

The title of the report on two lines

The subtitle of the report

Something on the right

1.1. MkDocs

Project documentation with Markdown.

MkDocs is a **fast**, **simple** and **downright gorgeous** static site generator that's geared towards building project documentation. Documentation source files are written in Markdown, and configured with a single YAML configuration file. Start by reading the introductory tutorial, then check the User Guide for more information.

Getting Started User Guide

1.1.1. Features

1.1.1.1. Great themes available

There's a stack of good looking themes available for MkDocs. Choose between the built in themes: `mkdocs` and `readthedocs`, select one of the third-party themes listed on the [MkDocs Themes wiki page](#), or build your own.

1.1.1.2. Easy to customize

Get your project documentation looking just the way you want it by customizing your theme and/or installing some plugins. Modify Markdown's behavior with Markdown extensions. Many configuration options are available.

1.1.1.3. Preview your site as you work

The built-in dev-server allows you to preview your documentation as you're writing it. It will even auto-reload and refresh your browser whenever you save your changes.

1.1.1.4. Host anywhere

MkDocs builds completely static HTML sites that you can host on GitHub pages, Amazon S3, or anywhere else you choose.

1.2. Getting Started with MkDocs

An introductory tutorial!

1.2.1. Installation

To install MkDocs, run the following command from the command line:

```
pip install mkdocs
```

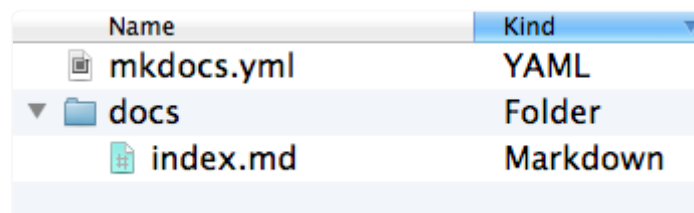
For more details, see the [Installation Guide](#).

1.2.2. Creating a new project

Getting started is super easy. To create a new project, run the following command from the command line:

```
mkdocs new my-project  
cd my-project
```

Take a moment to review the initial project that has been created for you.






Name	Kind
 mkdocs.yml	YAML
 docs	Folder
 index.md	Markdown

Figure 1 - The initial MkDocs layout

There's a single configuration file named `mkdocs.yml`, and a folder named `docs` that will contain your documentation source files (`docs` is the default value for the `docs_dir` configuration setting). Right now the `docs` folder just contains a single documentation page, named `index.md`.

MkDocs comes with a built-in dev-server that lets you preview your documentation as you work on it. Make sure you're in the same directory as the `mkdocs.yml` configuration file, and then start the server by running the `mkdocs serve` command:

```
$ mkdocs serve
INFO      - Building documentation...
INFO      - Cleaning site directory
[I 160402 15:50:43 server:271] Serving on http://127.0.0.1:8000
[I 160402 15:50:43 handlers:58] Start watching changes
[I 160402 15:50:43 handlers:60] Start detecting changes
```

Open up `http://127.0.0.1:8000/` in your browser, and you'll see the default home page being displayed:

