RSE-ops Documentation

Release 1.0.0

RSE-ops and contributors

Oct 18, 2021

GETTING STARTED

CHAPTER

ONE

RSEOPS DOCUMENTATION

1.1 Installation

Note: We recommend installing into a virtual environment, which is an isolated Python environment that allows you to install packages without admin privileges.

1.1.1 Install using pip

The simplest way to install from PyPi is using pip with the command:

```
$ pip install <yourproject>
```

You may need to use this instead, depending on your operating system:

```
$ python -m pip install <yourproject>
```

1.2 Tutorials

New to the project? Then these tutorials should get you up and running.

1.2.1 Contributing a documentation change

This tutorial will take you through the process of:

- Downloading the current documentation
- Installing required libraries
- Building and previewing the documentation
- Creating a new git branch
- Making a change to the documentation
- Committing the changes and making a pull request

Download the documentation

1. Sign up to GitHub and fork the project
2. Install Git. If you're new to Git, the Django project has a good introduction on working with Git and GitHub. You can also take a look at the GitHub branch-based workflow
3. Using the command line, `cd` to the directory where you want your local copy of the software to live. The documentation can then be downloaded using:

```
$ git clone https://github.com/YourUsername/readthedocs-theme.git
```

4. (Optional) We recommend that you install your development copy of the software in a virtual environment.

```
$ python -m venv env
```

5. Install the cloned copy of the software (`-e` for editable or development mode):

```
$ pip install -e project/
```

Install required libraries

```
$ pip install -r requirements.txt
```

Build and preview the documentation

To build the documentation locally, navigate to the `docs` directory and run `make html`:

```
$ cd docs/
$ make html
```

Occasionally you may need to clean up the generated files before a change gets applied:

```
$ make clean && make html
```

The HTML generated by the build can be viewed by opening `_build/html/index.html` in a web browser or by doing:

```
$ cd _build/html
$ python -m http.server 9999
```

And then going to http://localhost:9999

Create a new branch

Create a new branch `my-changes` for your changes (you can choose any name that you want instead). Any changes made in this branch will be specific to it and won't affect the main copy (the `main` branch) of the documentation:

```
$ git checkout -b my-changes
```

Make a change to the documentation

At this point you can make a change to a markdown or rst file in the docs! Let's say we add `reading.md` to the tutorials folder. When you add this new page, make sure that it appears on a toc tree somewhere. If not, you'll see:

```
checking consistency... [/path/to]/readthedocs-theme/docs/tutorials/reading.md: WARNING:␣
↪document isn't included in any toctree
```

Make sure to preview the docs, as shown above, to see that the page is visible. Here is how to include a new page in the docs. Let's say that we are adding the reading.rst file to the tutorials folder. You'dd add it to `tutorials/index.md`, as follows:

```
.. toctree::
   :maxdepth: 1

   installation
   contributing_code
   contributing_docs
   reading
```

If you rebuild the HTML you should find that the warning is gone and that your new page is reachable from the main documentation page. You have the choice to write markdown or rst. If you need help with the reST markup then you can check out Sphinx's reStructuredText primer

There are also a number of directives that tell Sphinx to do certain things (like inserting code blocks or a table of contents). Sphinx has a list of these here.

For more information on writing documentation see *writing documentation*.

Just like before, you should build and preview the updated page. When you're happy with the results move on to the next section.

Commit your changes and make a pull request

First we add our new file to git:

```
$ git add tutorials/reading.md
```

And then stage the remaining changes (`-a`) and commit at the same time:

```
$ git commit -am "Add new reading.md on <x>"
```

After committing the changes, send them to your fork:

```
$ git push origin my-changes
```

You can create a pull request by visiting the project GitHub page where you should see your branch under *"Your recently push branches"*. Click *"Compare & pull request"* and fill out the title (with a `[WIP]` prefix, i.e. `[WIP] Add new reading example for <x>`) and follow the instructions in the main entry window. At this point your code will be tested via continuous integration. If you set up Artifacts preview (e.g., CircleCI) then you should be able to click on the Artifacts tab in the CircleCI interface and click any html file to preview.

What happens next?

One or more reviewers would look at your pull request and may make suggestions, ask for clarification or request changes. Once the reviewers were happy, the pull request would be approved and your changes merged into the `main` branch.

1.3 Guides

These guides contain higher-level information for those already familiar with the project.

1.3.1 Example Folder

This is an example subfolder page!

How to evaluate rst

Column Name	Key	Value	Description
`Attribute`	Name	Value	Hello!

1.3.2 Writing documentation

Types of documentation

- **Tutorials**: take a reader unfamiliar with the project through a series of steps to achieve something useful
- **How-to/examples**: more advanced versions of tutorials, for readers that already have some understanding of how the project.
- **Guides**: aim to explain a subject at a fairly high level

1.4 General examples

Here are some general examples.

1.4.1 This is the title

The title is required for the example to show up. Hello world!

```
print(__doc__)

usage = "Usage: python hello-world.py"

def print_hello_world():
    print("Hello World!")

if __name__ == '__main__':
    print_hello_world()
```

Total running time of the script: (0 minutes 0.000 seconds)

1.5 Contributing to the project

If you're interested in contributing to the project there are many ways to help out:

- Adding new features, improving existing ones or submitting bug fixes for the source code
- Reporting bugs, suggesting improvements or requesting features on the issue tracker.
- Reviewing existing pull requests on GitHub.
- Improving the documentation, whether by fixing typos, contributing new material or adding code examples. No change is too small.

If it's your first time contributing then we have tutorials for:

- *Contributing a documentation change*

1.6 Frequently asked questions

1.6.1 General

What if I don't like the theme?

You can easily change theme assets in the _static folder at the root of the site.

How do I ask for help?

You can open an issue at the readthedocs-theme repository.

1.6.2 Content

How do I label a section?

Like this:

```
(faq-content)=
```

How do I make a table?

A markdown table looke like this:

Fruit	Color	Tasty?
apple	red	yes
grape	purple or green	sometimes
orange	orange	rarely
banana	yellow	no
watermelon	pink and green	sometimes

1.7 Getting Started

Welcome to the project documentation! Good documentation should have:

- installation instructions
- examples and tutorials
- a getting started guide

You can assume that users of all different levels of experience will want to use your software! For example, let's now direct to the *Installation*. Take a look at the markdown file `index.md` to see how we creted this link.

1.8 Examples

A set of examples illustrating the use of the different core elements.