markdown-publishing-guide

unknown

Nov 24, 2021

CONTENTS

CHAPTER

ONE

DOCUMENTS

1.1 Markdown Publishing Guide v1.0.3

A guide to publishing Markdown documents on GitHub and ReadTheDocs

1.1.1 Introduction

This is a guide to writing and publishing Markdown documents on GitHub and ReadTheDocs. It includes steb-by-step instructions for using GitHub, git, Markdown, and ReadTheDocs in order to produce a well-formatted set of documents with a detailed table of contents.

This document itself was created using the instructions in this document.

1.1.2 Motivation

This document was created to help others write documents in Markdown and have them look good on both GitHub and ReadTheDocs. By "good" I mean that on ReadTheDocs it has a table of contents page and there is a left navigation pane with topics that expands hierarchically.

It took a while to figure out how to setup my files on GitHub in a way would produce good results so I decided to document the procedures I came up with. I also included a very basic git workflow to support multiple versions.

These instructions are published with Creative Commons Zero license which basically says you can do what you want with this content and you don't have to add any attributions or do anything really - just copy it.

1.1.3 Scope

These instructions are intended for new, standalone documents like this one, that do not require a sophisticated git branching and collaboration workflow. This workflow does not use pull requests. Editing is done on the master branch. In order to handle developing a new version over multiple edits, the master branch is marked as a "prerelease" until the release is ready, at which point the "prerelease" status is removed. When a release is created, this workflow creates a tag for the new release and merges the release tag to a branch named "latest-release" for convenience. The default ReadTheDocs version will point to the master branch so that when you click the edit button in ReadTheDocs, it takes you to the appropriate branch for editing. Of course, you can also ignore my suggested git workflow and substitute your own in order to support collaboration.

1.1.4 Existing Documents and Troubleshooting

If you have an existing document and would like to add a table of contents, for example, these instructions are not sufficient for that situation. Existing documents are unlikely to have the same document structure (such as a using a docs subdirectory) and often have other settings and extensions which these instructions do not anticipate. This project not have enough resources to identify all (or really any of) the issues that can be present in existing documents that may produces unexpected results.

If you try to adapt these instructions and the document structure it describes to your current document and you run into problems, you might want to consider creating a separate project just for troubleshooting. You would use this new project in order to figure out what the problem is and then you can delete it later. The name could be the same as your original project, with "-test" added to the end, for example. You would also create a corresponding copy of the project over on ReadTheDocs for troubleshooting.

The troubleshooting strategy I will suggest is to start stripping away the differences between your new copy of your project and my simple example until the specific difference causing the problem is identified. Just start guessing as to what it could be.

- If you have an existing conf.py, try removing the extra configuration lines in chunks until it (the RTD index) works or your file is just like mine. You may also need to adapt or remove some of your content if you are relying on that configuration.
- Try removing files in your index.rst until it works or your index is pretty much like mine.
- Try moving files into the docs subdirectory so it is more like my setup.
- Try removing unusual content from your markdown docs.
- Try changing the content to be more like mine.

At this point you are just trying to figure out what difference, if removed, causes the index to show up correctly. This process basically has to work because you either find the problem or you eventually end up with two projects that are essentially identical although one works (mine) and one doesn't (yours). In my experience, that's so rare that it is not worth worrying about until it happens.

Once the difference that is causing the problem is identified, it is likely that there will be a straightforward solution that can be applied back to your original project.

1.1.5 Preparing for Development

Follow the instructions from each provider in order to perform the following tasks.

- Create a GitHub Account.
- Create a ReadTheDocs Account.
- Setup a Connection to GitHub in your ReadTheDocs account.

1.1.6 Setting up a New Project

1.1.6.1 Creating a GitHub Repo with Initial Content

1.1.6.1.1 Create a New Repo in the GitHub Web Interface

- Click "New repository" in plus (+) menu in upper right.
- Enter a short description of your project.
- Check "Initialize this repository with a README"

- For license, select “Creative Commons Zero v1.0 Universal”
- Click Create Repository

1.1.6.1.2 Create docs/user-guide.md with initial content

This document assumes your initial version is v0.0.0 but you can choose a different initial version. If you pick an initial version other than v0.0.0 then replace v0.0.0 throughout the rest of this document with your chosen version.

- Go to the repository main page.
- Click on “New file”.
- Choose a name for your file, such as “user-guide.md”
- Type “docs/user-guide.md” (replacing with your chosen file name) in the file name text box.
- Click in the file content area.
- Add project description and status.
- If you already have content, add it now

```
# Project Name User Guide v0.0.0

## Introduction
Put an introduction here.

## Main Content
Put more content here.
```

- Change commit message.
  - “Create docs/user-guide.md with initial content”
- Click “Commit new file”

1.1.6.1.3 Add project info and links to README.md

- Go to the repository main page
- Click on README.md
- Click on pencil icon
- Add project description and status
- Add links to both local doc and read the docs. Change ‘project-name’ in the text below.

```
## Read the document here:
  * Click [here](http://project-name.readthedocs.io/) for the latest formatted release of the document at readthedocs.io.
  * Click [here](docs/README.md) for the local version of the document in Markdown format.
```

- Edit commit message
  - “Add project information to README.md”
- Click “Commit changes”