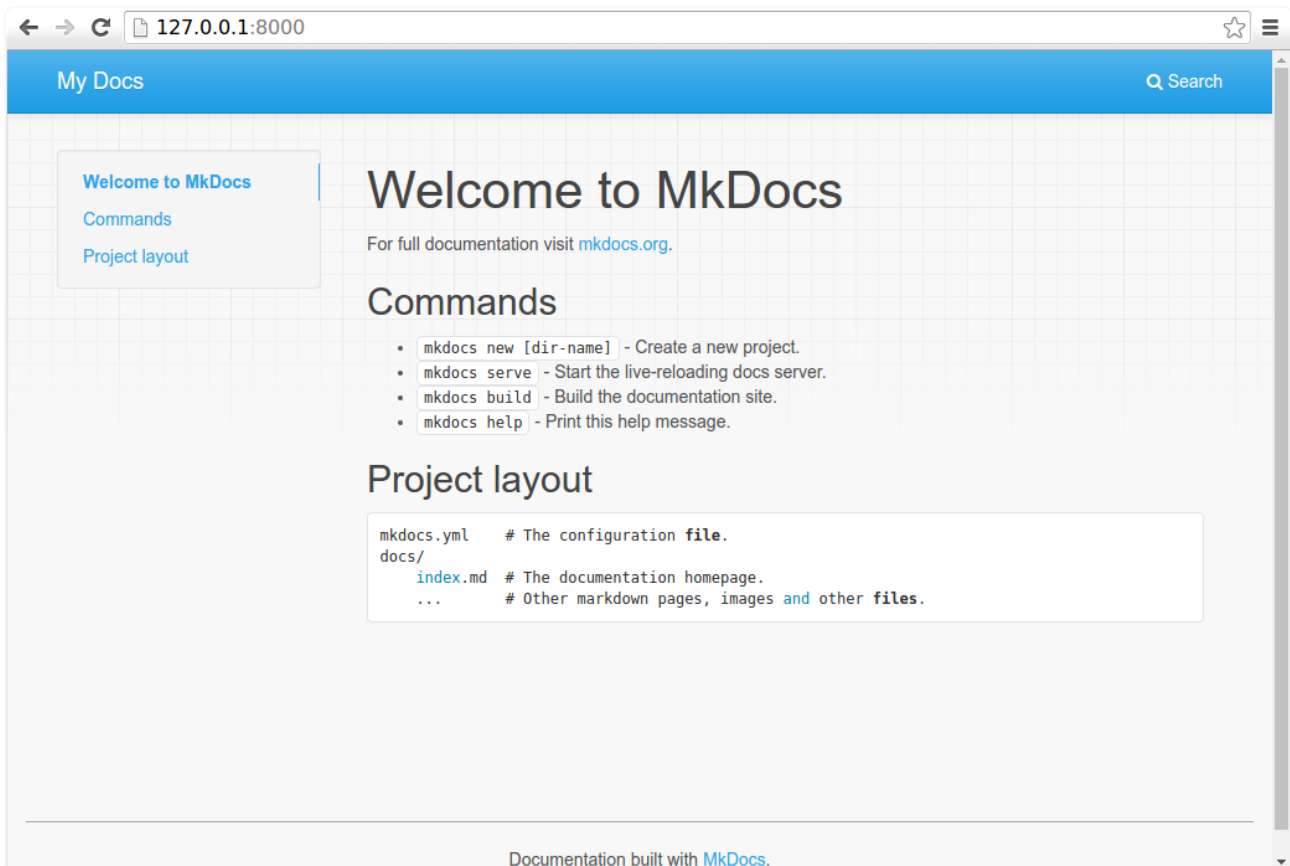


Figure 2 - The MkDocs live server

The dev-server also supports auto-reloading, and will rebuild your documentation whenever anything in the configuration file, documentation directory, or theme directory changes.

Open the `docs/index.md` document in your text editor of choice, change the initial heading to `MkLorum`, and save your changes. Your browser will auto-reload and you should see your updated documentation immediately.

Now try editing the configuration file: `mkdocs.yml`. Change the `site_name` setting to `MkLorum` and save the file.

```
site_name: MkLorum
site_url: https://example.com/
```

Your browser should immediately reload, and you'll see your new site name take effect.

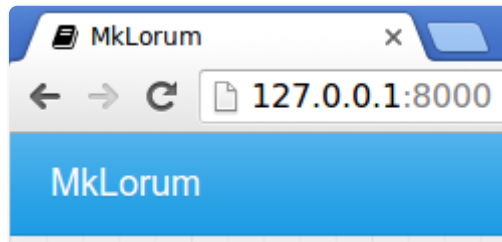


Figure 3 - The `site_name` setting

Note

The `site_name` and `site_url` configuration options are the only two required options in your configuration file. When you create a new project, the `site_url` option is assigned the placeholder value: `https://example.com`. If the final location is known, you can change the setting now to point to it. Or you may choose to leave it alone for now. Just be sure to edit it before you deploy your site to a production server.

