Read the Docs Documentation

Release 2.7

Eric Holscher, Charlie Leifer, Bobby Grace

Dec 12, 2018

First Steps

Read the Docs simplifies software documentation by automating building, versioning, and hosting of your docs for you. Think of it as *Continuous Documentation*.

Never out of sync Whenever you push code to your favorite version control system, whether that is Git, Mercurial, Bazaar, or Subversion, Read the Docs will automatically build your docs so your code and documentation are always up-to-date.

Multiple versions Read the Docs can host and build multiple versions of your docs so having a 1.0 version of your docs and a 2.0 version of your docs is as easy as having a separate branch or tag in your version control system.

Free and open source Read the Docs is free and open source and hosts documentation for nearly 100,000 large and small open source projects in almost every human and computer language.

CHAPTER 1

First steps

Are you new to software documentation or are you looking to use your existing docs with Read the Docs? Learn about documentation authoring tools such as Sphinx and MkDocs to help you create fantastic documentation for your project.

- **Getting started**: *With Sphinx* | *With MkDocs*
- **Importing your existing documentation**: *Import guide*

1.1 Getting Started with Sphinx

Sphinx is a powerful documentation generator that has many great features for writing technical documentation including:

- Generate web pages, printable PDFs, documents for e-readers (ePub), and more all from the same sources
- You can use reStructuredText or *Markdown* to write documentation
- An extensive system of cross-referencing code and documentation
- Syntax highlighted code samples
- A vibrant ecosystem of first and third-party extensions

1.1.1 Quick start video

This screencast will help you get started or you can *read our guide below*.

1.1.2 Quick start

Assuming you have Python already, install Sphinx:

```
$ pip install sphinx
```

Create a directory inside your project to hold your docs:

```
$ cd /path/to/project
$ mkdir docs
```

Run `sphinx-quickstart` in there:

```
$ cd docs
$ sphinx-quickstart
```

This quick start will walk you through creating the basic configuration; in most cases, you can just accept the defaults. When it's done, you'll have an `index.rst`, a `conf.py` and some other files. Add these to revision control.

Now, edit your `index.rst` and add some information about your project. Include as much detail as you like (refer to the reStructuredText syntax or this template if you need help). Build them to see how they look:

```
$ make html
```

Your `index.rst` has been built into `index.html` in your documentation output directory (typically `_build/html/index.html`). Open this file in your web browser to see your docs.

Edit your files and rebuild until you like what you see, then commit your changes and push to your public repository. Once you have Sphinx documentation in a public repository, you can start using Read the Docs by *importing your docs*.

Fig. 1: Your Sphinx project is built

1.1.3 Using Markdown with Sphinx

You can use Markdown and reStructuredText in the same Sphinx project. We support this natively on Read the Docs, and you can do it locally:

```
$ pip install recommonmark
```

Then in your `conf.py`:

```
from recommonmark.
↪parser import CommonMarkParser

source_parsers = {
    '.md': CommonMarkParser,
}

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

Warning: Markdown doesn't support a lot of the features of Sphinx, like inline markup and directives. However, it works for basic prose content. reStructuredText is the preferred format for technical documentation, please read this blog post for motivation.

1.1.4 External resources

Here are some external resources to help you learn more about Sphinx.

- Sphinx documentation
- RestructuredText primer
- An introduction to Sphinx and Read the Docs for technical writers

1.2 Getting Started with MkDocs

MkDocs is a documentation generator that focuses on speed and simplicity. It has many great features including:

- Preview your documentation as you write it
- Easy customization with themes and extensions
- Writing documentation with Markdown

Note: MkDocs is a great choice for building technical documentation. However, Read the Docs also supports *Sphinx*, another tool for writing and building documentation.

1.2.1 Quick start

Assuming you have Python already, install MkDocs:

```
$ pip install mkdocs
```

Setup your MkDocs project:

```
$ mkdocs new .
```

This command creates `mkdocs.yml` which holds your MkDocs configuration, and `docs/index.md` which is the Markdown file that is the entry point for your documentation.

You can edit this `index.md` file to add more details about your project and then you can build your documentation:

```
$ mkdocs serve
```

This command builds your Markdown files into HTML and starts a development server to browse your documentation. Open up http://127.0.0.1:8000/ in your web browser to see your documentation. You can make changes to your Markdown files and your docs will automatically rebuild.

Once you have your documentation in a public repository such as GitHub, Bitbucket, or GitLab, you can start using Read the Docs by *importing your docs*.

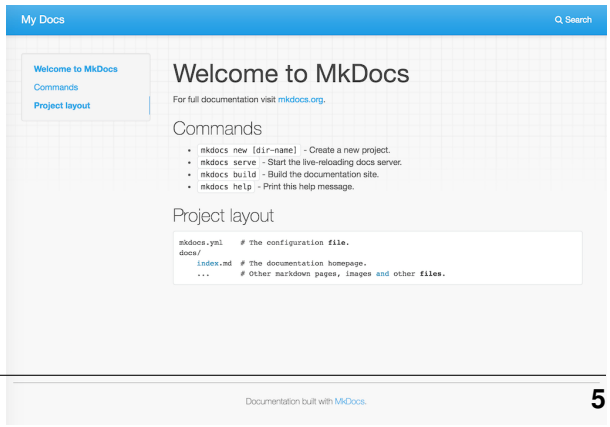


Fig. 2: Your MkDocs project is built

1.2.2 External resources

Here are some external resources to help you learn more about MkDocs.

- MkDocs documentation
- Markdown syntax guide
- Writing your docs with MkDocs

1.3 Importing Your Documentation

To import a documentation repository, visit your Read the Docs dashboard and click Import.

If you have *connected your Read the Docs account* to GitHub, Bitbucket, or GitLab, you will see a list of your repositories that we are able to import. To import one of these projects, just click the import icon next to the repository you'd like to import. This will bring up a form that is already filled with your project's information. Feel free to edit any of these properties, and then click **Next** to *build your documentation*.

1.3.1 Manually import your docs

If you do not have a connected account, you will need to select **Import Manually** and enter the information for your repository yourself. You will also need to manually configure the webhook for your repository as well. When importing your project, you will be asked for the repository URL, along with some other information for your new project. The URL is normally the URL or path name you'd use to checkout, clone, or branch your repository. Some examples:

- Git: `https://github.com/ericholscher/django-kong.git`
- Mercurial: `https://bitbucket.org/ianb/pip`
- Subversion: `http://varnish-cache.org/svn/trunk`
- Bazaar: `lp:pasta`

Add an optional homepage URL and some tags, and then click **Next**.

Once your project is created, you'll need to manually configure the repository webhook if you would like to have new changes trigger builds for your project on Read the Docs. Go to your project's **Integrations** page to configure a new webhook, or see *our steps for webhook creation* for more information on this process.

1.3.2 Building your documentation

Within a few seconds of completing the import process, your code will automatically be fetched from your public

repository, and the documentation will be built. Check out our *Build Process* page to learn more about how Read the Docs builds your docs, and to troubleshoot any issues that arise.

Some documentation projects require additional configuration to build such as specifying a certain version of Python or installing additional dependencies. You can configure these settings in a `readthedocs.yml` file. See our *Read the Docs YAML Config* docs for more details.

Read the Docs will host multiple versions of your code. You can read more about how to use this well on our *Versions* page.

If you have any more trouble, don't hesitate to reach out to us. The *Support* page has more information on getting in touch.

1.4 Versions

Read the Docs supports multiple versions of your repository. On the initial import, we will create a `latest` version. This will point at the default branch for your VCS control: `master`, `default`, or `trunk`.

We also create a `stable` version, if your project has any tagged releases. `stable` will be automatically kept up to date to point at your highest version.

1.4.1 How we envision versions working

In the normal case, the `latest` version will always point to the most up to date development code. If you develop on a branch that is different than the default for your VCS, you should set the **Default Branch** to that branch.

You should push a **tag** for each version of your project. These tags should be numbered in a way that is consistent with semantic versioning. This will map to your `stable` branch by default.

Note: We in fact are parsing your tag names against the rules given by PEP 440. This spec allows "normal" version numbers like `1.4.2` as well as pre-releases. An alpha version or a release candidate are examples of pre-releases and they look like this: `2.0a1`.

We only consider non pre-releases for the `stable` version of your documentation.

If you have documentation changes on a **long-lived branch**, you can build those too. This will allow you to see how the new docs will be built in this branch of the code. Generally you won't have more than 1 active branch over

a long period of time. The main exception here would be **release branches**, which are branches that are maintained over time for a specific release number.

1.4.2 Tags and branches

Read the Docs supports two workflows for versioning: based on tags or branches. If you have at least one tag, tags will take preference over branches when selecting the stable version.

1.4.3 Redirects on root URLs

When a user hits the root URL for your documentation, for example `http://pip.readthedocs.io/`, they will be redirected to the **Default version**. This defaults to **latest**, but could also point to your latest released version.

1.4.4 Version warning

This is a banner that appears on the top of every page of your docs that aren't stable or latest. This banner has a text with a link redirecting the users to the latest version of your docs.

This feature is disabled by default on new projects, you can enable it in the admin section of your docs (Advanced Settings).

1.5 Build Process

Files: tasks.py - doc_builder/

Every documentation build has limited resources. Our current build limits are:

- 15 minutes
- 1GB of memory

We can increase build limits on a per-project basis, if you provide a good reason your documentation needs more resources.

You can see the current Docker build images that we use in our docker repository. Docker Hub also shows the latest set of images that have been built.

Currently in production we're using the `readthedocs/build:2.0` docker image as our default image.

1.5.1 How we build documentation

When we import your documentation, we look at two things first: your *Repository URL* and the *Documentation Type*. We will clone your repository, and then build your documentation using the *Documentation Type* specified.

Sphinx

When you choose *Sphinx* as your *Documentation Type*, we will first look for a `conf.py` file in your repository. If we don't find one, we will generate one for you. We will look inside a `doc` or `docs` directory first, and then look within your entire project.

Then Sphinx will build any files with an `.rst` extension.

MkDocs

When you choose *Mkdocs* as your *Documentation Type*, we will first look for a `mkdocs.yml` file in the root of your repository. If we don't find one, we will generate one for you.

Then MkDocs will build any files with a `.md` extension within the directory specified as `docs_dir` in the `mkdocs.yml`.

If no `mkdocs.yml` was found in the root of your repository and we generated one for you, Read the Docs will attempt to set `docs_dir` by sequentially searching for a `docs`, `doc`, `Doc`, or `book` directory. The first of these directories that exists and contains files with a `.md` extension will be set to `docs_dir` within `mkdocs.yml`, and MkDocs will build the `.md` files in that directory. As MkDocs doesn't support automatic PDF generation, Read the Docs cannot create a PDF version of your documentation with the *Mkdocs* option.

Warning: We strongly recommend to *pin the MkDocs version* used for your project to build the docs to avoid potential future incompatibilities.

1.5.2 Understanding what's going on

Understanding how Read the Docs builds your project will help you with debugging the problems you have with the site. It should also allow you to take advantage of certain things that happen during the build process.

The first step of the process is that we check out your code from the repository you have given us. If the code is already checked out, we update the copy to the branch that you have specified in your projects configuration.

Then we build the proper backend code for the type of documentation you've selected.

If you have the *Install Project* option enabled, we will run `setup.py install` on your package, installing it into a virtual environment. You can also define additional packages to install with the *Requirements File* option.

When we build your Sphinx documentation, we run `sphinx-build -b html . _build/html`, where `html` would be replaced with the correct backend. We also create pdf's and ePub's automatically based on your project. For MkDocs, we run `mkdocs build`.

Then these files are copied across to our application servers from the build server. Once on the application servers, they are served from nginx.

An example in code:

```
update_docs_from_vcs(version)
if exists('setup.py'):
    run('python setup.py install')
if project.requirements_file:
    run('pip install␣
↪-r %s' % project.requirements_file)
build_docs(version=version)
copy_files(artifact_dir)
```

Note: Regardless of whether you build your docs with Sphinx or MkDocs, we recommend you pin the version of Sphinx or Mkdocs you want us to use. You can do this the same way other *dependencies are specified*. Some examples of pinning versions might be `sphinx<2.0` or `mkdocs>=1.0`.

1.5.3 Builder responsibility

Builders have a very specific job. They take the updated source code and generate the correct artifacts. The code lives in `self.version.project.checkout_path(self.version.slug)`. The artifacts should end up in `self.version.project.artifact_path(version=self.version.slug, type=self.type)` Where `type` is the name of your builder. All files that end up in the artifact directory should be in their final form.

1.5.4 The build environment

The build process is executed inside Docker containers, by default the image used is `readthedocs/build:2.0`,

but you can change that using a *configuration file*.

Note: The Docker images have a select number of C libraries installed, because they are used across a wide array of python projects. We can't install every C library out there, but we try and support the major ones.

Tip: If you want to know the specific version of a package that is installed in the image you can check the Ubuntu package search page.

2.0 (stable)

O.S Ubuntu 16.04

Conda Miniconda 4.3.31

Python

- `m2crypto`
- `matplolib`
- `numpy`
- `pandas`
- `pip`
- `scipy`

Other packages

- `doxygen`
- `graphviz`
- `libevent`
- `libjpeg`
- `libxml2-dev`
- `libxslt1.1`
- `pandoc`
- `textlive-full`

More details

3.0 (latest)

O.S Ubuntu 16.04

Conda Miniconda 4.4.10

Python

- `matplolib`
- `numpy`,
- `pandas`

- `pip`
- `scipy`

JavaScript

- `jsdoc`
- `nodejs`
- `npm`

Other packages

- `doxygen`
- `libevent-dev`
- `libgraphviz-dev`
- `libjpeg`
- `libxml2-dev`
- `libxslt1-dev`
- `pandoc`
- `plantuml`
- `textlive-full`

More details

1.5.5 Writing your own builder

Note: Builds happen on a server using only the RTD Public API. There is no reason that you couldn't build your own independent builder that wrote into the RTD namespace. The only thing that is currently unsupported there is a saner way than uploading the processed files as a zip.

The documentation build system in RTD is made pluggable, so that you can build out your own backend. If you have a documentation format that isn't currently supported, you can add support by contributing a backend.

The builder backends detail the higher level parts of the API that you need to implement. A basic run goes something like this:

