google Chrome (`playwright install chrome`), which v7 doesn't use unless `formats.pdf.browser.channel: chrome` is set: install Playwright's Chromium with `playwright install --with-deps --only-shell chromium`.

Builds

- Documents are rendered twice as many at a time as there are CPUs, up to 8 (4 before): set `formats.pdf.concurrency: 4` if your CI runs out of memory (the browser uses about 1.4 GB at 8 at a time).
- Unchanged documents are reused from a cache, in `.cache/plugin/exporter` next to `mkdocs.yml`: add `.cache/` to your `.gitignore`, and keep the directory between CI runs to only render the pages that changed (see Continuous integration).

Paged.js

Pages are now laid out by Chromium itself, which is faster and supports more CSS (nesting, `@page` rules in media queries, different page sizes), instead of the bundled Paged.js polyfill. Page sizes, margins, margin boxes with page counters, named pages and `@page :first` keep working.

- Running headers written with `string-set` / `string()`: use the `header` and `footer` options (`{section}`, `{page_title}`) instead.
- Other Paged.js features (`target-counter`, footnotes, `running()`, `:blank`, left/right page breaks, `Paged.registerHandlers`): load Paged.js yourself.
- Pagination may differ slightly, as layouts are computed by another engine.
- Elements with `position: fixed` are repeated on every page when they belong to the body (its children, `body::before` and `body::after`); fixed elements inside the page's content are only printed

on its first page.

- Backgrounds of `@page` rules are printed: when one is set (e.g. by a cover page), the website's own background is left out of documents, give it to pages with `@page { background: ... }` if needed.

Sass

Stylesheets are no longer compiled with Sass (LibSass has reached its end of life): a `.scss` or `.sass` file in `formats.pdf.stylesheets` is now a configuration error.

- Stylesheets only using nesting work as plain CSS: rename them to `.css`.
- Other Sass features (variables, mixins, functions...) have to be compiled beforehand, with Dart Sass for instance: `sass resources/stylesheets/pdf.scss resources/stylesheets/pdf.css`.

Cover pages

Cover pages are now Jinja templates rendered by the plugin, inserted into PDF documents only:

- Templates no longer require the macros plugin. When it is enabled, its environment is still used, so existing templates keep working.
- Templates are no longer processed as Markdown: use HTML instead of Markdown syntax and emoji shortcodes.
- Cover pages are no longer part of the website's pages (they were hidden from them).
- The back cover comes after footnotes.
- Covers sized to fill the page (e.g. with a fixed height) may overflow onto a blank page: end the front cover with `break-after: page`, and draw full-page backgrounds with `@page :first` instead (see Cover pages).
- Elements are moved elsewhere in the document (e.g. styles to its head) with the `data-mkdocs-exporter-teleport` attribute instead of `data-teleport`, which other tools may use. `data-teleport` still works, with a warning.

Books (combined documents)

Books are no longer made by rendering every page again and merging the documents: the pages are combined into a single HTML document, printed once (see How books are printed).

- `aggregator` is deprecated (it still works, with a warning): books are listed in `books`, with the same options (`enabled` is `true` by default, `output` is `book.pdf`).

  === "v6"

  ```
  ```yaml
  plugins:
  ```
  ```

```
    - exporter:
        formats:
          pdf:
            aggregator:
              enabled: true
              output: combined.pdf
              covers: limits
```

=== "v7"

```
    ```yaml
    plugins:
      - exporter:
          formats:
            pdf:
              books:
                - output: combined.pdf
                  covers: limits
    ```
```

- Identifiers of the pages' content are prefixed by page (e.g. `#installation` becomes `#p2-installation`): stylesheets and scripts targeting them don't match in books.
- Scripts of `formats.pdf.scripts` run once per book, and stylesheets or scripts loaded by some pages only are loaded once.
- `counter(page)` and `counter(pages)` in your stylesheets count the pages of the book.
- `@page :first` rules of your stylesheets apply to the pages of every front cover.

Front matter

The options of pages moved under the `exporter` key of their front matter, since other plugins may use the same top-level keys:

v6

```yaml
---
pdf: true
covers:
  front: covers/special.html.j2
---
```

v7

```yaml
---
exporter:
  pdf:
    enabled: true
    covers:
```

```
      front: covers/special.html.j2
---
```

Likewise, `hide: [covers]` becomes `exporter.pdf.covers.enabled: false` and `buttons` becomes `exporter.buttons` (see the front matter reference). `pdf-icon` is deprecated too: set the `icon` of the download button in `buttons` (the same for every page). The previous keys still work, with a warning: builds run with `--strict` fail until pages use `exporter`. Options are now checked: a misspelled option or a value of the wrong type fails the build.

Content of documents only

Elements meant for documents only are given the `mkdocs-exporter-only` class instead of a `data-decompose="true"` attribute (still supported, with a warning):

```
This paragraph is only printed.
{ .mkdocs-exporter-only }
```

While serving

Documents are no longer generated by `mkdocs serve`. Set `formats.pdf.serve: true` to restore the previous behavior.

Links

- Links to other pages are resolved from the page's location: `formats.pdf.url` now replaces the root of the website (it used to keep the path of `site_url`).
- `mailto:` and other non-HTTP links are left untouched.
- Links between pages of books point within the book.

Themes

- Themes other than `material` and `readthedocs` no longer fail: generic adjustments are used.
- The theme's header, navigation and sidebars are hidden from documents once its scripts have run: Material's scripts now work in documents (e.g. diagrams are rendered by Material, like on the website).
- Permalinks (`¶`) are no longer printed.

Python API

The internal modules (`mkdocs_exporter.preprocessor`, `mkdocs_exporter.renderer`, `mkdocs_exporter.themes`, `mkdocs_exporter.formats.pdf.plugin`...) were replaced by

`mkdocs_exporter.core`. The `exporter-pdf` plugin is no longer registered. The callables meant for your configuration (`mkdocs_exporter.formats.pdf.buttons.download.*`) are unchanged, but `download.icon`, which read `pdf-icon`, is deprecated.

See the changelog for every change of version 7.

COLOPHON

This document is a page of https://adrienbrignon.github.io/mkdocs-exporter/, printed by MkDocs Exporter 7.0.0 with Chromium.