1.2.3. Adding pages

Now add a second page to your documentation:

```
curl 'https://jaspervdj.be/lorem-markdownum/markdown.txt' > docs/about.md
```

As our documentation site will include some navigation headers, you may want to edit the configuration file and add some information about the order, title, and nesting of each page in the navigation header by adding a `nav` setting:

```
site_name: MkLorum
site_url: https://example.com/
nav:
  - Home: index.md
  - About: about.md
```

Save your changes and you'll now see a navigation bar with `Home` and `About` items on the left as well as `Search`, `Previous`, and `Next` items on the right.

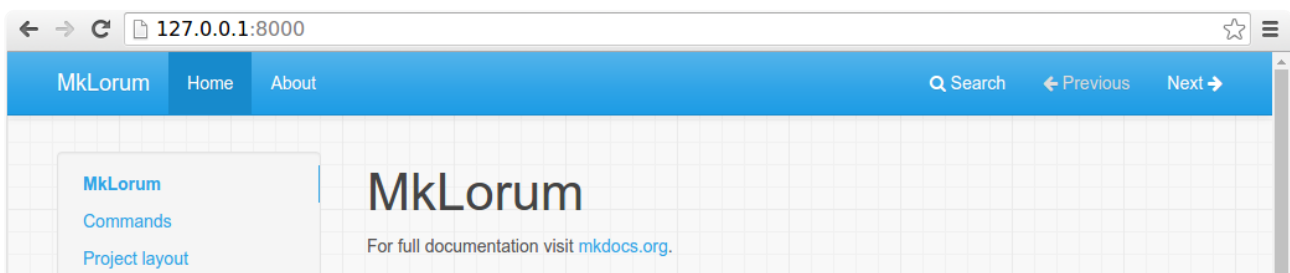


Figure 4 - Screenshot

Try the menu items and navigate back and forth between pages. Then click on `Search`. A search dialog will appear, allowing you to search for any text on any page. Notice that the search results include every occurrence of the search term on the site and links directly to the section of the page in which the search term appears. You get all of that with no effort or configuration on your part!