```
backend␣
↪= get_backend(project.documentation_type)
if force:
    backend.force(version)
backend.clean(version)
backend.build(version)
if success:
    backend.move(version)
```

1.5.6 Deleting a stale or broken build environment

If you're having trouble getting your version to build, try wiping out the existing build/environment files. On your version list page `/projects/[project]/versions` there is a "Wipe" button that will remove all of the files associated with your documentation build, but not the documentation itself.

1.5.7 Build environment

The *Sphinx* and *Mkdocs* builders set the following RTD-specific environment variables when building your documentation:

Environment variable	Description	Example value
`READTHEDOCS`	Whether the build is running inside RTD	`True`
`READTHEDOCS_VERSION`	The RTD name of the version which is being built	`latest`
`READTHEDOCS_PROJECT`	The RTD name of the project which is being built	`myexampleproject`

1.6 Read the Docs features

This will serve as a list of all of the features that Read the Docs currently has. Some features are important enough to have their own page in the docs, others will simply be listed here.

1.6.1 GitHub, Bitbucket and GitLab Integration

We now support linking by default in the sidebar. It links to the page on your host, which should help people quickly update typos and send pull requests to contribute to project documentation.

More information can be found in the *Version Control System Integration* page.

1.6.2 Auto-updating

The *Webhooks* page talks about the different ways you can ping RTD to let us know your project has been updated. We have official support for GitHub, Bitbucket and GitLab, and anywhere else we have a generic post-commit hook that allows you to POST to a URL to get your documentation built.

1.6.3 Internationalization

Read the Docs itself is localized, and we support documentation translated into multiple languages. Read more on the *Localization of Documentation* and *Internationalization* pages.

1.6.4 Canonical URLs

Canonical URLs give your docs better search performance, by pointing all URLs to one version. This also helps to solve the issues around users landing on outdated versions of documentation.

More information can be found in the *Canonical URLs* page.

1.6.5 Versions

We can build multiple versions of your documentation. Look on the "Versions" page of your project's admin (using the nav on the left) to find a list of available versions that we've inferred from the tags and branches in your source control system (according to the support matrix below). On the Versions page you can tell us which versions you'd like us to build docs for, whether each should be public, protected, or private, and what the default version should be (we'll redirect there when someone hits your main project page, e.g., http://my-project.rtfd.org/).

1.6.6 Version Control Support Matrix

	Git	hg	bzr	svn
Tags	Yes	Yes	Yes	No
Branches	Yes	Yes	Yes	No
Default	master	default		trunk

1.6.7 PDF Generation

When you build your project on RTD, we automatically build a PDF of your project's documentation. We also build them for every version that you upload, so we can host the PDFs of your latest documentation, as well as your latest stable releases as well.

1.6.8 Search

We provide full-text search across all of the pages of documentation hosted on our site. This uses the excellent

Haystack project and Elasticsearch as the search backend. We hope to be integrating this into the site more fully in the future.

1.6.9 Alternate Domains

We provide support for custom domains, subdomains, and a shorturl for your project as well. This is outlined in the alternate_domains section.

1.7 Connecting Your Account

If you are going to import repositories from GitHub, Bitbucket, or GitLab, you should connect your Read the Docs account to your repository host first. Connecting your account allows for:

- Easier importing of your repositories
- Automatically configure your repository *Webhooks* which allow Read the Docs to build your docs on every change to your repository
- Log into Read the Docs with your GitHub, Bitbucket, or GitLab credentials

  If you signed up or logged in to Read the Docs with your GitHub, Bitbucket, or GitLab credentials, you're all done. Your account is connected.

  To connect your unconnected account, go to your *Settings* dashboard and select Connected Services. From here, you'll be able to connect to your GitHub, Bitbucket or GitLab account. This process will ask you to authorize a connection to Read the Docs, that allows us to read information about and clone your repositories.

1.8 Support

1.8.1 Usage Questions

If you have questions about how to use Read the Docs, or have an issue that isn't related to a bug, Stack Overflow is the best place to ask. Tag questions with `read-the-docs` so other folks can find them easily.

Good questions for Stack Overflow would be:

- "What is the best way to structure the table of contents across a project?"
- "How do I structure translations inside of my project for easiest contribution from users?"

- "How do I use Sphinx to use SVG images in HTML output but PNG in PDF output?"

1.8.2 Community Support

Read the Docs is supported by community contributions and advertising. We hope to bring in enough money with our Gold and Ethical Ads programs to keep Read the Docs sustainable.

All people answering your questions are doing it with their own time, so please be kind and provide as much information as possible.

Bugs & Support Issues

You can file bug reports on our GitHub issue tracker, and they will be addressed as soon as possible. **Support is a volunteer effort**, and there is no guaranteed response time. If you need answers quickly, you can buy commercial support below.

Reporting Issues

When reporting a bug, please include as much information as possible that will help us solve this issue. This includes:

- Project name
- URL
- Action taken
- Expected result
- Actual result

1.8.3 Commercial Support

We offer commercial support for Read the Docs, commercial hosting, as well as consulting around all documentation systems. You can contact us at hello@readthedocs.com to learn more, or read more at https://readthedocs.com/services/#open-source-support.

1.9 Frequently Asked Questions

1.9.1 My project isn't building with autodoc

First, you should check out the Builds tab of your project. That records all of the build attempts that RTD has made

to build your project. If you see `ImportError` messages for custom Python modules, you should enable the virtualenv feature in the Admin page of your project, which will install your project into a virtualenv, and allow you to specify a `requirements.txt` file for your project.

If you are still seeing errors because of C library dependencies, please see *I get import errors on libraries that depend on C modules*.

1.9.2 How do I change my project slug (the URL your docs are served at)?

We don't support allowing folks to change the slug for their project. You can update the name which is shown on the site, but not the actual URL that documentation is served.

The main reason for this is that all existing URLs to the content will break. You can delete and re-create the project with the proper name to get a new slug, but you really shouldn't do this if you have existing inbound links, as it breaks the internet.

1.9.3 How do I change the version slug of my project?

We don't support allowing folks to change the slug for their versions. But you can rename the branch/tag to achieve this. If that isn't enough, you can ask to team to do this by creating an issue.

1.9.4 Help, my build passed but my documentation page is 404 Not Found!

This often happens because you don't have an `index.html` file being generated. Make sure you have one of the following files:

- `index.rst`
- `index.md`

  At the top level of your built documentation, otherwise we aren't able to serve a "default" index page.

  To test if your docs actually built correctly, you can navigate to a specific page (`/en/latest/README.html` for example).

1.9.5 How do I change behavior for Read the Docs?

When RTD builds your project, it sets the `READTHEDOCS` environment variable to the string `True`. So within your Sphinx `conf.py` file, you can vary the behavior based on this. For example:

```
import os
on_rtd␣
↪= os.environ.get('READTHEDOCS') == 'True'
if on_rtd:
    html_theme = 'default'
else:
    html_theme = 'nature'
```

The `READTHEDOCS` variable is also available in the Sphinx build environment, and will be set to `True` when building on RTD:

```
{% if READTHEDOCS %}
Woo
{% endif %}
```

1.9.6 I get import errors on libraries that depend on C modules

Note: Another use case for this is when you have a module with a C extension.

This happens because our build system doesn't have the dependencies for building your project. This happens with things like `libevent`, `mysql`, and other python packages that depend on C libraries. We can't support installing random C binaries on our system, so there is another way to fix these imports.

With Sphinx you can use the built-in autodoc_mock_imports for mocking. Alternatively you can use the mock library by putting the following snippet in your `conf.py`:

```
import sys
from unittest.mock import MagicMock

class Mock(MagicMock):
    @classmethod
    def __getattr__(cls, name):
        return MagicMock()

MOCK_MODULES = ['pygtk', 'gtk',␣
↪'gobject', 'argparse', 'numpy', 'pandas']
sys.modules.update((mod_name,
↪ Mock()) for mod_name in MOCK_MODULES)
```

You need to replace `MOCK_MODULES` with the modules that you want to mock out.

Tip: The library `unittest.mock` was introduced on python 3.3. On earlier versions install the `mock` library from PyPI with (ie `pip install mock`) and replace the above import:

```
from mock import Mock as MagicMock
```

If such libraries are installed via `setup.py`, you also will need to remove all the C-dependent libraries from your `install_requires` in the RTD environment.

1.9.7 `Client Error 401` when building documentation

If you did not install the `test_data` fixture during the installation instructions, you will get the following error:

```
slumber.exceptions.
↪HttpClientError: Client Error 401:␣
↪http://localhost:8000/api/v1/version/
```

This is because the API admin user does not exist, and so cannot authenticate. You can fix this by loading the test_data:

```
./manage.py loaddata test_data
```

If you'd prefer not to install the test data, you'll need to provide a database account for the builder to use. You can provide these credentials by editing the following settings:

```
SLUMBER_USERNAME = 'test'
SLUMBER_PASSWORD = 'test'
```

1.9.8 Deleting a stale or broken build environment

See *Wiping a Build Environment*.

1.9.9 How do I host multiple projects on one custom domain?

We support the concept of subprojects, which allows multiple projects to share a single domain. If you add a subproject to a project, that documentation will be served under the parent project's subdomain or custom domain.

For example, Kombu is a subproject of Celery, so you can access it on the `celery.readthedocs.io` domain:

http://celery.readthedocs.io/projects/kombu/en/latest/

This also works the same for custom domains:

http://docs.celeryproject.org/projects/kombu/en/latest/

You can add subprojects in the project admin dashboard.

1.9.10 Where do I need to put my docs for RTD to find it?

Read the Docs will crawl your project looking for a `conf.py`. Where it finds the `conf.py`, it will run `sphinx-build` in that directory. So as long as you only have one set of sphinx documentation in your project, it should Just Work.

1.9.11 I want to use the Blue/Default Sphinx theme

We think that our theme is badass, and better than the default for many reasons. Some people don't like change though :), so there is a hack that will let you keep using the default theme. If you set the `html_style` variable in your `conf.py`, it should default to using the default theme. The value of this doesn't matter, and can be set to `/default.css` for default behavior.

1.9.12 I want to use the Read the Docs theme locally

There is a repository for that: https://github.com/rtfd/sphinx_rtd_theme. Simply follow the instructions in the README.

1.9.13 Image scaling doesn't work in my documentation

Image scaling in docutils depends on PIL. PIL is installed in the system that RTD runs on. However, if you are using the virtualenv building option, you will likely need to include PIL in your requirements for your project.

1.9.14 I want comments in my docs

RTD doesn't have explicit support for this. That said, a tool like Disqus (and the sphinxcontrib-disqus plugin) can be used for this purpose on RTD.

1.9.15 How do I support multiple languages of documentation?

See the section on *Localization of Documentation*.

1.9.16 Does Read The Docs work well with "legible" docstrings?

Yes. One criticism of Sphinx is that its annotated docstrings are too dense and difficult for humans to read. In response, many projects have adopted customized docstring styles that are simultaneously informative and legible. The NumPy and Google styles are two popular docstring formats. Fortunately, the default Read The Docs theme handles both formats just fine, provided your `conf.py` specifies an appropriate Sphinx extension that knows how to convert your customized docstrings. Two such extensions are numpydoc and napoleon. Only `napoleon` is able to handle both docstring formats. Its default output more closely matches the format of standard Sphinx annotations, and as a result, it tends to look a bit better with the default theme.

1.9.17 Can I document a python package that is not at the root of my repository?

Yes. The most convenient way to access a python package for example via Sphinx's autoapi in your documentation is to use the *Install your project inside a virtualenv using setup.py install* option in the admin panel of your project. However this assumes that your `setup.py` is in the root of your repository.

If you want to place your package in a different directory or have multiple python packages in the same project, then create a pip requirements file. You can specify the relative path to your package inside the file. For example you want to keep your python package in the `src/python` directory, then create a `requirements.readthedocs.txt` file with the following contents:

```
src/python/
```

Please note that the path must be relative to the file. So the example path above would work if the file is in the root of your repository. If you want to put the requirements in a file called `requirements/readthedocs.txt`, the contents would look like:

```
../python/
```

After adding the file to your repository, go to the *Advanced Settings* in your project's admin panel and add the name of the file to the *Requirements file* field.

1.9.18 What commit of Read the Docs is in production?

We deploy readthedocs.org from the `rel` branch in our GitHub repository. You can see the latest commits that have been deployed by looking on GitHub: https://github.com/rtfd/readthedocs.org/commits/rel

1.10 Read the Docs YAML Config

Read the Docs now has support for configuring builds with a YAML file. The file, `readthedocs.yml`, must be in the root directory of your project.

> **Warning:** This feature is in a beta state. Please file an issue if you find anything wrong.

Here is an example of what this file looks like:

```
# .readthedocs.yml

build:
  image: latest

python:
  version: 3.6
  setup_py_install: true
```

1.10.1 Supported settings

formats

- Default: [`htmlzip`, `pdf`, `epub`]
- Options: `htmlzip`, `pdf`, `epub`
- Type: List

The formats of your documentation you want to be built. Set as an empty list `[]` to build none of the formats.

Note: We will always build an HTML & JSON version of your documentation. These are used for web serving & search indexing, respectively.

```
# Don't build any extra formats
formats: []
```

```
# Build PDF & ePub
formats:
```

(continues on next page)

(continued from previous page)

```
    - epub
    - pdf
```

requirements_file

- Default: `null`
- Type: Path (specified from the root of the project)

  The path to your pip requirements file.

```
requirements_file: requirements/docs.txt
```

conda

The `conda` block allows for configuring our support for Conda.

conda.file

- Default: `null`
- Type: Path (specified from the root of the project)

  The file option specified the Conda environment file to use.

```
conda:
  file: environment.yml
```

Note: Conda is only supported via the YAML file.

build

The `build` block configures specific aspects of the documentation build.

build.image

- Default: `readthedocs/build:2.0`
- Options: `1.0`, `2.0`, `latest`

  The build image to use for specific builds. This lets users specify a more experimental build image, if they want to be on the cutting edge.

  Certain Python versions require a certain build image, as defined here:

- `1.0`: 2, 2.7, 3, 3.4

- `2.0`: 2, 2.7, 3, 3.5
- `latest`: 2, 2.7, 3, 3.3, 3.4, 3.5, 3.6

```
build:
    image: latest

python:
    version: 3.6
```

python

The `python` block allows you to configure aspects of the Python executable used for building documentation.

python.version

- Default: `2.7`
- Options: `2.7`, `2`, `3.5`, `3`

This is the version of Python to use when building your documentation. If you specify only the major version of Python, the highest supported minor version will be selected.

Warning: The supported Python versions depends on the version of the build image your project is using. The default build image that is used to build documentation contains support for Python `2.7` and `3.5`. See the *build.image* for more information on supported Python versions.

```
python:
   version: 3.5
```

python.use_system_site_packages

- Default: `false`
- Type: Boolean

When true, it gives the virtual environment access to the global site-packages directory. Depending on the *build.image*, Read the Docs includes some libraries like scipy, numpy, etc. See *The build environment* for more details.

```
python:
   use_system_site_packages: true
```

python.setup_py_install

- Default: `false`
- Type: Boolean

When true, install your project into the Virtualenv with `python setup.py install` when building documentation.

```
python:
   setup_py_install: true
```

python.pip_install

- Default: `false`
- Type: Boolean

When `true`, install your project into the virtualenv with pip when building documentation.

```
python:
   pip_install: true
```

python.extra_requirements

- Default: `[]`
- Type: List

List of extra requirements sections to install, additionally to the package default dependencies. Only works if `python.pip_install` option above is set to `true`.

Let's say your Python package has a `setup.py` which looks like this:

```
from setuptools import setup

setup(
    name="my_package",
    # (...)
    install_requires=[
        'requests',
        'simplejson'],
    extras_require={
        'tests': [
            'nose',
            'pycodestyle >= 2.1.0'],
        'docs': [
            'sphinx >= 1.4',
            'sphinx_rtd_theme']}
)
```

Then to have all dependencies from the `tests` and `docs` sections installed in addition to the default `requests` and `simplejson`, use the `extra_requirements` as such:

```
python:
    extra_requirements:
        - tests
        - docs
```

Behind the scene the following Pip command will be run:

```
$ pip install .[tests,docs]
```

1.11 Guides

These guides will help walk you through the usage of Read the Docs.

1.11.1 Adding Custom CSS or JavaScript to a Sphinx Project

The easiest way to add custom stylesheets or scripts, and ensure that the files are added in a way that allows for overriding of existing styles or scripts, is to add these files using a `conf.py` Sphinx extension. Inside your `conf.py`, if a function `setup(app)` exists, Sphinx will call this function as a normal extension during application setup.

For example, if a custom stylesheet exists at `_static/css/custom.css`, a `conf.py` extension can be written to add this file to the list of stylesheets:

```
def setup(app):
    app.add_stylesheet('css/custom.css')
```

Using an extension to add the stylesheet allows for the file to be added to the list of stylesheets later in the Sphinx setup process, making overriding parts of the Read the Docs theme possible.

The same method can be used to add additional scripts that might override previously initialized scripts:

```
def setup(app):
    app.add_javascript('js/custom.js')
```

1.11.2 Enabling Build Notifications

Using email

Read the Docs allows you to configure emails that can be sent on failing builds. This makes sure you know when your builds have failed.

Take these steps to enable build notifications using email:

- Go to **Admin > Notifications** in your project.
- Fill in the **Email** field under the **New Email Notifications** heading
- Submit the form

You should now get notified by email when your builds fail!

Using webhook

Read the Docs can also send webhooks when builds fail.

Take these steps to enable build notifications using a webhook:

- Go to **Admin > Notifications** in your project.
- Fill in the **URL** field under the **New Webhook Notifications** heading
- Submit the form

The project name, slug and its details for the build that failed will be sent as POST request to your webhook URL:

```
{
      "name": "Read the Docs",
      "slug": "rtd",
      "build": {
          "id": 6321373,
          "success": false,
          "date": "2017-02-15 20:35:54"
      }
}
```

You should now get notified on your webhook when your builds fail!

1.11.3 My Build is Using Too Many Resources

We limit build resources to make sure that users don't overwhelm our build systems. If you are running into this issue, there are a couple fixes that you might try.

Note: The current build limits can be found on our *Build Process* page.

Reduce formats you're building

You can change the formats of docs that you're building with our YAML file's *formats* option.

In particular, the `htmlzip` takes up a decent amount of memory and time, so disabling that format might solve your problem.

Reduce documentation build dependencies

A lot of projects reuse their requirements file for their documentation builds. If there are extra packages that you don't need for building docs, you can create a custom requirements file just for documentation. This should speed up your documentation builds, as well as reduce your memory footprint.

Use pip when possible

If you don't need `conda` to create your *documentation* environment, consider using `pip` instead since `conda` could require too much memory to calculate the dependency tree when using multiple channels.

Tip: Even though your *project* environment is created with `conda`, it may be not necessary for the *documentation* environment. That is, to build the documentation is probably that you need fewer Python packages than to use your library itself. So, in this case, you could use `pip` to install those fewer packages instead of creating a big environment with `conda`.

Use system site-packages for pre-installed libs

There are a few libraries that Read the Docs has already installed (scipy, numpy, matplotlib, pandas, etc) in the Docker image used to build your docs. You can check the updated list of pre-installed libraries in the Docker image repository.

To use these pre-installed libraries and avoid consuming time re-downloading/compiling them, you can use the *python.use_system_site_packages* option to have access to them.

1.11.4 Enabling Google Analytics on your Project

Read the Docs has native support for Google Analytics. You can enable it by:

- Going to **Admin > Advanced Settings** in your project.

- Fill in the **Analytics code** heading with your Google Tracking ID (example `UA-123456674-1`)

  Once your documentation rebuilds it will include your Analytics tracking code and start sending data. Google Analytics usually takes 60 minutes, and sometimes can take up to a day before it starts reporting data.

Note: Read the Docs takes some extra precautions with analytics to protect user privacy. As a result, users with Do Not Track enabled will not be counted for the purpose of analytics.

For more details, see the *Do Not Track section* of our privacy policy.

1.11.5 Manage Translations

This guide walks through the process needed to manage translations of your documentation. Once this work is done, you can setup your project under Read the Docs to build each language of your documentation by reading *Localization of Documentation*.

Overview

There are many different ways to manage documentation in multiple languages by using different tools or services. All of them have their pros and cons depending on the context of your project or organization.

In this guide we will focus our efforts around two different methods: manual and using Transifex.

In both methods, we need to follow these steps to translate our documentation:

1. Create translatable files (`.pot` and `.po` extensions) from source language
2. Translate the text on those files from source language to target language
3. Build the documentation in *target language* using the translated texts

Besides these steps, once we have published our first translated version of our documentation, we will want to keep it updated from the source language. At that time, the workflow would be:

1. Update our translatable files from source language
2. Translate only *new* and *modified* texts in source language into target language
3. Build the documentation using the most up to date translations

Create translatable files

To generate these `.pot` files it's needed to run this command from your `docs/` directory:

```
$ sphinx-build -b gettext . _build/gettext
```

Tip: We recommend configuring Sphinx to use gettext_uuid as `True` and also gettext_compact as `False` to generate `.pot` files.

This command will leave the generated files under `_build/gettext`.

Translate text from source language

Manually

We recommend using sphinx-intl tool for this workflow.

First, you need to install it:

```
$ pip install sphinx-intl
```

As a second step, we want to create a directory with each translated file per target language (in this example we are using Spanish/Argentina and Portuguese/Brazil). This can be achieved with the following command:

```
$ sphinx-intl update␣
↪-p _build/gettext -l es_AR -l pt_BR
```

This command will create a directory structure similar to the following (with one `.po` file per `.rst` file in your documentation):

```
locale
├── es_AR
│   └── LC_MESSAGES
│       └── index.po
└── pt_BR
    └── LC_MESSAGES
        └── index.po
```

Now, you can just open those `.po` files with a text editor and translate them taking care of no breaking the reST notation. Example:

```
# b8f891b8443f4a45994c9c0a6bec14c3
#: ../../index.rst:4
msgid ""
"Read the Docs hosts documentation␣
↪for the open source community."
"It supports :ref:`Sphinx <sphinx>
↪` docs written with reStructuredText."
```

(continues on next page)

(continued from previous page)

```
msgstr ""
"FILL HERE BY TARGET LANGUAGE␣
↪FILL HERE BY TARGET LANGUAGE FILL HERE "
"BY TARGET LANGUAGE␣
↪:ref:`Sphinx <sphinx>` FILL HERE."
```

Using Transifex

Transifex is a platform that simplifies the manipulation of `.po` files and offers many useful features to make the translation process as smooth as possible. These features includes a great web based UI, Translation Memory, collaborative translation, etc.

You need to create an account in their service and a new project before start.

After that, you need to install the transifex-client tool which will help you in the process to upload source files, update them and also download translated files. To do this, run this command:

```
$ pip install transifex-client
```

After installing it, you need to configure your account. For this, you need to create an API Token for your user to access this service through the command line. This can be done under your User's Settings.

Now, you need to setup it to use this token:

```
$ tx init --token $TOKEN --no-interactive
```

The next step is to map every `.pot` file you have created in the previous step to a resource under Transifex. To achieve this, you need to run this command:

```
$ tx config mapping-bulk \
    --project $TRANSIFEX_PROJECT \
    --file-extension '.pot' \
    --source-file-dir docs/_build/gettext \
    --source-lang en \
    --type PO \
    --expression 'locale/<lang>
↪/LC_MESSAGES/{filepath}/{filename}.po' \
    --execute
```

This command will generate a file at `.tx/config` with all the information needed by the `transifext-client` tool to keep your translation synchronized.

Finally, you need to upload these files to Transifex platform so translators can start their work. To do this, you can run this command:

```
$ tx push --source
```

Now, you can go to your Transifex's project and check that there is one resource per `.rst` file of your documentation. After the source files are translated using Transifex, you can download all the translations for all the languages by running:

```
$ tx pull --all
```

This command will leave the `.po` files needed for building the documentation in the target language under `locale/<lang>/LC_MESSAGES`.

Warning: It's important to use always the same method to translate the documentation and do not mix them. Otherwise, it's very easy to end up with inconsistent translations or losing already translated text.

Build the documentation in target language

Finally, to build our documentation in Spanish(Argentina) we need to tell Sphinx builder the target language with the following command:

```
$ sphinx-build -b html␣
↪-D language=es_AR . _build/html/es_AR
```

Note: There is no need to create a new `conf.py` to redefine the `language` for the Spanish version of this documentation.

After running this command, the Spanish(Argentina) version of your documentation will be under `_build/html/es_AR`.

Summary

Update sources to be translated

Once you have done changes in your documentation, you may want to make these additions/modifications available for translators so they can update it:

1. Create the `.pot` files:

   ```
   $ sphinx-build -b gettext . _build/gettext
   ```

1. Push new files to Transifex

   ```
   $ tx push --sources
   ```

Build documentation from up to date translation

When translators have finished their job, you may want to update the documentation by pulling the changes from Transifex:

1. Pull up to date translations from Transifex:

```
$ tx pull --all
```

2. Commit and push these changes to our repo

```
$ git add locale/
$ git commit -m "Update translations"
$ git push
```

The last `git push` will trigger a build per translation defined as part of your project under Read the Docs and make it immediately available.

1.11.6 Keep Building Docs With Old Version Of MkDocs

Recent changes to `mkdocs` forced us to upgrade the default version installed by Read the Docs and this may be a breaking change for your documentation.

You should check that your docs continue building in any of these cases:

- your project doesn't have a `requirements.txt` file pinning `mkdocs` to a specific version
- your project is using a custom theme
- your project is using Markdown extensions

In case your builds are failing because of a `mkdocs` issue, you may want to follow one of the following solutions depending on the case.

Pin mkdocs to the previous version

Before Read the Docs upgraded its default version installed, `mkdocs==0.15.0` was used. To make your project continue using this version you will need to create a `requirements.txt` file with this content:

```
# requirements.txt
mkdocs==0.15.0
mkdocs-bootstrap==0.1.1
mkdocs-bootswatch==0.1.0
markdown>=2.3.1,<2.5
```

Note: Most of the breaking changes where introduced in `mkdocs==0.17`, so you may want to test only pinning `mkdocs` to `mkdocs<0.17` and check if your docs keep building successfully.

More detailed information about how to specify dependencies can be found *here*.

Upgrade your custom theme to be compatible with newer mkdocs versions

It is possible that event your build passes your documentation doesn't look correct. This may be because the new `mkdocs==0.17.3` version installed by Read the Docs introduced some breaking changes on the structure of the theme.

You should check the mkdocs' Custom Theme documentation to upgrade your custom theme and make it compatible with the new version.

Upgrade how extension arguments are defined

`mkdocs` has changed the way that `markdown_extensions` are defined and you may need to upgrade it. If you where passing arguments to the extension by defining them between brackets (`toc(permalink=true)`) in your `mkdocs.yaml` you may need to upgrade it to the new way.

For example, this definition:

```
markdown_extensions:
  - admonition
  - codehilite(guess_lang=false)
  - toc(permalink=true)
  - footnotes
  - meta
```

needs to be replaced by:

```
markdown_extensions:
  - admonition
  - codehilite
      guess_lang: false
  - toc
      permalink: true
  - footnotes
  - meta
```

1.11.7 Removing "Edit on ..." Buttons from Documentation

When building your documentation, Read the Docs automatically adds buttons at the top of your documentation

and in the versions menu that point readers to your repository to make changes. For instance, if your repository is on GitHub, a button that says "Edit on GitHub" is added in the top-right corner to your documentation to make it easy for readers to author new changes.

Remove links from top-right corner

The only way to remove these links currently is to override the Read the Docs theme templates:

- In your Sphinx project path, create a directory called `_templates`. If you use a different `templates_path` option in your `conf.py`, substitute that directory name.
- Create a file in this path called `breadcrumbs.html`

The new `breadcrumbs.html` should look like this:

```
{%- extends␣
↪"sphinx_rtd_theme/breadcrumbs.html" %}

{% block breadcrumbs_aside %}
{% endblock %}
```

Remove "On . . . " section from versions menu

This section can be removed with a custom CSS rule to hide them. Follow the instructions under *Adding Custom CSS or JavaScript to a Sphinx Project* and put the following content into the `.css` file:

```
/* Hide␣
↪"On GitHub" section from versions menu */
div.rst-versions > div.rst-other-versions␣
↪> div.injected > dl:nth-child(4) {
    display: none;
}
```

Warning: You may need to change the `4` number in `dl:nth-child(4)` for a different one in case your project has more sections in the versions menu. For example, if your project has translations into different languages, you will need to use the number `5` there.

Now when you build your documentation, your documentation won't include an edit button or links to the page source.

1.11.8 Specifying Dependencies

Any dependencies required for building a documentation project can be specified using a pip requirements file or a conda environment file.

Note: For the purpose of building your documentation with RTD, *project* is the documentation project, and *project root* is the directory where all the documentation is stored, often named `docs`.

Specifying a requirements file

The requirements file option is useful for specifying dependencies required for building the documentation. Additional uses specific to Read the Docs are mentioned at the end of this guide.

For details about the purpose of pip requirements file and how to create one, check out pip user guide.

To use the requirements file, create and place the requirements file in the root directory of your documentation directory. For example:

```
docs/requirements.txt
```

Using the YAML configuration file

The recommended approach for specifying a pip requirements file is to use a `readthedocs.yml` file.

The file's path should be relative to documentation root.

```
requirements_file: requirements.txt
```

See *Read the Docs YAML Config* for setting up the .yml file

Using the project admin dashboard

Once the requirements file has been created;

- Login to Read the Docs and go to the project admin dashboard.
- Go to `Admin > Advanced Settings > Requirements file`.
- Specify the path of the requirements file you just created. The path should be relative to the root directory of the documentation project.

Using a conda environment file

If using conda, the dependencies can be specified in the conda environment file, `environment.yml` .

More on Read the Doc's conda support: *Conda Support*

Working with conda and environment.yml

Note: Conda is only supported via the YAML file.

This conda environment file can be specified in `readthedocs.yml` in the `conda` block.

```
conda:
    file: environment.yml
```

As before, the path should be relative to the documentation repository root.

1.11.9 Wiping a Build Environment

Sometimes it happen that your Builds start failing because the build environment where the documentation is created is stale or broken. This could happen for a couple of different reasons like `pip` not upgrading a package properly or a corrupted cached Python package.

In any of these cases (and many others), the solution could be just wiping out the existing build environment files and allow Read the Docs to create a new fresh one.

Follow these steps to wipe the build environment:

- Go to **Versions**
- Click on the **Edit** button of the version you want to wipe on the right side of the page
- Go to the bottom of the page and click the **wipe** link, next to the "Save" button

Note: By wiping the documentation build environment, all the `rst`, `md`, and code files associated with it will be removed but not the documentation already built (`HTML` and `PDF` files). Your documentation will still online after wiping the build environment.

Now you can re-build the version with a fresh build environment!

1.12 Public API

This section of the documentation details the public API usable to get details of projects, builds, versions and other

details from Read the Docs.

Warning: Originally, the Read the Docs API allowed connections over insecure HTTP. Starting in January 2019, requests over HTTP will be automatically redirected to HTTPS and non-GET/HEAD requests over insecure HTTP will fail.

Tip: It is a good idea to put your email address, application name, or Read the Docs username into the user agent header of your requests. That way the Read the Docs' team can contact you in the event of issues.

1.12.1 API v2

The Read the Docs API uses REST (Representational State Transfer). JSON is returned by all API responses including errors and HTTP response status codes are to designate success and failure.

Note: A newer API v3 is in early development stages. Some improvements coming in v3 are:

- Search API
- Write access
- Simpler URLs which use slugs instead of numeric IDs
- Improved error reporting

If there are features you would like in v3, please get in touch in the issue tracker.

Authentication and authorization

Requests to the Read the Docs public API are for public information only and do not require any authentication.

Resources

Projects

Projects are the main building block of Read the Docs. Projects are built when there are changes to the code and the resulting documentation is hosted and served by Read the Docs.

As an example, this documentation is part of the Docs project which has documentation at https://docs.readthedocs.io.

You can always view your Read the Docs projects in your project dashboard.

Project list

GET /api/v2/project/
Retrieve a list of all Read the Docs projects.

Example request:

```
$ curl https:/
↪/readthedocs.org/api/v2/project/?slug=pip
```

Example response:

```
{
    "count": 1,
    "next": null,
    "previous": null,
    "results": [PROJECTS]
}
```

Response JSON Object

- **next** (*string*) – URI for next set of Projects.
- **previous** (*string*) – URI for previous set of Projects.
- **count** (*integer*) – Total number of Projects.
- **results** (*array*) – Array of `Project` objects.

Query Parameters

- **slug** (*string*) – Narrow the results by matching the exact project slug

Project details

GET /api/v2/project/(int: *id*)/
Retrieve details of a single project.

```
{
    "id": 6,
    "name": "Pip",
    "slug": "pip",
    "programming_language": "py",
    "default_version": "stable",
    "default_branch": "master",
    "repo_type": "git",
    "repo": "https://github.com/pypa/pip",
↪ "description": "Pip Installs Packages.",
    "language": "en",
    "documentation_type": "sphinx_htmldir",
    "canonical_
↪url": "http://pip.pypa.io/en/stable/",
```

(continues on next page)

(continued from previous page)

```
    "users": [USERS]
}
```

Response JSON Object

- **id** (*integer*) – The ID of the project
- **name** (*string*) – The name of the project.
- **slug** (*string*) – The project slug (used in the URL).
- **programming_language** (*string*) – The programming language of the project (eg. "py", "js")
- **default_version** (*string*) – The default version of the project (eg. "latest", "stable", "v3")
- **default_branch** (*string*) – The default version control branch
- **repo_type** (*string*) – Version control repository of the project
- **repo** (*string*) – The repository URL for the project
- **description** (*string*) – An RST description of the project
- **language** (*string*) – The language code of this project
- **documentation_type** (*string*) – An RST description of the project
- **canonical_url** (*string*) – The canonical URL of the default docs
- **users** (*array*) – Array of `User` IDs who are maintainers of the project.

Status Codes

- 200 OK – no error
- 404 Not Found – There is no `Project` with this ID

Project versions

GET /api/v2/project/(int: *id*)/active_versions/

Retrieve a list of active versions (eg. "latest", "stable", "v1.x") for a single project.