1.1.6.1.4 Create Document list for GitHub in docs/README.md

- Create a docs index in Markdown format for GitHub.
- Go to the repository main page
- Click on “New file”.
- Type “docs/README.md” in the file name text box.
- Click in the file content area and add the content below.
- Be sure to change ‘project-name’ in content below to actual project name.

```
# project-name v0.0.0

Documents
--------
* [User Guide](user-guide.md)
```

- Change commit message.
  - “Create docs/README.md with document list”
- Click “Commit new file”

1.1.6.1.5 Create a Table of Contents for ReadTheDocs in docs/index.rst

- Create a docs index in Markdown format for GitHub.
- Go to the repository main page
- Click on “New file”.
- Type “docs/index.rst” in the file name text box.
- Click in the file content area and add the content below.
- Be sure to change ‘project-name’ in content below to actual project name.

```
project-name
------------

Documents
=========
.. toctree::
   :maxdepth: 16

   user-guide
```

- Change commit message.
  - “Create docs/index.rst with table of contents”
- Click “Commit new file”

1.1.6.1.6 Create docs/conf.py

- Go to the repository main page
- Click on “New file”.
- Type “docs/conf.py” in the file name text box.
- Click in the file content area and add the content below.
- Be sure to change ‘project-name’ in content below to actual project name.

```
extensions = ['myst_parser']

source_suffix = ['.rst', '.md']

master_doc = 'index'
project = u'project-name'
```

- Edit commit message
  - “Add sphinx conf file for ReadTheDocs support”
- Click “Commit new file”

1.1.6.1.7 Create docs/requirements.txt

- Go to the repository main page
- Click on “New file”.
- Type “docs/requirements.txt” in the file name text box.
- Click in the file content area and add the content below.

```
myst_parser
```

- Edit commit message
  - “Add sphinx requirements file for ReadTheDocs support”
- Click “Commit new file”

1.1.6.2 Setup ReadTheDocs project

1.1.6.2.1 Import Project

- Go to My Projects by clicking on Account name in upper right
- Click Import a Project
- Click ‘+’ next to project name
- Click Next

1.1.7 Publish v0.0.0 Project Release

1.1.7.1 Add CHANGELOG.md with v0.0.0 description

- Go to the repository main page.
- Click on “New file”.
- Type “CHANGELOG.md” in the file name text box.
- Click in the file content area.
- Add content:

```
# v0.0.0
* Project description and status.
* Initial content
* ReadTheDocs support.
```

- Change commit message.
  - “Create CHANGELOG.md for v0.0.0 release”
- Click “Commit new file”

1.1.7.2 Create release v0.0.0

- Go to the repository main page.
- Click on “0 releases”
- Click “Create a new release”
- Type “v0.0.0” in the text box
- Enter “v0.0.0 Document Started” in the title box
- Enter the contents of the CHANGELOG.md entry in the description box
- Click “This is a pre-release”
- Click “Publish release”

1.1.7.3 Create branch “latest-release”

- Go to the repository main page
- Click on “Branch: master”.
- In the text box, type “latest-release”.
- Click “Create Branch: latest-release” “from master”.

1.1.7.4 Activate Version on ReadTheDocs

- Go to ReadTheDocs project
- Click Versions tab
- Click Edit next to v0.0.0
- Click Active
- Click Save

1.1.7.5 Setup ReadTheDocs default version

- Go to ReadTheDocs project
- Click Admin tab
- Click Advanced settings
- Change Default version to ‘stable’
- Click submit

1.1.7.6 Test versions on ReadTheDocs

- Go to ReadTheDocs project
- Click Builds tab
- Wait until builds are finished
- Click Versions tab
- Click on version number to view

1.1.8 Develop the Next Version

1.1.8.1 Update version on master to indicate version in development

- Update version at top of docs/*.md to indicate next anticipated version + “-prerelease”.
- Update version at top of README.md to indicate next anticipated version + “-prerelease”.

1.1.8.2 Improve the content

- Improve the documents in docs/ on the ‘master’ branch.

1.1.9 Release the Next Version

1.1.9.1 Set the version number

Set version number (vX.Y.Z) decided upon for the new release in file headers.

- Update top of README.md
- Update docs/*.md

1.1.9.2 Edit CHANGELOG.md to include vX.Y.Z description

- Go to the repository main page.
- Click on CHANGELOG.md.
- Click on pencil icon
- Click in the file content area.
- Add content:

```
# vX.Y.Z
* First improvement here.
* Next improvement here.
```

- Change commit message.
  - "Add vX.Y.Z release to CHANGELOG.md"
  - Click "Commit changes"

1.1.9.3 Create GitHub release

- Go to the repository main page.
- Click on "Releases" on the right hand side
- Click "Draft a new release"
- Click "Choose a tag"
- Type "vX.Y.Z" in the text box (replacing X, Y, and Z with the major, minor, and patch number.)
- Click "Create new tag: …" right below the text you just typed.
- Enter "vX.Y.Z Put Theme or CodeWord Here" in the title box (replacing with appropriate theme or code word)
- Enter the contents of the CHANGELOG.md entry in the description box
- If major version is zero, click "This is a pre-release"
- Click "Publish release"

1.1.9.4 Merge release tag to ‘latest-release’ branch.

Change user-name, project-name, and release-tag in commands below. The format vX.Y.Z should be used for release-tag.

```
$ git clone -b latest-release ssh://github.com/user-name/project-name
$ cd project-name
$ git merge --ff-only release-tag
$ git push
```