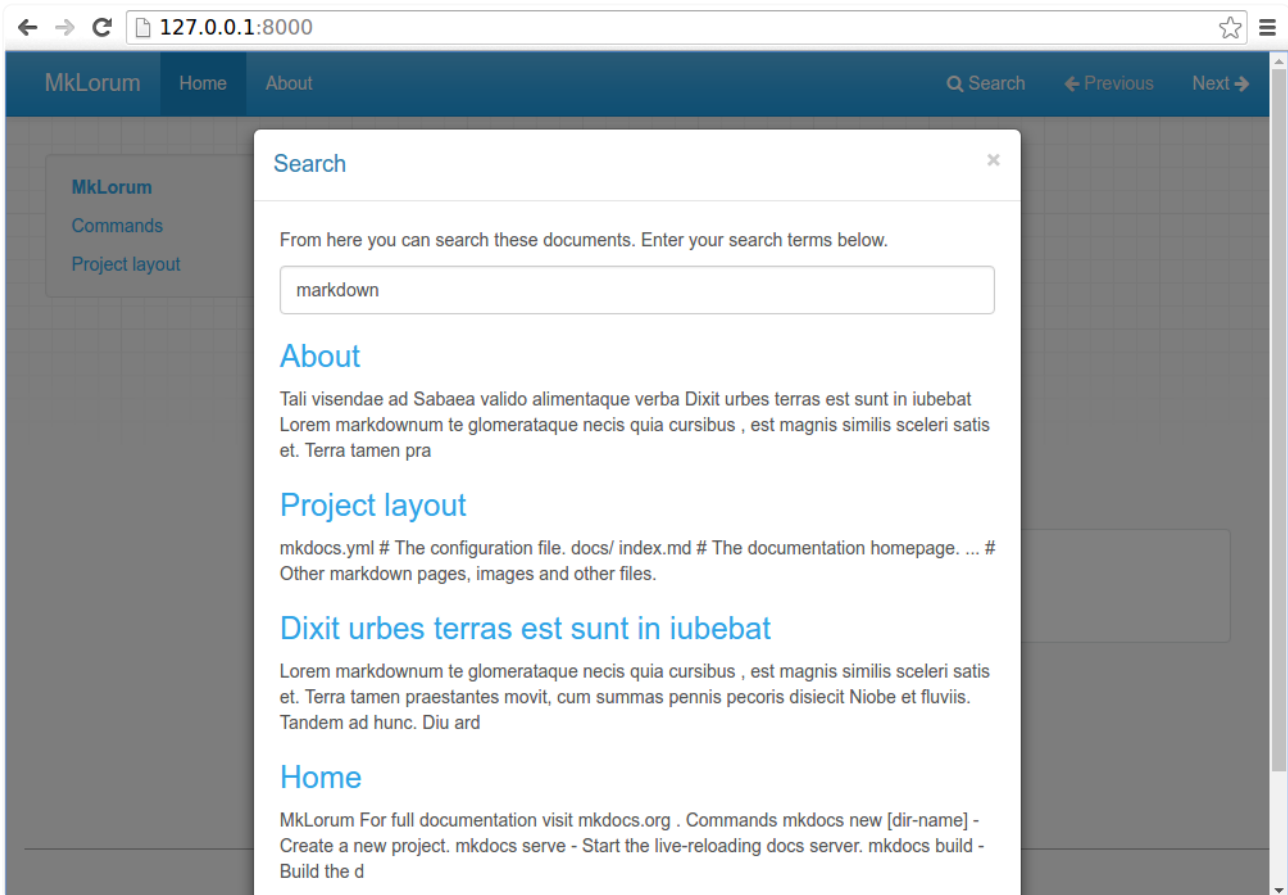


Figure 5 - Screenshot

1.2.4. Theming our documentation

Now change the configuration file to alter how the documentation is displayed by changing the theme. Edit the `mkdocs.yml` file and add a `theme` setting:

```
site_name: MkLorum
site_url: https://example.com/
nav:
  - Home: index.md
  - About: about.md
theme: readthedocs
```

Save your changes, and you'll see the ReadTheDocs theme being used.