```
{
    "versions": [VERSION, VERSION, ...]
}
```

Response JSON Object

- **versions** (*array*) – Version objects for the given `Project`

See the *Version detail* call for the format of the `Version` object.

Versions

Versions are different versions of the same project documentation

The versions for a given project can be viewed in a project's version screen. For example, here is the Pip project's version screen.

Version list

GET /api/v2/version/
Retrieve a list of all Versions for all projects

```
{
    "count": 1000,
    "previous": null,
    "results": [VERSIONS],
    "next": "https://readthedocs.
↪org/api/v2/version/?limit=10&offset=10"
}
```

Response JSON Object

- **next** (*string*) – URI for next set of Versions.
- **previous** (*string*) – URI for previous set of Versions.
- **count** (*integer*) – Total number of Versions.
- **results** (*array*) – Array of `Version` objects.

Query Parameters

- **project__slug** (*string*) – Narrow to the versions for a specific `Project`
- **active** (*boolean*) – Pass `true` or `false` to show only active or inactive versions. By default, the API returns all versions.

Version detail

GET /api/v2/version/(int: *id*)/
Retrieve details of a single version.

```
{
    "id": 1437428,
    "slug": "stable",
    "verbose_name": "stable",
    "built": true,
    "active": true,
```

(continues on next page)

(continued from previous page)

```
    "type": "tag",
    "identifier":␣
↪"3a6b3995c141c0888af6591a59240ba5db7d9914
↪",
    "downloads": {
        "pdf": "//readthedocs.
↪org/projects/pip/downloads/pdf/stable/",
        "htmlzip": "//readthedocs.org/
↪projects/pip/downloads/htmlzip/stable/",
        "epub": "//readthedocs.
↪org/projects/pip/downloads/epub/stable/"
    },
    "project": {PROJECT},
}
```

Response JSON Object

- **id** (*integer*) – The ID of the version
- **verbose_name** (*string*) – The name of the version.
- **slug** (*string*) – The version slug.
- **built** (*string*) – Whether this version has been built
- **active** (*string*) – Whether this version is still active
- **type** (*string*) – The type of this version (typically "tag" or "branch")
- **identifier** (*string*) – A version control identifier for this version (eg. the commit hash of the tag)
- **downloads** (*array*) – URLs to downloads of this version's documentation
- **project** (*object*) – Details of the `Project` for this version.

Status Codes

- 200 OK – no error
- 404 Not Found – There is no `Version` with this ID

Builds

Builds are created by Read the Docs whenever a `Project` has its documentation built. Frequently this happens automatically via a web hook but can be triggered manually.

Builds can be viewed in the build screen for a project. For example, here is Pip's build screen.

Build list

GET /api/v2/build/

Retrieve details of builds ordered by most recent first

Example request:

```
$ curl https://readthedocs.
↪org/api/v2/build/?project__slug=pip
```

Example response:

```
{
    "count": 100,
    "next": null,
    "previous": null,
    "results": [BUILDS]
}
```

Response JSON Object

- **next** (*string*) – URI for next set of Builds.
- **previous** (*string*) – URI for previous set of Builds.
- **count** (*integer*) – Total number of Builds.
- **results** (*array*) – Array of `Build` objects.

Query Parameters

- **project__slug** (*string*) – Narrow to builds for a specific `Project`
- **commit** (*string*) – Narrow to builds for a specific `commit`

Build detail

GET /api/v2/build/(int: *id*)/

Retrieve details of a single build.

```
{
    "id": 7367364,
    "date": "2018-06-19T15:15:59.135894",
    "length": 59,
    "type": "html",
    "state": "finished",
    "success": true,
    "error": "",
    "commit":␣
↪"6f808d743fd6f6907ad3e2e969c88a549e76db30
↪",
    "docs_
↪url": "http://pip.pypa.io/en/latest/",
    "project": 13,
    "project_slug": "pip",
    "version": 3681,
    "version_slug": "latest",
    "commands": [
        {
            "description": "",
            "start_
↪time": "2018-06-19T20:16:00.951959",
```

(continues on next page)

(continued from previous page)

```
            "exit_code": 0,
            "build": 7367364,
            "command": "git remote set-url␣
↪origin git://github.com/pypa/pip.git",
            "run_time": 0,
            "output": "",
            "id": 42852216,

          ␣
↪ "end_time": "2018-06-19T20:16:00.969170"
        },
        ...
    ],
    ...
}
```

Response JSON Object

- **id** (*integer*) – The ID of the build
- **date** (*string*) – The ISO-8601 datetime of the build.
- **length** (*integer*) – The length of the build in seconds.
- **type** (*string*) – The type of the build (one of "html", "pdf", "epub")
- **state** (*string*) – The state of the build (one of "triggered", "building", "installing", "cloning", or "finished")
- **success** (*boolean*) – Whether the build was successful
- **error** (*string*) – An error message if the build was unsuccessful
- **commit** (*string*) – A version control identifier for this build (eg. the commit hash)
- **docs_url** (*string*) – The canonical URL of the build docs
- **project** (*integer*) – The ID of the project being built
- **project_slug** (*string*) – The slug for the project being built
- **version** (*integer*) – The ID of the version of the project being built
- **version_slug** (*string*) – The slug for the version of the project being built
- **commands** (*array*) – Array of commands for the build with details including output.

Status Codes

- 200 OK – no error
- 404 Not Found – There is no `Build` with this ID

Some fields primarily used for UI elements in Read the Docs are omitted.

Undocumented resources and endpoints

There are some undocumented endpoints in the API. These should not be used and could change at any time. These include:

- The search API (`/api/v2/search/`)
- Endpoints for returning footer and version data to be injected into docs. (`/api/v2/footer_html`)
- Endpoints used for advertising (`/api/v2/sustainability/`)
- Any other endpoints not detailed above.

1.12.2 API v1 (deprecated)

We have a limited public API that is available for you to get data out of the site. This document covers only part of the API provided. We have plans to create a read/write API, so that you can easily automate interactions with your project.

> **Warning:** This API is **deprecated** but will be maintained through at least January 2019. Please connect with *API v2*.

A basic API client using slumber

You can use Slumber to build basic API wrappers in python. Here is a simple example of using slumber to interact with the RTD API:

```
from __future__ import print_function
import slumber
import json

show_objs = True
api = slumber.API(base_
↪url='https://readthedocs.org/api/v1/')

val = api.project.get(slug='pip')

if show_objs:
    for obj in val['objects']:
        print(json.dumps(obj, indent=4))
else:
    print(json.dumps(val, indent=4))
```

Alternatively you can try with the following value:

```
# fetch project pip without metadata.
val = api.project('pip').get()
```

(continues on next page)

(continued from previous page)

```
# get a specific build
val = api.build(2592228).get()

# get the build of a specific project.
val = api.
↪build.get(project__slug='read-the-docs')

# get a specific user by `username`
val = api.user.get(username='eric')

#val = api.version('pip').get()
#val = api.version('pip').get(slug='1.0.1')
```

API Endpoints

Feel free to use cURL and python to look at formatted json examples. You can also look at them in your browser, if it handles returned json.

```
curl␣
↪https://readthedocs.org/api/v1/project/
↪pip/?format=json | python -m json.tool
```

Doc Search

GET /api/v2/docsearch/

string project Required. The slug of a project.

string version Required. The slug of the version for this project.

string q Required. The search query

You can search a specific set of documentation using our doc search endpoint. It returns data in the format of Elastic Search, which requires a bit of traversing to use.

In the future we might change the format of this endpoint to make it more abstract.

An example URL: https://readthedocs.org/api/v2/docsearch/?project=docs&version=latest&q=subdomains

Results:

```
{
    "results": {
        "hits": {
            "hits": [
                {
                    "fields": {
                        "link
↪": "http://localhost:9999/docs/test-docs/
↪en/latest/history/classes/coworking",
```

(continues on next page)

(continued from previous page)

```
                        "path": [
                            "history/classes/coworking"
                        ],
                        "project": [
                            "test-docs"
                        ],
                        "title": [
                            "PIE coworking"
                        ],
                        "version": [
                            "latest"
                        ]
                    },
                    "highlight": {
                        "content": [
                            "\nhelp fund more endeavors.
↪Beta <em>test</em>  This first iteration
↪of PIE was a very underground project"
                        ]
                    }
                },
            ],
            "max_score": 0.47553805,
            "total": 2
        }
    }
}
```

Root

GET /api/v1/

Retrieve a list of resources.

```
{
    "build": {
        "list_endpoint": "/api/v1/build/",
        "schema": "/api/v1/build/schema/"
    },
    "file": {
        "list_endpoint": "/api/v1/file/",
        "schema": "/api/v1/file/schema/"
    },
    "project": {
        "list_endpoint": "/api/v1/project/",
        "schema": "/api/v1/project/schema/"
    },
    "user": {
        "list_endpoint": "/api/v1/user/",
        "schema": "/api/v1/user/schema/"
    },
```

(continues on next page)

(continued from previous page)

```
    "version": {
        "list_endpoint": "/api/v1/version/",
        "schema": "/api/v1/version/schema/"
    }
}
```

Response JSON Object

- **list_endpoint** (*string*) – API endpoint for resource.
- **schema** (*string*) – API endpoint for schema of resource.

Builds

GET /api/v1/build/

Retrieve a list of Builds.

```
{
    "meta": {
        "limit": 20,
        "next": "/api/v1/build/?limit=20&offset=20",
        "offset": 0,
        "previous": null,
        "total_count": 86684
    },
    "objects": [BUILDS]
}
```

Response JSON Object

- **limit** (*integer*) – Number of Builds returned.
- **next** (*string*) – URI for next set of Builds.
- **offset** (*integer*) – Current offset used for pagination.
- **previous** (*string*) – URI for previous set of Builds.
- **total_count** (*integer*) – Total number of Builds.
- **objects** (*array*) – Array of *Build* objects.

Build

GET /api/v1/build/{id}/

Parameters

- **id** – A Build id.

Retrieve a single Build.

```
{
    "date": "2012-03-12T19:58:29.307403",
    "error": "SPHINX ERROR",
    "id": "91207",
    "output": "SPHINX OUTPUT",
    "project": "/api/v1/project/2599/",
    "resource_uri": "/api/v1/build/91207/",
    "setup
↪": "HEAD is now at cd00d00 Merge pull␣
↪request #181 from Nagyman/solr_setup\n",
    "setup_error": "",
    "state": "finished",
    "success": true,
    "type": "html",
    "version": "/api/v1/version/37405/"
}
```

Response JSON Object

- **date** (*string*) – Date of Build.
- **error** (*string*) – Error from Sphinx build process.
- **id** (*string*) – Build id.
- **output** (*string*) – Output from Sphinx build process.
- **project** (*string*) – URI for Project of Build.
- **resource_uri** (*string*) – URI for Build.
- **setup** (*string*) – Setup output from Sphinx build process.
- **setup_error** (*string*) – Setup error from Sphinx build process.
- **state** (*string*) – "triggered", "building", or "finished"
- **success** (*boolean*) – Was build successful?
- **type** (*string*) – Build type ("html", "pdf", "man", or "epub")
- **version** (*string*) – URI for Version of Build.

Files

GET /api/v1/file/

Retrieve a list of Files.

```
{
    "meta": {
        "limit": 20,
        "next
↪": "/api/v1/file/?limit=20&offset=20",
        "offset": 0,
        "previous": null,
```

(continues on next page)

(continued from previous page)

```
        "total_count": 32084
    },
    "objects": [FILES]
}
```

Response JSON Object

- **limit** (*integer*) – Number of Files returned.
- **next** (*string*) – URI for next set of Files.
- **offset** (*integer*) – Current offset used for pagination.
- **previous** (*string*) – URI for previous set of Files.
- **total_count** (*integer*) – Total number of Files.
- **objects** (*array*) – Array of *File* objects.

File

GET /api/v1/file/{id}/

Parameters

- **id** – A File id.

  Retrieve a single File.

```
{
    "absolute_url":␣
↪"/docs/keystone/en/latest/search.html",
    "id": "332692",
    "name": "search.html",
    "path": "search.html",
    "project": {PROJECT},
    "resource_uri": "/api/v1/file/332692/"
  }
```

Response JSON Object

- **absolute_url** (*string*) – URI for actual file (not the File object from the API.)
- **id** (*string*) – File id.
- **name** (*string*) – Name of File.
- **path** (*string*) – Name of Path.
- **project** (*object*) – A *Project* object for the file's project.
- **resource_uri** (*string*) – URI for File object.

Projects

GET /api/v1/project/

Retrieve a list of Projects.

```
{
    "meta": {
        "limit": 20,
        "next
↪": "/api/v1/project/?limit=20&offset=20",
        "offset": 0,
        "previous": null,
        "total_count": 2067
    },
    "objects": [PROJECTS]
}
```

Response JSON Object

- **limit** (*integer*) – Number of Projects returned.
- **next** (*string*) – URI for next set of Projects.
- **offset** (*integer*) – Current offset used for pagination.
- **previous** (*string*) – URI for previous set of Projects.
- **total_count** (*integer*) – Total number of Projects.
- **objects** (*array*) – Array of *Project* objects.

Project

GET /api/v1/project/{id}

Parameters

- **id** – A Project id.

Retrieve a single Project.

```
{
    "absolute_url": "/projects/docs/",
    "analytics_code": "",
    "copyright": "",
    "crate_url": "",
    "default_branch": "",
    "default_version": "latest",
    "description
↪": "Make docs.readthedocs.io work :D",
    "django_packages_url": "",
    "documentation_type": "sphinx",
    "id": "2599",
    "modified_
↪date": "2012-03-12T19:59:09.130773",
    "name": "docs",
```

(continues on next page)

(continued from previous page)

```
    "project_url": "",
    "pub_date": "2012-02-19T18:10:56.582780",
    "repo": "git://github.com/rtfd/readthedocs.org",
    "repo_type": "git",
    "requirements_file": "",
    "resource_uri": "/api/v1/project/2599/",
    "slug": "docs",
    "subdomain": "http://docs.readthedocs.io/",
    "suffix": ".rst",
    "theme": "default",
    "use_virtualenv": false,
    "users": [
        "/api/v1/user/1/"
    ],
    "version": ""
}
```

Response JSON Object

- **absolute_url** (*string*) – URI for project (not the Project object from the API.)
- **analytics_code** (*string*) – Analytics tracking code.
- **copyright** (*string*) – Copyright
- **crate_url** (*string*) – Crate.io URI.
- **default_branch** (*string*) – Default branch.
- **default_version** (*string*) – Default version.
- **description** (*string*) – Description of project.
- **django_packages_url** (*string*) – Djangopackages.com URI.
- **documentation_type** (*string*) – Either "sphinx" or "sphinx_html".
- **id** (*string*) – Project id.
- **modified_date** (*string*) – Last modified date.
- **name** (*string*) – Project name.
- **project_url** (*string*) – Project homepage.
- **pub_date** (*string*) – Last published date.
- **repo** (*string*) – URI for VCS repository.
- **repo_type** (*string*) – Type of VCS repository.
- **requirements_file** (*string*) – Pip requirements file for packages needed for building docs.
- **resource_uri** (*string*) – URI for Project.

- **slug** (*string*) – Slug.
- **subdomain** (*string*) – Subdomain.
- **suffix** (*string*) – File suffix of docfiles. (Usually ".rst".)
- **theme** (*string*) – Sphinx theme.
- **use_virtualenv** (*boolean*) – Build project in a virtualenv? (True or False)
- **users** (*array*) – Array of readthedocs.org user URIs for administrators of Project.
- **version** (*string*) – DEPRECATED.

Users

GET /api/v1/user/

Retrieve List of Users

```
{
    "meta": {
        "limit": 20,
        "next
↪": "/api/v1/user/?limit=20&offset=20",
        "offset": 0,
        "previous": null,
        "total_count": 3200
    },
    "objects": [USERS]
}
```

Response JSON Object

- **limit** (*integer*) – Number of Users returned.
- **next** (*string*) – URI for next set of Users.
- **offset** (*integer*) – Current offset used for pagination.
- **previous** (*string*) – URI for previous set of Users.
- **total_count** (*integer*) – Total number of Users.
- **USERS** (*array*) – Array of *User* objects.

User

GET /api/v1/user/{id}/

Parameters

- **id** – A User id.

Retrieve a single User

```
{
    "id": "1",
    "resource_uri": "/api/v1/user/1/",
    "username": "testuser"
}
```

Response JSON Object

- **id** (*string*) – User id.
- **resource_uri** (*string*) – URI for this user.
- **username** (*string*) – User name.

Important: This API was changed after the initial release to remove private fields.

Versions

GET /api/v1/version/

Retrieve a list of Versions.

