Setting up the Material theme for MkDocs

> Enhance your document with the Material theme.

In this tutorial, we'll cover how to:

1. Set up tooltips with definitions for concepts.

2. Add a footer.

3. Add a PDF export button.

4. Set up a site footer.

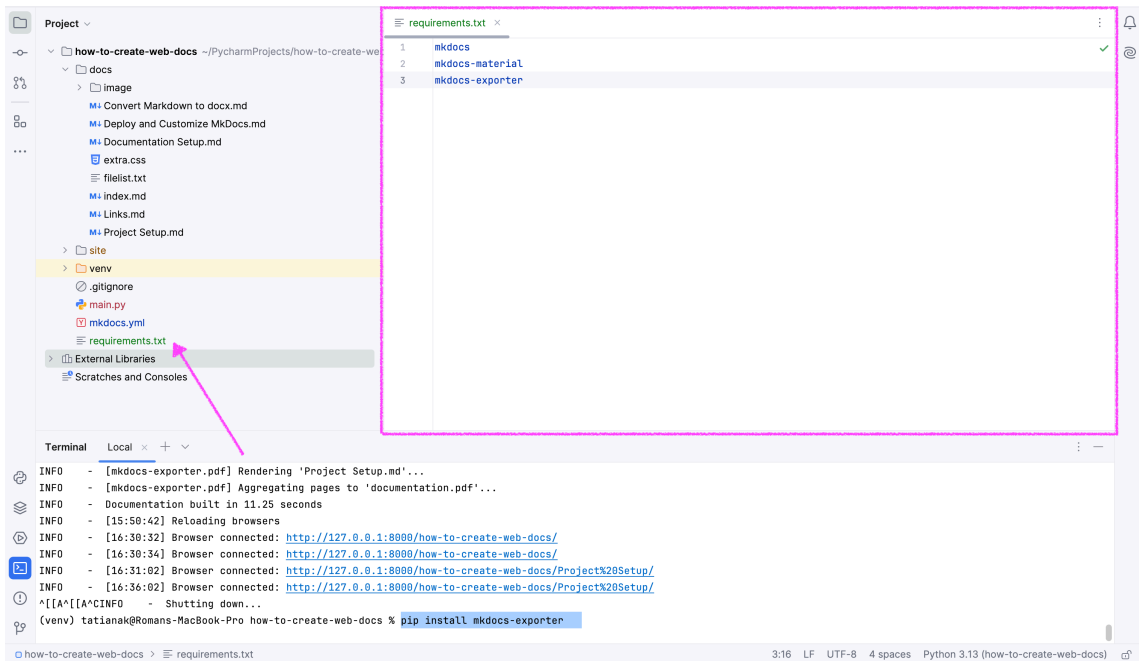5. Change the site color scheme and add styles.

Reconfiguring a virtual environment

If you encounter errors while setting up a theme, follow these steps:

1. Delete the old virtual environment if you created one.

2. Create a new one:

   - In the IDE panel select `Add New Interpreter` > `Add Local Interpreter`.

   - Specify the settings and confirm.

3. Activate the environment: `source venv/bin/activate`

4. Add a `requirements.txt` file to the root of the project with the following content:

   - mkdocs

   - mkdocs-material

   - mkdocs-exporter

> Why this file? It simplifies dependency management, allowing you to quickly install all the necessary libraries with a single command.

5. Install dependencies: `pip install -r requirements.txt`

6. If the `mkdocs-exporter` library is not installed, run the command `pip install mkdocs-exporter`.

MkDocs Configuration

This config is suitable for a basic project with an emphasis on interface customization (color scheme, custom footer) and support for exporting documentation to PDF. Here is the `mkdocs.yml` configuration file with explanations:

Basic settings

```
site_name: Web-docs using MkDocs
site_url: https://MkDocs-web-docs.github.io/how-to-create-web-docs/
```

- **site_name:** The name of your documentation site. It appears in the browser title and site header.
- **site_url:** The base URL for generating links when building a static site.

Navigation

```yaml
nav:
  - Creating a web manual using MkDocs: index.md
  - Links: Links.md
  - Project Setup: Project Setup.md
  - Documentation Setup: Documentation Setup.md
  - Deploy and Customize: Deploy and Customize MkDocs.md
  - Convert Markdown to docx: Convert Markdown to docx.md
  - Glossary: Glossary.md
```

> Navigation defines the menu structure. Each line links the section title to a Markdown file.

Theme settings

```yaml
theme:
  palette:
    primary: pink
    accent: lime
  name: material
  custom_dir: overrides
  language: en
  features:
    - navigation.instant
    - header.autohide
    - content.tooltips
  hide:
    - footer
```

- **palette:** sets the color scheme.

    - **primary:** the main color (e.g. header and buttons).

    - **accent:** accent color (e.g. links).