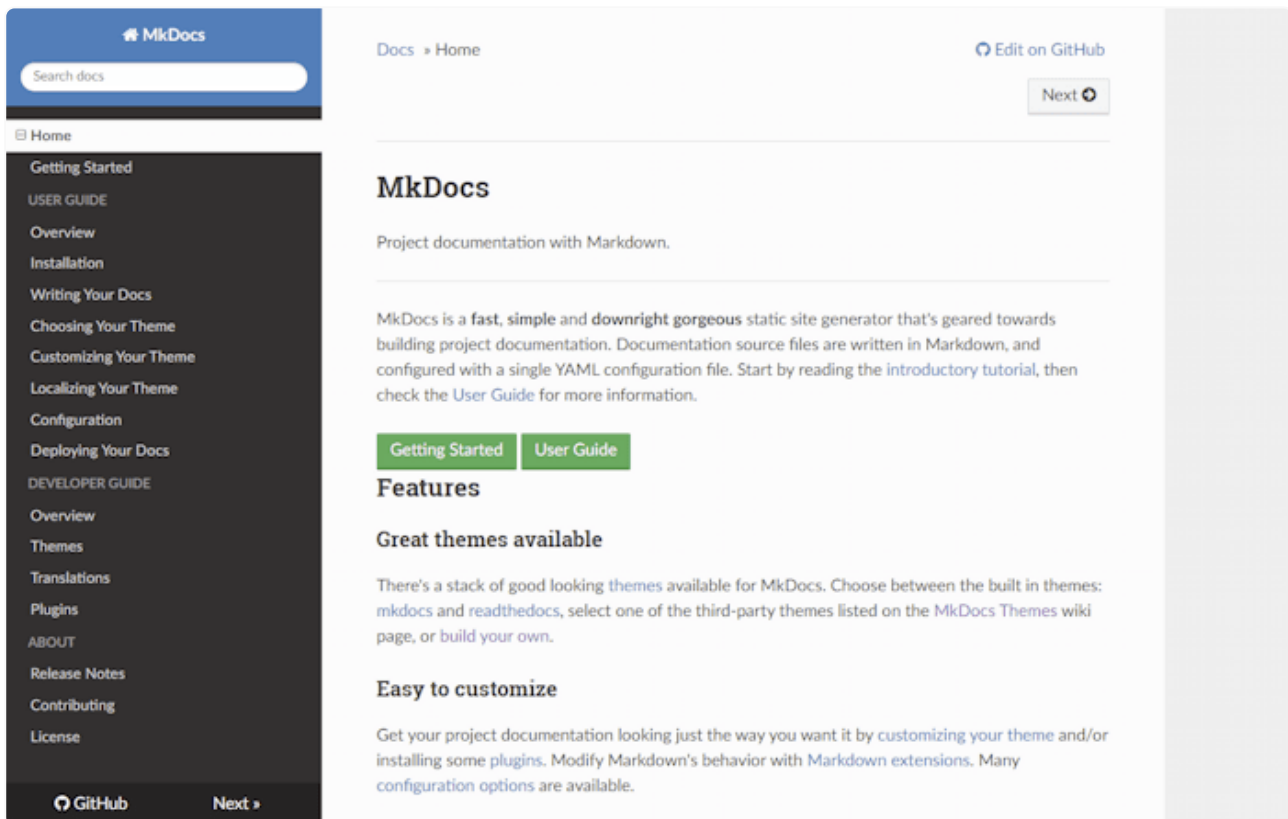


Figure 6 - Screenshot

1.2.5. Changing the Favicon Icon

By default, MkDocs uses the MkDocs favicon icon. To use a different icon, create an `img` subdirectory in the `docs` directory and copy your custom `favicon.ico` file to that directory. MkDocs will automatically detect and use that file as your favicon icon.

1.2.6. Building the site

That's looking good. You're ready to deploy the first pass of your `MkLorum` documentation. First build the documentation:

```
mkdocs build
```

This will create a new directory, named `site`. Take a look inside the directory:

```
$ ls site
about  fonts  index.html  license  search.html
css    img    js          mkdocs   sitemap.xml
```

Notice that your source documentation has been output as two HTML files named `index.html` and `about/index.html`. You also have various other media that's been copied into the `site`

directory as part of the documentation theme. You even have a `sitemap.xml` file and `mkdocs/search_index.json`.

If you're using source code control such as `git` you probably don't want to check your documentation builds into the repository. Add a line containing `site/` to your `.gitignore` file.

```
echo "site/" >> .gitignore
```

If you're using another source code control tool you'll want to check its documentation on how to ignore specific directories.

1.2.7. Other Commands and Options

There are various other commands and options available. For a complete list of commands, use the `--help` flag:

```
mkdocs --help
```

To view a list of options available on a given command, use the `--help` flag with that command. For example, to get a list of all options available for the `build` command run the following:

```
mkdocs build --help
```

1.2.8. Deploying

The documentation site that you just built only uses static files so you'll be able to host it from pretty much anywhere. Simply upload the contents of the entire `site` directory to wherever you're hosting your website from and you're done. For specific instructions on a number of common hosts, see the [Deploying your Docs](#) page.

1.2.9. Getting help

See the [User Guide](#) for more complete documentation of all of MkDocs' features.

To get help with MkDocs, please use the [GitHub discussions](#) or [GitHub issues](#).