```
{
    "meta": {
        "limit": 20,
        "next
↪": "/api/v1/version/?limit=20&offset=20",
        "offset": 0,
        "previous": null,
        "total_count": 16437
    },
    "objects": [VERSIONS]
}
```

Response JSON Object

- **limit** (*integer*) – Number of Versions returned.
- **next** (*string*) – URI for next set of Versions.
- **offset** (*integer*) – Current offset used for pagination.
- **previous** (*string*) – URI for previous set of Versions.
- **total_count** (*integer*) – Total number of Versions.
- **objects** (*array*) – Array of *Version* objects.

Version

GET /api/v1/version/{id}

Parameters

- **id** – A Version id.

  Retrieve a single Version.

```
{
    "active": false,
    "built": false,
    "id": "12095",
    "identifier
↪": "remotes/origin/zip_importing",
    "project": {PROJECT},
  ␣
↪"resource_uri": "/api/v1/version/12095/",
    "slug": "zip_importing",
    "uploaded": false,
    "verbose_name": "zip_importing"
}
```

Response JSON Object

- **active** (*boolean*) – Are we continuing to build docs for this version?
- **built** (*boolean*) – Have docs been built for this version?
- **id** (*string*) – Version id.
- **identifier** (*string*) – Identifier of Version.
- **project** (*object*) – A *Project* object for the version's project.
- **resource_uri** (*string*) – URI for Version object.
- **slug** (*string*) – String that uniquely identifies a project
- **uploaded** (*boolean*) – Were docs uploaded? (As opposed to being build by Read the Docs.)
- **verbose_name** (*string*) – Usually the same as Slug.

Filtering Examples

File Search

```
https://readthedocs.org/api/v1/file/
↪search/?format=json&q=virtualenvwrapper
```

GET /api/v1/file/search/?q={search_term}

Parameters

- **search_term** – Perform search with this term.

  Retrieve a list of File objects that contain the search term.

```
{
    "objects": [
        {
↪  "absolute_url": "/docs/python-guide/en/
↪latest/scenarios/virtualenvs/index.html",
            "id": "375539",
            "name": "index.html",
            "path
↪": "scenarios/virtualenvs/index.html",
            "project": {
                "absolute_
↪url": "/projects/python-guide/",
                "analytics_code": null,
                "copyright": "Unknown",
                "crate_url": "",
                "default_branch": "",
↪              "default_version": "latest",
                "description
↪": "[WIP] Python best practices...",
                "django_packages_url": "",
↪  "documentation_type": "sphinx_htmldir",
                "id": "530",
                "modified_
↪date": "2012-03-13T01:05:30.191496",
                "name": "python-guide",
                "project_url": "",
↪"pub_date": "2011-03-20T19:40:03.599987",
                "repo": "git://github.
↪com/kennethreitz/python-guide.git",
                "repo_type": "git",
                "requirements_file": "",
↪  "resource_uri": "/api/v1/project/530/",
                "slug": "python-guide",
                "subdomain
↪": "http://python-guide.readthedocs.io/",
                "suffix": ".rst",
                "theme": "kr",
                "use_virtualenv": false,
                "users": [
                    "/api/v1/user/130/"
                ],
                "version": ""
            },
↪  "resource_uri": "/api/v1/file/375539/",
↪   "text": "...<span class=\"highlighted\
↪">virtualenvwrapper</span>\n..."
        },
        ...
    ]
}
```

Anchor Search

```
https://readthedocs.org/api/
↪v1/file/anchor/?format=json&q=virtualenv
```

GET /api/v1/file/anchor/?q={search_term}

Parameters

- **search_term** – Perform search of files containing anchor text with this term.

  Retrieve a list of absolute URIs for files that contain the search term.

```
{
    "objects": [
        "http//django-
↪fab-deploy.readthedocs.io/en/latest/...",
        "http//dimagi-
↪deployment-tools.readthedocs.io/en/...",
        "http/
↪/openblock.readthedocs.io/en/latest/
↪install/base_install.html#virtualenv",
        ...
    ]
}
```

1.13 Embed API

Read the Docs allow you to embed content from any of the projects we host. This allows for content re-use across sites, making sure the content is always up to date.

1.13.1 Workflow

There are many uses of our Embed API. One of our favorites is for inline help. We have started using this on Read the Docs itself to provide inline help on our main site pages. This allows us to keep the official documentation as the single source of truth, while having great inline help in our application as well.

We recommend you point at **tagged releases** instead of `latest`. Tags don't change over time, so you don't have to worry about the content you are embedding disappearing.

Note: All relative links to pages contained in the remote content will continue to point at the remote page.

1.13.2 How to use it

Sphinx Extension

You can embed content directly in Sphinx with builds on Read the Docs. We support default configuration variables for your `conf.py`:

- readthedocs_embed_project
- readthedocs_embed_version
- readthedocs_embed_doc

These are overridable per-call as well. Then you simply use the directive:

```
# All arguments
.. readthedocs-embed::
    :project: myproject
    :version: latest
    :doc: index
    :section: User Guide

# Or with some defaults
.. readthedocs-embed::
    :doc: index
    :section: User Guide
```

Javascript

We provide a Javascript library that you can embed on any site. An example:

```
<!-- In your <head> -->
<link rel="stylesheet" href="http:/
↪/localhost:5555/static/css/public.css">
<script type="text/javascript
↪" src="http://localhost:5555/static/
↪javascript/bundle-public.js"></script>
<script>
embed = ReadTheDocs.Embed(project, version)
rtd.into(
↪'page', 'section', function(content){
    $('#foobar').html(content)
})

</script>

<!-- In your <body> -->
<div id="help_container">
  <a href="#" class="readthedocs-
↪help-embed" data-section="How␣
↪we envision versions working">(Help)</a>
</div>
```

This will provide a pop-out for a user with the `How we envision versions working` section of the `versions` page. You can see this in action here:

Note: All Read the Docs pages already have the library loaded, so you can ignore the `link` and first `script` tags on all documentation.

Warning: We currently do not provide caching on this API. If the remote source you are including changes their page structure or deletes the content, your embed will break.

In Version 2 of this API we will provide a full-formed workflow that will stop this from happening.

1.13.3 Example API Response

Pure API use will return JSON:

```
{
    "content": [
        "<div class=\"section\
↪" id=\"encoded-data\">\n<h2>Encoded␣
↪Data?<a class=\"headerlink\" href=\
↪"/docs/requests/en/latest/community/faq.
↪html#encoded-data\" title=\"Permalink␣
↪to this headline\">\u00b6</a></h2>\n<p>
↪Requests automatically decompresses gzip-
↪encoded responses, and does\nits best to␣
↪decode response content to unicode when␣
↪possible.</p>\n<p>You can get direct␣
↪access to the raw response (and even the␣
↪socket),\nif needed as well.</p>\n</div>"
    ],
    "wrapped": [
        "\n<div class=\"readthedocs-embed-
↪wrapper\">\n    <div class=\"readthedocs-
↪embed-content\">\n        <div␣
↪class=\"section\" id=\"encoded-data\">
↪\n<h2>Encoded Data?<a class=\"headerlink\
↪" href=\"/docs/requests/en/latest/
↪community/faq.html#encoded-data\" title=\
↪"Permalink to this headline\">\u00b6
↪</a></h2>\n<p>Requests automatically␣
↪decompresses gzip-encoded responses,
↪ and does\nits best to decode response␣
↪content to unicode when possible.</p>\n
↪<p>You can get direct access to the raw response (and even the socket),\nif needed␣
↪as well.</p>\n</div>\n    </div>\n    <div class=\"readthedocs-embed-badge\">\n    ␣
↪   Embedded from <a href=\"/docs/requests/en/latest/community/faq.html\">Read the␣
↪Docs</a>\n    </div>\n</div>\n    "
    ],
    "meta": {
```

(continues on next page)

(continued from previous page)

```
        "project": "requests",
        "doc": "community/faq",
        "section": "Encoded Data?",
        "version": "latest",
        "modified
↪": "Wed, 04 Feb 2015 08:59:59 GMT"
    },
    "url": "/docs/
↪requests/en/latest/community/faq.html",
    "headers": [
        {
↪         "Frequently Asked Questions": "#"
        },
        {
↪          "Encoded Data?": "#encoded-data"
        },
        {
            "Custom␣
↪User-Agents?": "#custom-user-agents"
        },
        {
↪ "Why not Httplib2?": "#why-not-httplib2"
        },
        {
↪ "Python 3 Support?": "#python-3-support"
        },
        {
            "What are \u201chostname␣
↪doesn\u2019t match\u201d errors?":␣
↪"#what-are-hostname-doesn-t-match-errors"
        }
    ]
}
```

1.14 Webhooks

The primary method that Read the Docs uses to detect changes to your documentation is through the use of *webhooks*. Webhooks are configured with your repository provider, such as GitHub, Bitbucket or GitLab, and with each commit, merge, or other change to your repository, Read the Docs is notified. When we receive a webhook notification, we determine if the change is related to an active version for your project, and if it is, a build is triggered for that version.

1.14.1 Webhook Integrations

You'll find a list of configured webhook integrations on your project's admin dashboard, under **Integrations**. You can select any of these integrations to see the *integration detail page*. This page has additional configuration details and a list of HTTP exchanges that have taken place for the integration.

You need this information for the URL, webhook, or Payload URL needed by the repository provider such as GitHub, GitLab, or Bitbucket.

1.14.2 Webhook Creation

If you have *connected your Read the Docs account* to GitHub, Bitbucket, or GitLab, a webhook will be set up automatically for your repository. However, if your project was not imported through a connected account, you may need to manually configure a webhook for your project.

To manually set up a webhook, click **Add integration** on your project's **Integrations** Admin dashboard page and select the integration type you'd like to add. After you have added the integration, you'll see a link to information about the integration.

As an example, the URL pattern looks like this: *readthedocs.org/api/v2/webhook/<project-name>/<id>/*.

Use this URL when setting up a new webhook with your provider – these steps vary depending on the provider:

GitHub

- Go to the **Settings** page for your project
- Click **Webhooks** and then **Add webhook**
- For **Payload URL**, use the URL of the integration on Read the Docs, found on the the project's **Integrations** Admin dashboard page
- For **Content type**, both *application/json* and *application/x-www-form-urlencoded* work
- Select **Just the push event**
- Finish by clicking **Add webhook**

  You can verify if the webhook is working at the bottom of the GitHub page under **Recent Deliveries**. If you see a Response 200, then the webhook is correctly configured. For a 403 error, it's likely that the Payload URL is incorrect.

Note: The webhook token, intended for the GitHub **Secret** field, is not yet implemented.

Bitbucket

- Go to the **Settings** page for your project
- Click **Webhooks** and then **Add webhook**
- For **URL**, use the URL of the integration on Read the Docs, found on the **Dashboard > Admin > Integrations** page
- Under **Triggers**, **Repository push** should be selected
- Finish by clicking **Save**

GitLab

- Go to the **Settings** page for your project
- Click **Integrations**
- For **URL**, use the URL of the integration on Read the Docs, found on the **Dashboard > Admin > Integrations** page
- Leave the default **Push events** selected and mark **Tag push events** also
- Finish by clicking **Add Webhook**

1.14.3 Using the generic API integration

For repositories that are not hosted with a supported provider, we also offer a generic API endpoint for triggering project builds. Similar to webhook integrations, this integration has a specific URL, found on the project's **Integrations** Admin dashboard page on readthedocs.org.

Token authentication is required to use the generic endpoint, you will find this token on the integration details page. The token should be passed in as a request parameter, either as form data or as part of JSON data input.

Parameters

This endpoint accepts the following arguments during an HTTP POST:

branches The names of the branches to trigger builds for. This can either be an array of branch name strings, or just a single branch name string.

Default: **latest**

token The integration token. You'll find this value on the project's **Integrations** Admin dashboard page.

For example, the cURL command to build the `dev` branch, using the token `1234`, would be:

```
curl -X POST -d "branches=dev
↪" -d "token=1234" https://readthedocs.
↪org/api/v2/webhook/example-project/1/
```

A command like the one above could be called from a cron job or from a hook inside Git, Subversion, Mercurial, or Bazaar.

Authentication

This endpoint requires authentication. If authenticating with an integration token, a check will determine if the token is valid and matches the given project. If instead an authenticated user is used to make this request, a check will be performed to ensure the authenticated user is an owner of the project.

1.14.4 Debugging webhooks

If you are experiencing problems with an existing webhook, you may be able to use the integration detail page to help debug the issue. Each project integration, such as a webhook or the generic API endpoint, stores the HTTP exchange that takes place between Read the Docs and the external source. You'll find a list of these exchanges in any of the integration detail pages.

1.14.5 Resyncing webhooks

It might be necessary to re-establish a webhook if you are noticing problems. To resync a webhook from Read the Docs, visit the integration detail page and follow the directions for re-syncing your repository webhook.

1.15 Badges

Badges let you show the state of your documentation to your users. They are great for embedding in your README, or putting inside your actual doc pages.

1.15.1 Status Badges

They will display in green for passing, red for failing, and yellow for unknown states.

Here are a few examples:

You can see it in action in the Read the Docs README. They will link back to your project's documentation page on Read the Docs.

1.15.2 Style

Now you can pass the `style` GET argument, to get custom styled badges same as you would for shields.io. If no argument is passed, `flat` is used as default.

STYLE	BADGE
flat	
flat-square	
for-the-badge	
plastic	
social	

1.15.3 Project Pages

You will now see badges embedded in your project page. The default badge will be pointed at the *default version* you have specified for your project. The badge URLs look like this:

```
https://readthedocs.org/projects/
↪pip/badge/?version=latest&style=plastic
```

You can replace the version argument with any version that you want to show a badge for. If you click on the badge icon, you will be given snippets for RST, Markdown, and HTML; to make embedding it easier.

If you leave the version argument off, it will default to your latest version. This is probably best to include in your README, since it will stay up to date with your Read the Docs project:

```
https://readthedocs.org/projects/pip/badge/
```

1.16 Localization of Documentation

Note: This feature only applies to Sphinx documentation. We are working to bring it to our other documentation backends.

Read the Docs supports hosting your docs in multiple languages. There are two different things that we support:

- A single project written in another language

- A project with translations into multiple languages

1.16.1 Single project in another language

It is easy to set the *Language* of your project. On the project *Admin* page (or *Import* page), simply select your desired *Language* from the dropdown. This will tell Read the Docs that your project is in the language. The language will be represented in the URL for your project.

For example, a project that is in Spanish will have a default URL of `/es/latest/` instead of `/en/latest/`.

Note: You must commit the `.po` files for Read the Docs to translate your documentation.

1.16.2 Project with multiple translations

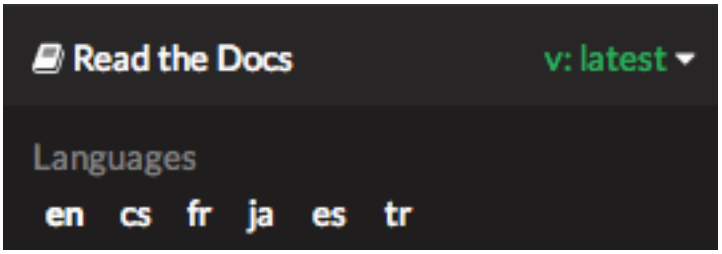
This situation is a bit more complicated. To support this, you will have one parent project and a number of projects marked as translations of that parent. Let's use `phpmyadmin` as an example.

The main `phpmyadmin` project is the parent for all translations. Then you must create a project for each translation, for example `phpmyadmin-spanish`. You will set the *Language* for `phpmyadmin-spanish` to `Spanish`. In the parent projects *Translations* page, you will say that `phpmyadmin-spanish` is a translation for your project.

This has the results of serving:

- `phpmyadmin` at `http://phpmyadmin.readthedocs.io/en/latest/`
- `phpmyadmin-spanish` at `http://phpmyadmin.readthedocs.io/es/latest/`

It also gets included in the Read the Docs flyout:



Note: The default language of a custom domain is determined by the language of the parent project that the domain was configured on. See alternate_domains for more information.

Note: You can include multiple translations in the same repository, with same `conf.py` and `.rst` files, but each project must specify the language to build for those docs.

Note: You can read *Manage Translations* to understand the whole process for a documentation with multiples languages in the same repository and how to keep the translations updated on time.

1.17 Version Control System Integration

Note: We plan to implement a new approach regarding the Theme Context as a whole, although the VCS documentation page will still be valid, we prefer the users to move in that direction.

If you want to integrate editing into your own theme, you will have to declare few variables inside your configuration file `conf.py` in the `'html_context'` setting, for the template to use them.

More information can be found on Sphinx documentation.

1.17.1 GitHub

If you want to integrate GitHub, you can put the following snippet into your `conf.py`:

