LLNL RtD Template

LLNL Open Source

Jul 27, 2023

CONTENTS:

This documentation site and the files associated with the .github repo are designed to be used as a template for developers who are new to Read the Docs (RtD) documentation.

CHAPTER

ONE

STEPS FOR SETTING UP DOCS

The process of creating a Read the Docs (RtD) site is widely documented (see also *Documentation Resources*), though your local environment will introduce variables. The examples that follow use `pipenv` as one variation.

The basic steps are:

1. Create repo on GitHub or create local directory to existing repo
2. Install Sphinx
3. Build and configure doc files
4. Set up GitHub and RtD webhook integration
5. Create .rst files

1.1 1. Create Repo/Directory

The instructions below assume that developers understand step 1.

1.2 2. Install Sphinx

- Create a **/docs** folder in your project. Then `cd docs` to complete most of the remaining steps (exceptions are noted below).
- Install RtD template from GitHub using Sphinx (`pipenv install sphinx-rtd-theme`).
- Initiate Sphinx quickstart (`pipenv run sphinx-quickstart`). Answer the questions that follow installation, such as selecting No for setting up build and source separately.
- Move the makefile into the `/docs` folder if it's not there already.
- Confirm the Sphinx version (`pipenv run sphinx-build --version`).
- Confirm that docs can build locally (`pipenv run make html`). You must be in the `/docs` directory for this to work. Unlike viewing a site on a local port, where newly saved changes auto-regenerate the site, you will need to run this command after every saved change in order to see the new change.

1.3 3. Build and Configure Doc Files

- Update the **conf.py** file. For instance, you can change the RtD theme (`html_theme`). In the General Configuration section, add the name of master .rst file:

```
# Tells Sphinx the name of the master .rst file.
master_doc = 'index'
```

- Create the **.gitignore** file (`touch .gitignore`) so your commits will include only source files, not the HTML-rendered files (GitHub instructions).
- Create the **requirements.txt** file to specify dependency versions (RtD instructions). This file usually contains information like this:

```
# These dependencies should be installed using pip in order
# to build the documentation.

sphinx==2.0.1
sphinxcontrib-programoutput==0.14
sphinx-rtd-theme==0.4.3
python-levenshtein
```

- Create the **.readthedocs.yml** file in the *top directory* of the repo. Paste in the V2 content.
- Commit configuration changes, then open and merge PR before moving on.

1.4 4. Establish Webhook Integration

This process generates a **readthedocs.io** URL for your project, which you can then link to in your repo's README file.

- Log into the RtD website via your GitHub account. You will have to go through an account verification process the first time.
- My Projects > Import a Project > select your repo. You may need to refresh the list or filter out an organization.
- Enter repo name and GitHub URL if not pre-populated. Confirm **.git** is appended to the end of the URL.
- Follow webhook creation instructions. When you try to build the docs for the first time, you may get an error: *fatal: could not read Username for 'https://github.com': terminal prompts disabled*. So you must configure the webhook to recognize GitHub's 2FA via `git config --global --add url."git@github.com:".insteadOf "https://github.com/"`.
- Add doc build status to your repo's README file. Open the file and paste in the badge syntax (where **name** is your repo):

```
[![Documentation Status](https://readthedocs.org/projects/name/badge/?
↪version=latest)](https://name.readthedocs.io/en/latest/?badge=latest)
```

- Confirm that docs still build locally, then open and merge PR.
- On your RtD dashboard, select the project and initiate a build to test: Projects > your repo > Builds > Build Version.
- You may want to change some of RtD's advanced settings, such as specifying which version is considered "latest" or which branch is the default. Don't forget to update this area if you change the name of your default branch (e.g., `master` to `main`). Projects > your repo > Admin > Advanced Settings.

1.5 5. Create and Populate .rst Files

All documentation files should contain your repo's copyright date(s) and license information at the top, set off by two periods (`..`).

- Create the **index.rst** file (or whatever file name you specified as the master file in step *C1* above).
- Set up table of contents (TOC), specifying how many levels deep you want to expose. For example, this site's TOC looks like this:

  ```
  .. toctree::
     :maxdepth: 2
     :caption: Contents:

     setup_steps
     resources
  ```

- Create subordinate .rst files with reference to the link you established in the TOC. The page doesn't have to have the same title as the file name. For example, this page is titled *Steps for Setting up Docs* while the TOC linkage is based on the file name of **setup_steps.rst**:

  ```
  .. _setup_steps:
  ```

- Now you can begin a cycle of adding/editing files, building locally, and pushing to GitHub. The configuration settings above should trigger automatic RtD builds with every commit or PR, but you can always manually build the docs site from your RtD dashboard.

1.5.1 Images

Adding inline images to your documentation is as simple as saving, then referencing, the image file at the proper level of the repo directory. This repo's images reside in the `/docs` folder.