User Guide

1.1. User Guide

Building Documentation with MkDocs

The MkDocs Developer Guide provides documentation for users of MkDocs. See [Getting Started](#) for an introductory tutorial. You can jump directly to a page listed below, or use the next and previous buttons in the navigation bar at the top of the page to move through the documentation in order.

- [Installation](#)
- [Writing Your Docs](#)
- [Choosing Your Theme](#)
- [Customizing Your Theme](#)
- [Configuration](#)
- [Deploying Your Docs](#)

1.2. MkDocs Installation

A detailed guide.

1.2.1. Requirements

MkDocs requires a recent version of Python and the Python package manager, pip, to be installed on your system.

You can check if you already have these installed from the command line:

```
$ python --version
Python 3.8.2
$ pip --version
pip 20.0.2 from /usr/local/lib/python3.8/site-packages/pip (python 3.8)
```

If you already have those packages installed, you may skip down to Installing MkDocs.

1.2.1.1. Installing Python

Install Python using your package manager of choice, or by downloading an installer appropriate for your system from python.org and running it.

Note

If you are installing Python on Windows, be sure to check the box to have Python added to your PATH if the installer offers such an option (it's normally off by default).

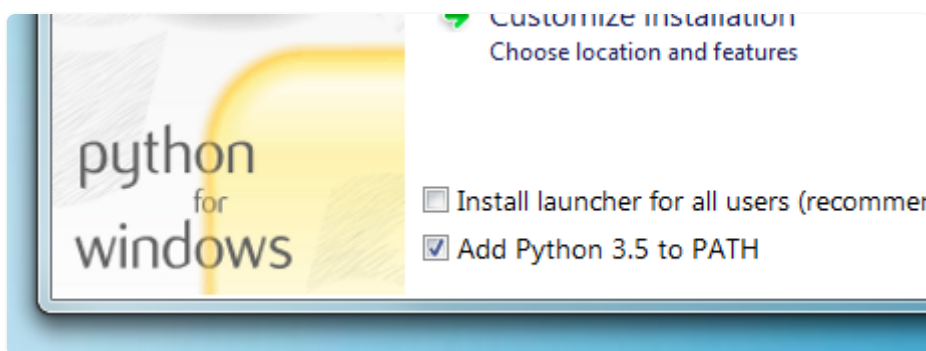


Figure 7 - Add Python to PATH

1.2.1.2. Installing pip

If you're using a recent version of Python, the Python package manager, pip, is most likely installed by default. However, you may need to upgrade pip to the latest version:

```
pip install --upgrade pip
```

If you need to install pip for the first time, download `get-pip.py`. Then run the following command to install it:

```
python get-pip.py
```

1.2.2. Installing MkDocs

Install the `mkdocs` package using pip:

```
pip install mkdocs
```

You should now have the `mkdocs` command installed on your system. Run `mkdocs --version` to check that everything worked okay.

```
$ mkdocs --version
mkdocs, version 1.2.0 from /usr/local/lib/python3.8/site-packages/mkdocs (Python 3.8
```

Note

If you would like manpages installed for MkDocs, the `click-man` tool can generate and install them for you. Simply run the following two commands:

```
pip install click-man
click-man --target path/to/man/pages mkdocs
```

See the [click-man documentation](#) for an explanation of why manpages are not automatically generated and installed by pip.

Note

If you are using Windows, some of the above commands may not work out-of-the-box.

A quick solution may be to preface every Python command with `python -m` like this:

```
python -m pip install mkdocs
python -m mkdocs
```

For a more permanent solution, you may need to edit your `PATH` environment variable to include the `Scripts` directory of your Python installation. Recent versions of Python include a script to do this for you. Navigate to your Python installation directory (for example `C:\Python38\`), open the `Tools`, then `Scripts` folder, and run the `win_add2path.py` file by double clicking on it. Alternatively, you can download the [script](#) and run it (`python win_add2path.py`).

Thank for reading :-)