- **custom_dir:** specifies the folder with custom templates (e.g. for changing the footer).

- **features:**

    - **navigation.instant:** instant navigation without reloading the page.

    - **header.autohide:** hide the header when scrolling down.

    - **content.tooltips:** improved tooltips for the title attribute.

- **hide.footer:** hides the standard footer.

Plugins

```
plugins:
  - exporter:
      formats:
        pdf:
          enabled: true
          aggregator:
            enabled: true
            output: documentation.pdf
            covers: all
      buttons:
        - title: Сохранить в PDF
          icon: material-file-download-outline
  - search
```

- **exporter:** adds PDF export functionality. The button with the text "Save to PDF" will be available on the site.

- **search:** enables built-in search.

Markdown Extensions

```
markdown_extensions:
  - abbr
  - attr_list
  - pymdownx.snippets
```

- **abbr:** allows you to specify abbreviations.

- **attr_list:** adds attribute support for Markdown elements.

- **pymdownx.snippets:** allows you to reuse certain Markdown blocks.

Adding tooltips

1. Add the following configuration to your mkdocs.yml file to enable tooltips: `yaml`
   ```
   theme:
     features:
       - content.tooltips
   ```

2. Add the title attribute to the Markdown text:
   Example:
   `markdown`
   ```
   [Ctrl+C](# "A keyboard shortcut used to stop a terminal process")
   ```

This code will add a tooltip when you hover over a concept, and it will display its definition.

➤ The site will automatically reload when you make changes in PyCharm.

➤ To stop the server, press Ctrl+C on your keyboard.

A keyboard shortcut used to stop a terminal process, such as a local MkDocs server

Deploy to GitHub Pages.

> The title attribute will not work inside a code block.

3. Check the result.
   When you run the site with mkdocs serve, you will be able to see the tooltip that appears when you hover over `Ctrl+C` .

> Tooltip display is not shown in PyCharm preview

Removing the footer

Next, remove the footer if necessary. The instructions indicate a simple method, which is described here.

If the standard method did not work, use the Material theme customization:

1. Create an overrides folder: `mkdir overrides`

2. Create a footer.html file: `mkdir -p overrides/partials` `echo "" > overrides/partials/footer.html`

3. In the config, specify the custom_dir directory:

```
theme:
  name: material
  custom_dir: overrides
```

Now the footer will be completely removed.

Changing the color scheme

To change the colors, use the instructions Changing the colors.

In the config, change the parameters palette:

> My example is pink)

```
palette:
  primary: pink
  accent: lime
```

> The material is an amazing theme with very clear and concise instructions. Imagine that from boring Word you can now create beautiful web documents simply and easily.

For take this example, you can just copy this config:

```yaml
site_name: Web-docs using MkDocs
site_url: https://MkDocs-web-docs.github.io/how-to-create-web-docs/

nav:
  - Creating a web manual using MkDocs: index.md
  - Links: Links.md
  - Project Setup: Project Setup.md
  - Documentation Setup: Documentation Setup.md
  - Deploy and Customize: Deploy and Customize MkDocs.md
  - Convert Markdown to docx: Convert Markdown to docx.md
  - Material theme for MkDocs: Material theme for MkDocs.md
  - Glossary: Glossary.md

theme:
  palette:
    primary: pink
    accent: lime
  name: material
  custom_dir: overrides
  language: en
  features:
    - navigation.instant
    - header.autohide
    - content.tooltips
  hide:
    - footer

plugins:
  - exporter:
      formats:
        pdf:
          enabled: true
          aggregator:
            enabled: true
            output: documentation.pdf
            covers: all
      buttons:
        - title: Save to PDF
          icon: material-file-download-outline
          enabled:
!!python/name:mkdocs_exporter.formats.pdf.buttons.download.enabled
          attributes:
!!python/name:mkdocs_exporter.formats.pdf.buttons.download.attributes
  - search

markdown_extensions:
  - abbr
  - attr_list
  - pymdownx.snippets

extra: {}
```

Now our project structure looks this:

```
how-to-create-web-docs/
├── docs/ # Directory with Markdown documentation files
| ├── Convert Markdown to docx.md
| ├── Deploy and Customize MkDocs.md
| ├── Documentation Setup.md
| ├── extra.css # Custom styles (if applicable)
| ├── filelist.txt # Auxiliary file (e.g. file list)
| ├── Glossary.md
| ├── index.md
| ├── Links.md
| ├── Material theme for MkDocs.md
| ├── Project Setup.md
├── image/ # Image folder
├── overrides/ # Template customization
| ├── partials/ # Template parts
|| ├── footer.html # Custom footer
├── site/ # Generated site (created automatically)
├── venv/ # Python virtual environment
├── .gitignore # File to exclude from repository
├── mkdocs.yml # Main MkDocs configuration file
├── requirements.txt # Project dependencies file
```