Favicon (displays in the browser tab); in the `confy.py` file, add the image file name:

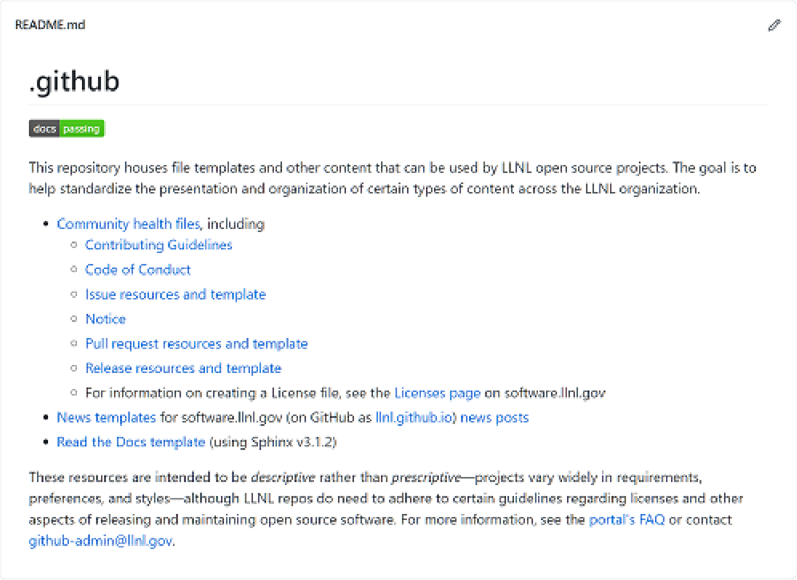
```
html_favicon = 'OS-icon-color.png'
```

Home image (displays in the top left corner), also in the `confy.py` file:

```
html_logo = 'OS-logo-horizontal-white.png'
```

Inline image, for which you can specify dimensions and alignment:

```
.. image:: OS-inline-example.png
    :width: 400px
    :align: center
```



README.md

.github

This repository houses file templates and other content that can be used by LLNL open source projects. The goal is to help standardize the presentation and organization of certain types of content across the LLNL organization.

- Community health files, including
  - Contributing Guidelines
  - Code of Conduct
  - Issue resources and template
  - Notice
  - Pull request resources and template
  - Release resources and template
  - For information on creating a License file, see the Licenses page on software.llnl.gov
- News templates for software.llnl.gov (on GitHub as llnl.github.io) news posts
- Read the Docs template (using Sphinx v3.1.2)

These resources are intended to be *descriptive* rather than *prescriptive*—projects vary widely in requirements, preferences, and styles—although LLNL repos do need to adhere to certain guidelines regarding licenses and other aspects of releasing and maintaining open source software. For more information, see the portal's FAQ or contact github-admin@llnl.gov.

CHAPTER

TWO

CONTRIBUTE TO EXISTING DOCS

To contribute to a repo that already has Sphinx/RtD set up, you will need to initiate a temporary virtual environment where you can build docs locally before pushing your changes.

2.1 Install requirements.txt

After forking/cloning the repo normally, locate the `requirements.txt` file in the project directory and navigate into that folder.

Create a virtual environment for the repo's docs:

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

Note:

- An error means a requirement is missing from your system or you are not in the correct directory.
- A repo's *gemfile* says what you need to install; the *lockfile* conveys the particular versions needed.

Activate the virtual environment:

```
$ pipenv shell
```

2.2 Build Docs Locally

Build the docs as static HTML files for editing:

```
$ pipenv run make html
```

The resulting information should indicate that the docs are ready in `_build/html`.

Note:

- Run this in the directory where the `Makefile` lives.
- Run this again after installing any missing requirements.
- Run this when you want to view the docs (i.e., as a QA check before committing and opening a PR).
- Run this whenever you need to rebuild docs (e.g., after un-staging a staged file).

- Prepend `pipenv run` when building docs via a virtual environment (otherwise, it would just be `make html`).

You can open a file in your browser to QA your changes:

```
$ open _build/html/Filename.html
```

Commit your edits and submit a PR normally.

2.3 When Finished Editing

You may want to update your `.gitignore` file with any changes to system files that you do not want committed.

Exit the virtual environment before switching to another repo by pressing `ctrl-d`. *If you don't do this, repo B will look at the requirements in repo A instead of finding its own.*

CHAPTER

THREE

DOCUMENTATION RESOURCES

These sites are good places to learn about Read the Docs (RtD), Sphinx, and all the things you can do in .rst files:

- RtD dashboard (you must be logged in)
- RtD documentation (in other words, RtD's RtD site)
- Sphinx (Python Document Generator)
- Sphinx Getting Started
- Sphinx tutorial
- reStructuredText Primer
- Graphviz syntax

To write more effective software documentation generally, here are some useful sites:

- The 18F Content Guide houses best practices for software documentation focused on user experiences and needs, particularly as related to government websites.
- Write the Docs, a community focused on communication and documentation, has a catalog of learning resources (including videos and tutorials) as well as a robust documentation guide. This community also has a GitHub repo.
- Docs as Code provides resources for development teams to build documentation repeatably and consistently across multiple platforms.
- The Documentation System emphasizes four types of documentation: tutorials, how-to guides, technical reference, and explanation documents. The site provides examples of and tips for each.
- The Good Docs Project, to which you can contribute, includes templates and writing instructions for documenting open source software. The organization's templates go through a rigorous writing, reviewing, and refining process.
- The Beautiful Docs repo has a list of language- and program-specific documentation resources.

Note: This site is running Sphinx v4.5.0 but downgraded to Docutils v0.16 for `sphinx_rtd_theme`. Docutils v0.17 has a bug in which bulleted (unordered) lists are not rendered properly.