docs/customization.md for @host.today/ht-docker-mkdocs

Customization

A great starting point

Project documentation is as diverse as the projects themselves and the Material theme is a good starting point for making it look great. However, as you write your documentation, you may reach a point where some small adjustments are necessary to preserve the desired style.

Adding assets

MkDocs provides several ways to interfere with themes. In order to make a few tweaks to an existing theme, you can just add your stylesheets and JavaScript files to the `docs` directory.

Additional stylesheets

If you want to tweak some colors or change the spacing of certain elements, you can do this in a separate stylesheet. The easiest way is by creating a new stylesheet file in your `docs` directory:

```
mkdir docs/stylesheets
touch docs/stylesheets/extra.css
```

Then, add the following line to your `mkdocs.yml`:

```
extra_css:
  - 'stylesheets/extra.css'
```

Spin up the development server with `mkdocs serve` and start typing your changes in your additional stylesheet file – you can see them instantly after saving, as the MkDocs development server implements live reloading.

Additional JavaScript

The same is true for additional JavaScript. If you want to integrate another syntax highlighter or add some custom logic to your theme, create a new JavaScript file in your `docs` directory:

```
mkdir docs/javascripts
touch docs/javascripts/extra.js
```

Then, add the following line to your `mkdocs.yml`:

```
extra_javascript:
  - 'javascripts/extra.js'
```

Further assistance can be found in the MkDocs documentation.

Extending the theme

If you want to alter the HTML source (e.g. add or remove some part), you can extend the theme.

From version 0.16 on MkDocs implements theme extension, an easy way to override parts of a theme without forking and changing the main theme.

Setup and theme structure

Reference the Material theme as usual in your `mkdocs.yml`, and create a new folder for overrides, e.g. `theme`, which you reference using `custom_dir`:

```
theme:
  name: 'material'
  custom_dir: 'theme'
```

!!! warning "Theme extension prerequisites"

```
As the `custom_dir` variable is used for the theme extension process, the
Material theme needs to be installed via `pip` and referenced with the
`name` parameter in your `mkdocs.yml`.
```

The structure in the theme directory must mirror the directory structure of the original theme, as any file in the theme directory will replace the file with the same name which is part of the original theme. Besides, further assets may also be put in the theme directory.

The directory layout of the Material theme is as follows:

```
.
├─ assets/
│  ├─ images/                    # Images and icons
│  ├─ javascripts/               # JavaScript
│  └─ stylesheets/               # Stylesheets
├─ partials/
│  ├─ integrations/              # 3rd-party integrations
│  ├─ language/                  # Localized languages
│  ├─ footer.html                # Footer bar
│  ├─ header.html                # Header bar
│  ├─ hero.html                  # Hero teaser
│  ├─ language.html              # Localized labels
│  ├─ nav-item.html              # Main navigation item
│  ├─ nav.html                   # Main navigation
│  ├─ search.html                # Search box
│  ├─ social.html                # Social links
│  ├─ source.html                # Repository information
│  ├─ tabs-item.html             # Tabs navigation item
│  ├─ tabs.html                  # Tabs navigation
│  ├─ toc-item.html              # Table of contents item
│  └─ toc.html                   # Table of contents
├─ 404.html                      # 404 error page
├─ base.html                     # Base template
└─ main.html                     # Default page
```

Overriding partials

In order to override the footer, we can replace the `footer.html` partial with our own partial. To do this, create the file `partials/footer.html` in the theme directory. MkDocs will now use the new partial when rendering the theme. This can be done with any file.

Overriding template blocks

Besides overriding partials, one can also override so called template blocks, which are defined inside the Material theme and wrap specific features. To override a template block, create a `main.html` inside the theme directory and define the block, e.g.:

```
{% extends "base.html" %}

{% block htmltitle %}
  <title>Lorem ipsum dolor sit amet</title>
{% endblock %}
```

The Material theme provides the following template blocks:

Block name	Wrapped contents
`analytics`	Wraps the Google Analytics integration
`content`	Wraps the main content
`disqus`	Wraps the disqus integration
`extrahead`	Empty block to define additional meta tags
`fonts`	Wraps the webfont definitions
`footer`	Wraps the footer with navigation and copyright
`header`	Wraps the fixed header bar
`hero`	Wraps the hero teaser
`htmltitle`	Wraps the `<title>` tag
`libs`	Wraps the JavaScript libraries, e.g. Modernizr
`scripts`	Wraps the JavaScript application logic
`source`	Wraps the linked source files
`site_meta`	Wraps the meta tags in the document head
`site_nav`	Wraps the site navigation and table of contents
`styles`	Wraps the stylesheets (also extra sources)

For more on this topic refer to the MkDocs documentation

Theme development

The Material theme uses Webpack as a build tool to leverage modern web technologies like Babel and SASS. If you want to make more fundamental changes, it may be necessary to make the adjustments directly in the source of the Material theme and recompile it. This is fairly easy.

Environment setup

In order to start development on the Material theme, a Node.js version of at least 8 is required. First, clone the repository:

```
git clone https://github.com/squidfunk/mkdocs-material
```

Next, all dependencies need to be installed, which is done with:

```
cd mkdocs-material
pip install -r requirements.txt
npm install
```

Development mode

The development server can be started with:

```
npm run watch
```

This will also start the MkDocs development server which will monitor changes on assets, templates and documentation. Point your browser to localhost:8000 and you should see this documentation in front of you.

For example, changing the color palette is as simple as changing the `$md-color-primary` and `$md-color-accent` variables in `src/assets/stylesheets/_config.scss`:

```
$md-color-primary: $clr-red-400;
$md-color-accent:  $clr-teal-a700;
```

!!! warning "Automatically generated files"

```
Never make any changes in the `material` directory, as the contents of this
directory are automatically generated from the `src` directory and will be
overridden when the theme is built.
```

Build process

When you've finished making your changes, you can build the theme by invoking:

```
npm run build
```

This triggers the production-level compilation and minification of all stylesheets and JavaScript sources. When the command exits, the final theme is located in the `material` directory. Add the `theme_dir` variable pointing to the aforementioned directory in your original `mkdocs.yml`.

Now you can run `mkdocs build` and you should see your documentation with your changes to the original Material theme.

Revision #4
Created 2026-03-28 11:09:28 UTC by foss.global Team
Updated 2026-03-28 12:15:41 UTC by foss.global Team