```
html_context = {
    "display_
↪github": True, # Integrate GitHub
    "github_user": "MyUserName", # Username
    "github_repo": "MyDoc", # Repo name
    "github_version": "master", # Version
    "conf_py_path": "/source/",
↪ # Path in the checkout to the docs root
}
```

It can be used like this:

```
{% if display_github %}
    <li><a href="https://github.
↪com/{{ github_user }}/{{ github_repo }}
    /blob/{{ github_version␣
↪}}{{ conf_py_path }}{{ pagename }}.rst">
    Show on GitHub</a></li>
{% endif %}
```

1.17.2 Bitbucket

If you want to integrate Bitbucket, you can put the following snippet into your `conf.py`:

```
html_context = {
    "display_
↪bitbucket": True, # Integrate Bitbucket
    "bitbucket_
↪user": "MyUserName", # Username
    "bitbucket_repo": "MyDoc", # Repo name
  ␣
↪ "bitbucket_version": "master", # Version
    "conf_py_path": "/source/",
↪ # Path in the checkout to the docs root
}
```

It can be used like this:

```
{% if display_bitbucket %}
    <a href="https://bitbucket.org/
↪{{ bitbucket_user }}/{{ bitbucket_repo }}
    /src/{{ bitbucket_version}
↪}{{ conf_py_path }}{{ pagename }}.rst'"
    class="icon␣
↪icon-bitbucket"> Edit on Bitbucket</a>
{% endif %}
```

1.17.3 Gitlab

If you want to integrate Gitlab, you can put the following snippet into your `conf.py`:

```
html_context = {
    "display_
↪gitlab": True, # Integrate Gitlab
    "gitlab_user": "MyUserName", # Username
    "gitlab_repo": "MyDoc", # Repo name
    "gitlab_version": "master", # Version
    "conf_py_path": "/source/",
↪ # Path in the checkout to the docs root
}
```

It can be used like this:

```
{% if display_gitlab %}
    <a href="https:/
↪/{{ gitlab_host|default("gitlab.com") }}/
    {{ gitlab_user }}/{{␣
↪gitlab_repo }}/blob/{{ gitlab_version }}
    {{ conf_py_path }}{{ pagename␣
↪}}{{ suffix }}" class="fa fa-gitlab">
    Edit on GitLab</a>
{% endif %}
```

1.17.4 Additional variables

- `'pagename'` - Sphinx variable representing the name of the page you're on.

1.18 Subprojects

Projects can be configured in a nested manner, by configuring a project as a *subproject* of another project. This allows for documentation projects to share a search index and a namespace or custom domain, but still be maintained independently.

For example, a parent project, `Foo` is set up with a subproject, `Bar`. The documentation for `Foo` will be available at:

https://foo.readthedocs.io/en/latest/

The documentation for `Bar` will be available under this same path:

https://foo.readthedocs.io/projects/bar/en/latest/

1.18.1 Adding a Subproject

In the admin dashboard for your project, select "Subprojects" from the menu. From this page you can add a subproject by typing in the project slug.

1.18.2 Sharing a Custom Domain

Projects and subprojects can also be used to share a custom domain with a number of projects. To configure this, one project should be established as the parent project. This project will be configured with a custom domain. Projects can then be added as subprojects to this parent project.

If the example project `Foo` was set up with a custom domain, `docs.example.com`, the URLs for projects `Foo` and `Bar` would respectively be at: http://docs.example.com/en/latest/ and http://docs.example.com/projects/bar/en/latest/

1.18.3 Search

Projects that are configured as subprojects will share a search index with their parent and sibling projects. This is currently the only way to share search indexes between projects, we do not yet support sharing search indexes between arbitrary projects.

1.19 Conda Support

Warning: This feature is in a beta state. Please file an issue if you find anything wrong.

Read the Docs supports Conda as an environment management tool, along with Virtualenv. Conda support is useful for people who depend on C libraries, and need them installed when building their documentation.

This work was funded by Clinical Graphics – many thanks for their support of Open Source.

1.19.1 Activating Conda

Conda Support is the first feature enabled with *Read the Docs YAML Config*. You can enable it by creating a `readthedocs.yml` file in the root of your repository with the contents:

```
conda:
    file: environment.yml
```

This Conda environment will also have Sphinx and other build time dependencies installed. It will use the same order of operations that we support currently:

- Environment Creation (`conda env create`)
- Dependency Installation (Sphinx)

1.19.2 Custom Installs

If you are running a custom installation of Read the Docs, you will need the `conda` executable installed somewhere on your `PATH`. Because of the way `conda` works, we can't safely install it as a normal dependency into the normal Python virtualenv.

Warning: Installing conda into a virtualenv will override the `activate` script, making it so you can't properly activate that virtualenv anymore.

1.20 Canonical URLs

Canonical URLs allow people to have consistent page URLs for domains. This is mainly useful for search engines, so that they can send people to the correct page.

Read the Docs uses these in two ways:

- We point all versions of your docs at the "latest" version as canonical
- We point at the user specified canonical URL, generally a custom domain for your docs.

1.20.1 Example

Fabric hosts their docs on Read the Docs. They mostly use their own domain for them `http://docs.fabfile.org`. This means that Google will index both `http://fabric-docs.readthedocs.io` and `http://docs.fabfile.org` for their documentation.

Fabric will want to set `http://docs.fabfile.org` as their canonical URL. This means that when Google indexes `http://fabric-docs.readthedocs.io`, it will know that it should really point at `http://docs.fabfile.org`.

1.20.2 Enabling

You can set the canonical URL for your project in the Project Admin page. Check your `Domains` tab for the domains that we know about.

1.20.3 Implementation

If you look at the source code for documentation built after you set your canonical URL, you should see a bit of HTML like this:

```
<link rel=
↪"canonical" href="http://pip.readthedocs.
↪io/en/latest/installing.html">
```

1.20.4 Links

This is a good explanation of the usage of canonical URLs in search engines:

http://www.mattcutts.com/blog/seo-advice-url-canonicalization/

This is a good explanation for why canonical pages are good for SEO:

http://moz.com/blog/canonical-url-tag-the-most-important-advancement-in-seo-practices-since-sitemaps

1.21 Single Version Documentation

Single Version Documentation lets you serve your docs at a root domain. By default, all documentation served by Read the Docs has a root of `/<language>/<version>/`. But, if you enable the "Single Version" option for a project, its documentation will instead be served at `/`.

Warning: This means you can't have translations or multiple versions for your documentation.

You can see a live example of this at http://www.contribution-guide.org

1.21.1 Enabling

You can toggle the "Single Version" option on or off for your project in the Project Admin page. Check your dashboard for a list of your projects.

1.21.2 Effects

Links generated on Read the Docs will now point to the proper URL. For example, if pip was set as a "Single Version" project, then links to its documentation would point to `http://pip.readthedocs.io/` rather than the default `http://pip.readthedocs.io/en/latest/`.

Documentation at `/<language>/<default_version>/` will still be served for backwards compatibility reasons. However, our usage of *Canonical URLs* should stop these from being indexed by Google.

1.22 Privacy Levels

Read the Docs supports 3 different privacy levels on 2 different objects; Public, Protected, Private on Projects and Versions.

1.22.1 Understanding the Privacy Levels

Level	Detail	Listing	Search	Viewing
Private	No	No	No	Yes
Protected	Yes	No	No	Yes
Public	Yes	Yes	Yes	Yes

Note: With a URL to view the actual documentation, even private docs are viewable. This is because our architecture doesn't do any logic on documentation display, to increase availability.

Public

This is the easiest and most obvious. It is also the default. It means that everything is available to be seen by everyone.

Protected

Protected means that your object won't show up in Listing Pages, but Detail pages still work. For example, a Project that is Protected will not show on the homepage Recently Updated list, however, if you link directly to the project, you will get a 200 and the page will display.

Protected Versions are similar, they won't show up in your version listings, but will be available once linked to.

Private

Private objects are available only to people who have permissions to see them. They will not display on any list view, and will 404 when you link them to others.

1.23 User-defined Redirects

You can set up redirects for a project in your project dashboard's Redirects page.

1.23.1 Quick Summary

- Log into your readthedocs.org account.
- From your dashboard, select the project on which you wish to add redirects.
- From the project's top navigation bar, select the Admin tab.
- From the left navigation menu, select Redirects.

- In the form box "Redirect Type" select the type of redirect you want. See below for detail.
- Depending on the redirect type you select, enter FROM and/or TO URL as needed.
- When finished, click the SUBMIT Button.

Your redirects will be effective immediately.

1.23.2 Redirect Types

Prefix Redirects

The most useful and requested feature of redirects was when migrating to Read the Docs from an old host. You would have your docs served at a previous URL, but that URL would break once you moved them. Read the Docs includes a language and version slug in your documentation, but not all documentation is hosted this way.

Say that you previously had your docs hosted at `http://docs.example.com/dev/`, you move `docs.example.com` to point at Read the Docs. So users will have a bookmark saved to a page at `http://docs.example.com/dev/install.html`.

You can now set a *Prefix Redirect* that will redirect all 404's with a prefix to a new place. The example configuration would be:

```
Type: Prefix Redirect
From URL: /dev/
```

Your users query would now redirect in the following manner:

```
docs.example.com/dev/install.html ->
docs.example.com/en/latest/install.html
```

Where `en` and `latest` are the default language and version values for your project.

Note: In other words, a *Prefix Redirect* removes a prefix from the original URL. This prefix is removed from the rest of the URL's `path` after `/$lang/$version`. For example, if the URL is `/es/1.0/guides/tutorial/install.html` the "From URL's prefix" will be removed from `/guides/tutorial/install.html` part.

Page Redirects

A more specific case is when you move a page around in your docs. The old page will start 404'ing, and your users will be confused. *Page Redirects* let you redirect a specific page.

Say you move the `example.html` page into a subdirectory of examples: `examples/intro.html`. You would set the following configuration:

```
Type: Page Redirect
From URL: /example.html
To URL: /examples/intro.html
```

Note that the `/` at the start doesn't count the `/en/latest`, but just the user-controlled section of the URL.

Tip: *Page Redirects* can redirect URLs **outside** Read the Docs platform just by defining the "To URL" as the absolute URL you want to redirect to.

Exact Redirects

If you're redirecting from an old host AND you aren't maintaining old paths for your documents, a Prefix Redirect won't suffice and you'll need to create *Exact Redirects* to redirect from a specific URL, to a specific page.

Say you're moving `docs.example.com` to Read the Docs and want to redirect traffic from an old page at `http://docs.example.com/dev/install.html` to a new URL of `http://docs.example.com/en/latest/installing-your-site.html`.

The example configuration would be:

```
Type: Exact Redirect
From URL: /dev/install.html
To URL:␣
↪  /en/latest/installing-your-site.html
```

Your users query would now redirect in the following manner:

```
docs.example.com/dev/install.html ->
docs.example.
↪com/en/latest/installing-your-site.html
```

Note that you should insert the desired language for "en" and version for "latest" to achieve the desired redirect.

Exact Redirects could be also useful to redirect a whole sub-path to a different one by using a special `$rest` keyword in the "From URL". Let's say that you want to redirect your readers of your version `2.0` of your documentation under `/en/2.0/` because it's deprecated, to the newest `3.0` version of it at `/en/3.0/`.

This example would be:

```
Type: Exact Redirect
From URL: /en/2.0/$rest
To URL: /en/3.0/
```

The readers of your documentation will now be redirected as:

```
docs.example.com/en/2.0/dev/install.html ->
docs.example.com/en/3.0/dev/install.html
```

Tip: *Exact Redirects* can redirect URLs **outside** Read the Docs platform just by defining the "To URL" as the absolute URL you want to redirect to.

Sphinx Redirects

We also support redirects for changing the type of documentation Sphinx is building. If you switch between *HTMLDir* and *HTML*, your URL's will change. A page at `/en/latest/install.html` will be served at `/en/latest/install/`, or vice versa. The built in redirects for this will handle redirecting users appropriately.

1.23.3 Implementation

Since we serve documentation in a highly available way, we do not run any logic when we're serving documentation. This means that redirects will only happen in the case of a *404 File Not Found*.

In the future we might implement redirect logic in Javascript, but this first version is only implemented in the 404 handlers.

1.24 Automatic Redirects

Read the Docs supports redirecting certain URLs automatically. This is an overview of the set of redirects that are fully supported and will work into the future.

1.24.1 Root URL

A link to the root of your documentation will redirect to the *default version*, as set in your project settings. For example:

```
pip.readthedocs.
↪io -> pip.readthedocs.io/en/latest/
www.pip-installer.
↪org -> www.pip-installer.org/en/latest
```

This only works for the root URL, not for internal pages. It's designed to redirect people from http://pip.readthedocs.io/ to the default version of your documentation, since serving up a 404 here would be a pretty terrible user experience. (If your "develop" branch was designated as your default version, then it would redirect to http://pip.readthedocs.io/en/develop.) But, it's not a universal redirecting solution. So, for example, a link to an internal page like http://pip.readthedocs.io/usage.html doesn't redirect to http://pip.readthedocs.io/en/latest/usage.html.

The reasoning behind this is that RTD organizes the URLs for docs so that multiple translations and multiple versions of your docs can be organized logically and consistently for all projects that RTD hosts. For the way that RTD views docs, http://pip.readthedocs.io/en/latest/ is the root directory for your default documentation in English, not http://pip.readthedocs.io/. Just like http://pip.readthedocs.io/en/develop/ is the root for your development documentation in English.

Among all the multiple versions of docs, you can choose which is the "default" version for RTD to display, which usually corresponds to the git branch of the most recent official release from your project.

rtfd.org

Links to rtfd.org are treated the same way as above. They redirect the root URL to the default version of the project. They are intended to be easy and short for people to type.

1.24.2 Supported Top-Level Redirects

Note: These "implicit" redirects are supported for legacy reasons. We will not be adding support for any more magic redirects. If you want additional redirects, they should live at a prefix like *Redirecting to a Page*

The main challenge of URL routing in Read the Docs is handling redirects correctly. Both in the interest of redirecting older URLs that are now obsolete, and in the interest of handling "logical-looking" URLs (leaving out the lang_slug or version_slug shouldn't result in a 404), the following redirects are supported:

```
/           -> /en/latest/
/en/        -> /en/latest/
/latest/    -> /en/latest/
```

The language redirect will work for any of the defined `LANGUAGE_CODES` we support. The version redirect will work for supported versions.

1.24.3 Redirecting to a Page

You can link to a specific page and have it redirect to your default version. This is done with the `/page/` URL. For example:

```
pip.readthedocs.
↪io/page/quickstart.html -> pip.
↪readthedocs.io/en/latest/quickstart.html
www.pip-installer.
↪org/page/quickstart.html -> www.pip-
↪installer.org/en/latest/quickstart.html
```

This allows you to create links that are always up to date.

Another way to handle this is the *latest* version. You can set your `latest` version to a specific version and just always link to latest.

1.25 Content Embedding

Using the Read the Docs JavaScript Client, or with basic calls to our REST API, you can retrieve embeddable content for use on your own site. Content is embedded in an iframe element, primarily for isolation. To get example usage of the API, see the tools tab under an active project and select the page and section that you would like to test.

Note: The client library is still alpha quality. This guide is still lacking advanced usage of the library, and information on styling the iframe content. We plan to have more usage outlined as the library matures.

Example usage of the client library:

```
var embed = Embed();
embed.section(
    'read-the-docs', 'latest
↪', 'features', 'Read the Docs features',
    function (section) {
        section.insertContent($('#help'));
    }
);
```

1.26 Contributing to Read the Docs

You are here to help on Read the Docs? Awesome, feel welcome and read the following sections in order to know how to ask questions and how to work on something.

All members of our community are expected to follow our *Code of Conduct*. Please make sure you are welcoming

and friendly in all of our spaces.

1.26.1 Get in touch

- Ask usage questions ("How do I?") on StackOverflow.
- Report bugs, suggest features or view the source code on GitHub.
- Discuss topics on Gitter.
- On IRC find us at #readthedocs.

1.26.2 Contributing to development

If you want to deep dive and help out with development on Read the Docs, then first get the project installed locally according to the *Installation Guide*. After that is done we suggest you have a look at tickets in our issue tracker that are labelled Good First Issue. These are meant to be a great way to get a smooth start and won't put you in front of the most complex parts of the system.

If you are up to more challenging tasks with a bigger scope, then there are a set of tickets with a Feature or Improvement tag. These tickets have a general overview and description of the work required to finish. If you want to start somewhere, this would be a good place to start (make sure that the issue also have the Accepted label). That said, these aren't necessarily the easiest tickets. They are simply things that are explained. If you still didn't find something to work on, search for the Sprintable label. Those tickets are meant to be standalone and can be worked on ad-hoc.

When contributing code, then please follow the standard Contribution Guidelines set forth at contribution-guide.org.

We have a strict code style that is easy to follow since you just have to install pre-commit and it will automatically run different linting tools (autoflake, autopep8, docformatter, isort, prospector, unify and yapf) to check your changes before you commit them. `pre-commit` will let you know if there were any problems that is wasn't able to fix automatically.

To run the `pre-commit` command and check your changes:

```
$ pip install -U pre-commit
$ git add <your-modified-files>
$ pre-commit run
```

or to run against a specific file:

```
$ pre-commit run --files <file.py>
```

`pre-commit` can also be run as a git pre-commit hook. You can set this up with:

```
$ pre-commit install
```

After this installation, the next time you run `git commit` the `pre-commit run` command will be run immediately and will inform you of the changes and errors.

Note: Our code base is still maturing and the core team doesn't yet recommend running this as a pre-commit hook due to the number of changes this will cause while constructing a pull request. Independent pull requests with linting changes would be a great help to making this possible.

1.26.3 Triaging tickets

Here is a brief explanation on how we triage incoming tickets to get a better sense of what needs to be done on what end.

Note: You will need Triage permission on the project in order to do this. You can ask one of the members of the *Read the Docs Team* to give you access.

Initial triage

When sitting down to do some triaging work, we start with the list of untriaged tickets. We consider all tickets that do not have a label as untriaged. The first step is to categorize the ticket into one of the following categories and either close the ticket or assign an appropriate label. The reported issue ...

... is not valid If you think the ticket is invalid comment why you think it is invalid, then close the ticket. Tickets might be invalid if they were already fixed in the past or it was decided that the proposed feature will not be implemented because it does not conform with the overall goal of Read the Docs. Also if you happen to know that the problem was already reported, reference the other ticket that is already addressing the problem and close the duplicate.

Examples:

- *Builds fail when using matplotlib*: If the described issue was already fixed, then explain and instruct to re-trigger the build.
- *Provide way to upload arbitrary HTML files*: It was already decided that Read the Docs is not a dull hosting platform for HTML. So explain this and close the ticket.

... does not provide enough information Add the label **Needed: more information** if the reported issue does not

contain enough information to decide if it is valid or not and ask on the ticket for the required information to go forward. We will re-triage all tickets that have the label **Needed: more information** assigned. If the original reporter left new information we can try to re-categorize the ticket. If the reporter did not come back to provide more required information after a long enough time, we will close the ticket (this will be roughly about two weeks).

Examples:

- *My builds stopped working. Please help!* Ask for a link to the build log and for which project is affected.

... is a valid feature proposal If the ticket contains a feature that aligns with the goals of Read the Docs, then add the label **Feature**. If the proposal seems valid but requires further discussion between core contributors because there might be different possibilities on how to implement the feature, then also add the label **Needed: design decision**.

Examples:

- *Provide better integration with service XYZ*
- *Achieve world domination* (also needs the label **Needed: design decision**)

... is a small change to the source code If the ticket is about code cleanup or small changes to existing features would likely have the **Improvement** label. The distinction for this label is that these issues have a lower priority than a Bug, and aren't implementing new features.

Examples:

- *Refactor namedtuples to dataclasess*
- *Change font size for the project's title*

... is a valid problem within the code base: If it's a valid bug, then add the label **Bug**. Try to reference related issues if you come across any.

Examples:

- *Builds fail if conf.py contains non-ascii letters*

... is a currently valid problem with the infrastructure: Users might report about web server downtimes or that builds are not triggered. If the ticket needs investigation on the servers, then add the label **Operations**.

Examples:

- *Builds are not starting*

... is a question and needs answering: If the ticket contains a question about the Read the Docs platform or the code, then add the label **Support**.

Examples:

- *My account was set inactive. Why?*
- *How to use C modules with Sphinx autodoc?*
- *Why are my builds failing?*

... requires a one-time action on the server: Tasks that require a one time action on the server should be assigned the two labels **Support** and **Operations**.

Examples:

- *Please change my username*
- *Please set me as owner of this abandoned project*

After we finished the initial triaging of new tickets, no ticket should be left without a label.

Additional labels for categorization

Additionally to the labels already involved in the section above, we have a few more at hand to further categorize issues.

High Priority If the issue is urgent, assign this label. In the best case also go forward to resolve the ticket yourself as soon as possible.

Good First Issue This label marks tickets that are easy to get started with. The ticket should be ideal for beginners to dive into the code base. Better is if the fix for the issue only involves touching one part of the code.

Sprintable Sprintable are all tickets that have the right amount of scope to be handled during a sprint. They are very focused and encapsulated.

For a full list of available labels and their meanings, see *Overview of issue labels*.

Helpful links for triaging

Here is a list of links for contributors that look for work:

- Untriaged tickets: Go and triage them!
- Tickets labelled with Needed: more information: Come back to these tickets once in a while and close those that did not get any new information from the reporter. If new information is available, go and re-triage the ticket.
- Tickets labelled with Operations: These tickets are for contributors who have access to the servers.
- Tickets labelled with Support: Experienced contributors or community members with a broad knowledge about the project should handle those.
- Tickets labelled with Needed: design decision: Project leaders must take actions on these tickets. Otherwise no other contributor can go forward on them.

1.26.4 Helping on translations

If you wish to contribute translations, please do so on Transifex.

1.27 Roadmap

1.27.1 Process

Read the Docs has adopted the following workflow with regards to how we prioritize our development efforts and where the core development team focuses its time.

Triaging issues

Much of this is already covered in our guide on *Contributing to Read the Docs*, however to summarize the important pieces:

- New issues coming in will be triaged, but won't yet be considered part of our roadmap.
- If the issue is a valid bug, it will be assigned the `Accepted` label and will be prioritized, likely on an upcoming point release.
- If the issues is a feature or improvement, the issue might go through a design decision phase before being accepted and assigned to a milestone. This is a good time to discuss how to address the problem technically. Skipping this phase might result in your PR being blocked, sent back to design decision, or perhaps even discarded. It's best to be active here before submitting a PR for a feature or improvement.
- The core team will only work on accepted issues, and will give PR review priority to accepted issues. Pull requests addressing issues that are not on our roadmap are welcome, but we cannot guarantee review response, even for small or easy to review pull requests.

Milestones

We maintain two types of milestones: point release milestones for our upcoming releases, and group milestones, for blocks of work that have priority in the future.

Generally there are 2 or 3 point release milestones lined up. These point releases dictate the issues that core team has discussed as priority already. Core team should not focus on issues outside these milestones as that implies either the issue was not discussed as a priority, or the issue isn't a priority.

We follow semantic versioning for our release numbering and our point release milestones follow these guidelines. For example, our next release milestones might be `2.8`, `2.9`, and `3.0`. Releases `2.8` and `2.9` will contain bug fix issues and one backwards compatible feature (this dictates the change in minor verison). Release `3.0` will contain bugfixes and at least one backwards incompatible change.

Point release milestones try to remain static, but can shift upwards on a release that included an unexpected feature addition. Sometimes the resulting PR unexpectedly includes changes that dictate a minor version increment though, according to semantic versioning. In this case, the current milestone is closed, future milestones are increased a minor point if necessary, and the remaining milestone issues are migrated to a new milestone for the next upcoming release number.

Group milestones are blocks of work that will have priority in the future, but aren't included on point releases yet. When the core team does decide to make these milestones a priorty, they will be moved into point release milestones.

Where to contribute

It's best to pick off an issue from our current point release or group milestones, to ensure your pull request gets attention. You can also feel free to contribute on our Cleanup or Refactoring milestones. Though not a development priority, these issues are generally discrete, easier to jump into than feature development, and we especially appreciate outside contributions here as these milestones are not a place the core team can justify spending time in development currently.

1.27.2 Current roadmap

In addition to the point release milestones currently established, our current roadmap priorities also include:

Admin UX https://github.com/rtfd/readthedocs.org/milestone/16

Search Improvements https://github.com/rtfd/readthedocs.org/milestone/23

YAML File Completion https://github.com/rtfd/readthedocs.org/milestone/28

There are also several milestones that are explicitly *not* a priority for the core team:

Cleanup https://github.com/rtfd/readthedocs.org/milestone/10

Refactoring https://github.com/rtfd/readthedocs.org/milestone/34

Core team will not be focusing their time on these milestones in development.

1.28 Read the Docs Team

readthedocs.org is the largest open source documentation hosting service. It's provided as a free service to the open source community, and is worked on by a community of volunteers that we're hoping to expand! We currently serve over 20,000,000 pageviews a month, and we hope to maintain a reliable and stable hosting platform for years to come.

There are three major parts of this work:

Note: You may notice that a number of names appear on multiple teams. This is because we are lacking contributors. So please be bold and contact us, and we'll get you sorted into the right team.

1.28.1 Support Team

Read the Docs has thousands of users who depend on it everyday. Every day at least one of them has an issue that needs to be addressed by a site admin. This might include tasks like:

- Resetting a password
- Asking for a project name to be released
- Troubleshooting build errors

Members

- Eric Holscher (Pacific Time)
- Anthony Johnson (Mountain Time)
- Manuel Kaufmann (Central Time)

- Your Name Here

Feel free to ask any of us if you have questions or want to join!

Joining

The best place to start would be to start addressing some of the issues in our issue tracker. We have our support policies quite well documented in our *Contributing to Read the Docs*. **Be bold**. Start trying to reproduce issues that people have, or talk to them to get more information. After you get the hang of things, we'll happily give you the ability to tag and close issues by joining our Support Team.

1.28.2 Operations Team

readthedocs.org is a service that millions of people depend on each month. As part of operating the site, we maintain a 24/7 on-call rotation. This means that folks have to be available and have their phone in service.

Members

- Eric Holscher (Pacific Time)
- Anthony Johnson (Mountain Time)
- Matt Robenolt (Pacific Time)

- Your Name Here

  Feel free to ask any of us if you have questions or want to join!

Joining

We are always looking for more people to share the on-call responsibility. If you are on-call for your job already, we'd love to piggy back on that duty as well.

You can email us at dev@readthedocs.org if you want to join our operations team. Because of the sensitive nature (API tokens, secret keys, SSL certs, etc.) of the work, we keep a private GitHub repository with the operations code & documentation.

The tools that we use are:

- Salt
- Nagios
- Graphite/Grafana
- Nginx
- Postgres
- Django
- Celery

It's fine if you aren't familiar with all of these things, but are willing to help with part of it!

Please reach out if you want to share the on-call responsibility. It really is an important job, and we'd love to have it be more geographically distributed.

1.28.3 Development Team

Also known as the "Core Team" in other projects. These folks have the ability to commit code to the project.

Members

- Eric Holscher
- Anthony Johnson
- Manuel Kaufmann
- David Fischer

- Your name here

Feel free to ask any of us if you have questions or want to join!

Joining

We try to be pretty flexible with who we allow on the development team. The best path is to send a few pull requests, and follow up to make sure they get merged successfully. You can check out our *Contributing to Read the Docs* to get more information, and find issues that need to be addressed. After that, feel free to ask for a commit bit.

1.29 Google Summer of Code

Note: Thanks for your interest in Read the Docs! We are working hard to update the ideas list now that we are accepted in GSOC. Please give us a little while to work on things, and check back on this page for updates.

Read the Docs is participating in the Google Summer of Code in 2018. This page will contain all the information for students and anyone else interested in helping.

1.29.1 Skills

Incoming students will need the following skills:

- Intermediate Python & Django programming

- Familiarity with Markdown, reStructuredText, or some other plain text markup language
- Familiarity with git, or some other source control
- Ability to set up your own development environment for Read the Docs
- Basic understanding of web technologies (HTML/CSS/JS)
- An interest in documentation and improving open source documentation tools!

  We're happy to help you get up to speed, but the more you are able to demonstrate ability in advance, the more likely we are to choose your application!

1.29.2 Mentors

Currently we have a few folks signed up:

- Eric Holscher
- Manuel Kaufmann
- Anthony Johnson

> **Warning:** Please do not reach out directly to anyone about the Summer of Code. It will **not** increase your chances of being accepted!

1.29.3 Getting Started

The *Installation* doc is probably the best place to get going. It will walk you through getting a basic environment for Read the Docs setup.

Then you can look through our *Contributing to Read the Docs* doc for information on how to get started contributing to RTD.

People who have a history of submitting pull requests will be prioritized in our Summer of Code selection process.

1.29.4 Want to get involved?

If you're interested in participating in GSoC as a student, you can apply during the normal process provided by Google. We are currently overwhelmed with interest, so we are not able to respond individually to each person who is interested.

1.29.5 Project Ideas

We have written our some loose ideas for projects to work on here. We are also open to any other ideas that students might have.

These projects are sorted by priority. We will consider the priority on our roadmap as a factor, along with the skill of the student, in our selection process.

Refactor & improve our search code

Currently we're using a homegrown library for Elastic Search. There is a new elasticsearch-dsl library that we should be using. This project will include:

- Improving our search indexing
- Refactoring how we "model" our search data to use elasticsearch-dsl Models
- Add additional search data into our indexes, like the programming languages, type of document (tutorial, api, etc.) and other data for users to filter by
- (Optional) Improve the UX of the search for users in various ways

Finish YAML config

Currently we have a basic *Read the Docs YAML Config* for Read the Docs. It's still considered beta, and there are a number of features that it doesn't support. We need to support everying users can currently do from our website dashboard inside the YAML file, and then plan a smooth transition path from the database UI to the YAML file.

This is a *large* project and will likely require a good deal of understanding of both Python as well as web technologies. We have a starting list of issues put together, but there will be much more work.

API V3

We currently have a "v2" API that isn't well documented and doesn't allow users to write to it. We want to continue using Django REST Framework for this, but rethink how we're presenting our information to our users.

Currently we're showing everything as simple "models", and we want to start exposing "methods" on our data, similar to GitHub.

This is a large project and should only be done by someone who has done some basic API design previously.

Improve Translation Workflow

Currently we have our documentation & website translated on Transifex, but we don't have a management process for it. This means that translations will often sit for months before making it back into the site and being available to users.

This project would include putting together a workflow for translations:

- Communicate with existing translators and see what needs they have
- Help formalize the process that we have around Transifex to make it easier to contribute to
- Improve our tooling so that integrating new translations is easier

Support for additional build steps for linting & testing

Currently we only build documentation on Read the Docs, but we'd also like to add additional build steps that lets users perform more actions. This would likely take the form of wrapping some of the existing Sphinx builders, and giving folks a nice way to use them inside Read the Docs.

It would be great to have wrappers for the following as a start:

- Link Check (http://www.sphinx-doc.org/en/stable/builders.html#sphinx.builders.linkcheck.CheckExternalLinksBuilder)
- Spell Check (https://pypi.python.org/pypi/sphinxcontrib-spelling/)
- Doctest (http://www.sphinx-doc.org/en/stable/ext/doctest.html#module-sphinx.ext.doctest)
- Coverage (http://www.sphinx-doc.org/en/stable/ext/coverage.html#module-sphinx.ext.coverage)

The goal would also be to make it quite easy for users to contribute third party build steps for Read the Docs, so that other useful parts of the Sphinx ecosystem could be tightly integrated with Read the Docs.

Collections of Projects

This project involves building a user interface for groups of projects in Read the Docs (`Collections`). Users would be allowed to create, publish, and search a `Collection` of projects that they care about. We would also allow for automatic creation of `Collections` based on a project's `setup.py` or `requirements.txt`.

Once a user has a `Collection`, we would allow them to do a few sets of actions on them:

- Search across all the projects in the `Collection` with one search dialog
- Download all the project's documentation (PDF, HTMLZip, Epub) for offline viewing
- Build a landing page for the collection that lists out all the projects, and could even have a user-editable description, similar to our project listing page.

There is likely other ideas that could be done with `Collections` over time.

Integrated Redirects

Right now it's hard for users to rename files. We support redirects, but don't create them automatically on file rename, and our redirect code is brittle.

We should rebuild how we handle redirects across a number of cases:

- Detecting a file change in git/hg/svn and automatically creating a redirect
- Support redirecting an entire domain to another place
- Support redirecting versions

There will also be a good number of things that spawn from this, including version aliases and other related concepts, if this task doesn't take the whole summer.

Additional Ideas

We have some medium sized projects sketched out in our issue tracker with the tag *Feature Overview*. Looking through these issues is a good place to start. You might also look through our milestones on GitHub, which provide outlines on the larger tasks that we're hoping to accomplish.

1.29.6 Thanks

This page was heavily inspired by Mailman's similar GSOC page. Thanks for the inspiration.

1.30 Code of Conduct

Like the technical community as a whole, the Read the Docs team and community is made up of a mixture of pro-

fessionals and volunteers from all over the world, working on every aspect of the mission - including mentorship, teaching, and connecting people.

Diversity is one of our huge strengths, but it can also lead to communication issues and unhappiness. To that end, we have a few ground rules that we ask people to adhere to. This code applies equally to founders, mentors and those seeking help and guidance.

This isn't an exhaustive list of things that you can't do. Rather, take it in the spirit in which it's intended - a guide to make it easier to enrich all of us and the technical communities in which we participate.

This code of conduct applies to all spaces managed by the Read the Docs project. This includes IRC, the mailing lists, the issue tracker, and any other forums created by the project team which the community uses for communication. In addition, violations of this code outside these spaces may affect a person's ability to participate within them.

If you believe someone is violating the code of conduct, we ask that you report it by emailing dev@readthedocs.org.

- **Be friendly and patient.**
- **Be welcoming.** We strive to be a community that welcomes and supports people of all backgrounds and identities. This includes, but is not limited to members of any race, ethnicity, culture, national origin, colour, immigration status, social and economic class, educational level, sex, sexual orientation, gender identity and expression, age, size, family status, political belief, religion, and mental and physical ability.
- **Be considerate.** Your work will be used by other people, and you in turn will depend on the work of others. Any decision you take will affect users and colleagues, and you should take those consequences into account when making decisions. Remember that we're a world-wide community, so you might not be communicating in someone else's primary language.
- **Be respectful.** Not all of us will agree all the time, but disagreement is no excuse for poor behavior and poor manners. We might all experience some frustration now and then, but we cannot allow that frustration to turn into a personal attack. It's important to remember that a community where people feel uncomfortable or threatened is not a productive one. Members of the Read the Docs community should be respectful when dealing with other members as well as with people outside the Read the Docs community.
- **Be careful in the words that you choose.** We are a community of professionals, and we conduct ourselves professionally. Be kind to others. Do not insult or put down other

participants. Harassment and other exclusionary behavior aren't acceptable. This includes, but is not limited to:

- Violent threats or language directed against another person.
- Discriminatory jokes and language.
- Posting sexually explicit or violent material.
- Posting (or threatening to post) other people's personally identifying information ("doxing").
- Personal insults, especially those using racist or sexist terms.
- Unwelcome sexual attention.
- Advocating for, or encouraging, any of the above behavior.
- Repeated harassment of others. In general, if someone asks you to stop, then stop.

- **When we disagree, try to understand why.** Disagreements, both social and technical, happen all the time and Read the Docs is no exception. It is important that we resolve disagreements and differing views constructively. Remember that we're different. The strength of Read the Docs comes from its varied community, people from a wide range of backgrounds. Different people have different perspectives on issues. Being unable to understand why someone holds a viewpoint doesn't mean that they're wrong. Don't forget that it is human to err and blaming each other doesn't get us anywhere. Instead, focus on helping to resolve issues and learning from mistakes.

  Original text courtesy of the Speak Up! project. This version was adopted from the Django Code of Conduct.

1.31 Privacy Policy

Effective date: **May 30, 2018**

Welcome to Read the Docs. At Read the Docs, we believe in protecting the privacy of our users, authors, and readers.

1.31.1 The short version

We collect your information only with your consent; we only collect the minimum amount of personal information that is necessary to fulfill the purpose of your interaction with us; we don't sell it to third parties; and we only use it as this Privacy Policy describes.

Of course, the short version doesn't tell you everything, so please read on for more details!

1.31.2 Our services

Read the Docs is made up of:

readthedocs.org This is a website aimed at documentation authors writing and building software documentation. This Privacy Policy applies to this site in full without reservation.

readthedocs.com This website is a commercial hosted offering for hosting private documentation for corporate clients. It is governed by this privacy policy but also separate terms.

readthedocs.io, readthedocs-hosted.com, and other domains ("Documentation Sites") These websites are where Read the Docs hosts documentation on behalf of documentation authors. A best effort is made to apply this Privacy Policy to these sites but the documentation may contain content and files created by documentation authors.

1.31.3 What information Read the Docs collects and why

Information from website browsers

If you're **just browsing the website**, we collect the same basic information that most websites collect. We use common internet technologies, such as cookies and web server logs. We collect this basic information from everybody, whether they have an account or not.

The information we collect about all visitors to our website includes:

- the visitor's browser type
- language preference
- referring site
- the date and time of each visitor request

We also collect potentially personally-identifying information like Internet Protocol (IP) addresses.

Why do we collect this?

We collect this information to better understand how our website visitors use Read the Docs, and to monitor and protect the security of the website.

Information from users with accounts

If you **create an account**, we require some basic information at the time of account creation. You will create

your own user name and password, and we will ask you for a valid email account. You also have the option to give us more information if you want to, and this may include "User Personal Information."

"User Personal Information" is any information about one of our users which could, alone or together with other information, personally identify him or her. Information such as a user name and password, an email address, a real name, and a photograph are examples of "User Personal Information."

User Personal Information does not include aggregated, non-personally identifying information. We may use aggregated, non-personally identifying information to operate, improve, and optimize our website and service.

Why do we collect this?

- We need your User Personal Information to create your account, and to provide the services you request.
- We use your User Personal Information, specifically your user name, to identify you on Read the Docs.
- We use it to fill out your profile and share that profile with other users.
- We will use your email address to communicate with you but it is not shared publicly.
- We limit our use of your User Personal Information to the purposes listed in this Privacy Statement. If we need to use your User Personal Information for other purposes, we will ask your permission first. You can always see what information we have in your user account.

1.31.4 What information Read the Docs does not collect

We do not intentionally collect **sensitive personal information**, such as social security numbers, genetic data, health information, or religious information.

Documentation Sites hosted on Read the Docs are public, anyone (including us) may view their contents. If you have included private or sensitive information in your Documentation Site, such as email addresses, that information may be indexed by search engines or used by third parties.

If you're a **child under the age of 13**, you may not have an account on Read the Docs. Read the Docs does not knowingly collect information from or direct any of our content specifically to children under 13. If we learn or have reason to suspect that you are a user who is under the age of 13, we will unfortunately have to close your account.

We don't want to discourage you from writing software documentation, but those are the rules.

1.31.5 How we share the information we collect

We **do not** share, sell, rent, or trade User Personal Information with third parties for their commercial purposes.

We do not disclose User Personal Information outside Read the Docs, except in the situations listed in this section or in the section below on compelled disclosure.

We **do** share certain aggregated, non-personally identifying information with others about how our users, collectively, use Read the Docs. For example, we may compile statistics on the prevalence of different types of documentation across Read the Docs for a blog post or popularity of programming languages for advertising partners.

We **do** host advertising on Documentation Sites. This advertising is first-party advertising hosted by Read the Docs. We **do not** run any code from advertisers and all ad images are hosted on Read the Docs' servers. For more details, see our document on *Advertising Details*.

We may use User Personal Information with your permission, so we can perform services you have requested. For example, if you request service on commercially hosted docs, we will ask your permission to sync your private repositories.

We may share User Personal Information with a limited number of third party vendors who process it on our behalf to provide or improve our service, and who have agreed to privacy restrictions similar to our own Privacy Statement. For more details, see our next section on *third parties*.

Third party vendors

As we mentioned, we may share some information with third party vendors or it may be collected by them on our behalf. The information collected and stored by third parties is subject to their policies and practices. This list will be updated from time to time and we encourage you to check back periodically.

Payment processing

Should you choose to become a Supporter, purchase a Gold Membership, or become a subscriber to Read the Docs' commercial hosting product, your payment information and details will be processed by Stripe. Read the Docs does not store your payment information.

Site monitoring

Read the Docs uses Sentry and New Relic to diagnose errors and improve the performance of our site. Both companies take part in the EU-US Privacy Shield framework. We aim to minimize the amount of personal information shared, but the information may include your IP address.

Analytics

We go into detail on analytics in a *separate section specific to analytics*.

Support Desk

Read the Docs uses Intercom to manage support requests for documentation hosted through our commercial hosting on readthedocs.com. If you request support – typically via email – Intercom may process your contact information.

Email newsletter

If you sign up for the Read the Docs email newsletter, your email address and name will be stored by Mailchimp. This newsletter is separate from creating a Read the Docs account and signing up for Read the Docs does not opt you in for the newsletter.

You can manage your email subscription including unsubscribing and deleting your records with Mailchimp. There is a link to do so in the footer of any newsletter you receive from us.

Public Information on Read the Docs

Most of Read the Docs is public-facing including user names, project names, and Documentation Sites. If your content is public-facing, third parties may access it. We do not sell that content; it is yours.

1.31.6 Our use of cookies and tracking

Do Not Track

Read the Docs supports Do Not Track (DNT) and respects users' tracking preferences. Specifically, we support the W3C's tracking preference expression and the EFF's DNT Policy.

For Read the Docs, this means:

- We **do not** do behavioral ad targeting regardless of your DNT preference.
- When DNT is enabled, both logged-in and logged-out users are considered opted-out of *analytics*.
- Regardless of DNT preference, our logs that contain IP addresses and user agent strings are deleted after 10 days unless a DNT exception applies.
- Our full DNT policy is available here.

  Our DNT policy applies without reservation to readthedocs.org and readthedocs.com. A best effort is made to apply this to Documentation Sites, but we do not have complete control over the contents of these sites.

  For more details about DNT, visit All About Do Not Track.

Important: Due to the nature of our environment where documentation is built as necessary, the DNT analytics opt-out for Documentation Sites only applies for those sites generated after May 1, 2018.

Cookies

Read the Docs uses cookies to make interactions with our service easy and meaningful. We use cookies to keep you logged in, remember your preferences, and provide information for future development of Read the Docs.

A cookie is a small piece of text that our web server stores on your computer or mobile device, which your browser sends to us when you return to our site. Cookies do not necessarily identify you if you are merely visiting Read the Docs; however, a cookie may store a unique identifier for each logged in user. The cookies Read the Docs sets are essential for the operation of the website, or are used for performance or functionality. By using our website, you agree that we can place these types of cookies on your computer or device. If you disable your browser or device's ability to accept cookies, you will not be able to log in to Read the Docs.

Google Analytics

We use Google Analytics as a third party tracking service, but we don't use it to track you individually or collect your User Personal Information. We use Google Analytics to collect information about how our website performs and how our users, in general, navigate through and use Read the Docs. This helps us evaluate our users' use of Read the Docs; compile statistical reports on activity; and improve our content and website performance.

Google Analytics gathers certain simple, non-personally identifying information over time, such as your IP address,

browser type, internet service provider, referring and exit pages, time stamp, and similar data about your use of Read the Docs. We do not link this information to any of your personal information such as your user name.

Read the Docs will not, nor will we allow any third party to, use the Google Analytics tool to track our users individually; collect any User Personal Information other than IP address; or correlate your IP address with your identity. Google provides further information about its own privacy practices and offers a browser add-on to opt out of Google Analytics tracking. You may also opt-out of analytics on Read the Docs by enabling *Do Not Track*.

1.31.7 How Read the Docs secures your information

Read the Docs takes all measures reasonably necessary to protect User Personal Information from unauthorized access, alteration, or destruction; maintain data accuracy; and help ensure the appropriate use of User Personal Information. We follow generally accepted industry standards to protect the personal information submitted to us, both during transmission and once we receive it.

No method of transmission, or method of electronic storage, is 100% secure. Therefore, we cannot guarantee its absolute security.

1.31.8 Read the Docs' global privacy practices

Information that we collect will be stored and processed in the United States in accordance with this Privacy Policy. However, we understand that we have users from different countries and regions with different privacy expectations, and we try to meet those needs.

We provide the same standard of privacy protection to all our users around the world, regardless of their country of origin or location, Additionally, we require that if our vendors or affiliates have access to User Personal Information, they must comply with our privacy policies and with applicable data privacy laws.

In particular:

- Read the Docs provides clear methods of unambiguous, informed consent at the time of data collection, when we do collect your personal data.
- We collect only the minimum amount of personal data necessary, unless you choose to provide more. We encourage you to only give us the amount of data you are comfortable sharing.

- We offer you simple methods of accessing, correcting, or deleting the data we have collected.
- We also provide our users a method of recourse and enforcement.

1.31.9 Resolving Complaints

If you have concerns about the way Read the Docs is handling your User Personal Information, please let us know immediately by emailing us at privacy@readthedocs.org.

1.31.10 How we respond to compelled disclosure

Read the Docs may disclose personally-identifying information or other information we collect about you to law enforcement in response to a valid subpoena, court order, warrant, or similar government order, or when we believe in good faith that disclosure is reasonably necessary to protect our property or rights, or those of third parties or the public at large.

In complying with court orders and similar legal processes, Read the Docs strives for transparency. When permitted, we will make a reasonable effort to notify users of any disclosure of their information, unless we are prohibited by law or court order from doing so, or in rare, exigent circumstances.

1.31.11 How you can access and control the information we collect

If you're already a Read the Docs user, you may access, update, alter, or delete your basic user profile information by editing your user account.

Data retention and deletion

Read the Docs will retain User Personal Information for as long as your account is active or as needed to provide you services.

We may retain certain User Personal Information indefinitely, unless you delete it or request its deletion. For example, we don't automatically delete inactive user accounts, so unless you choose to delete your account, we will retain your account information indefinitely.

If you would like to delete your User Personal Information, you may do so in your user account. We will retain and use your information as necessary to comply with our

legal obligations, resolve disputes, and enforce our agreements, but barring legal requirements, we will delete your full profile.

Our web server logs for readthedocs.org, readthedocs.com, and Documentation Sites are deleted after 10 days barring legal obligations.

1.31.12 Changes to our Privacy Policy

We reserve the right to revise this Privacy Policy at any time. If we change this Privacy Policy in the future, we will post the revised Privacy Policy and update the "Effective Date," above, to reflect the date of the changes.

1.31.13 Contacting Read the Docs

Questions regarding Read the Docs' Privacy Policy or information practices should be directed to privacy@readthedocs.org.

1.32 Advertising

Advertising is the single largest source of funding for Read the Docs. It allows us to:

- Serve over **35 million pages** of documentation per month
- Serve over **40 TB** of documentation per month
- Host over **80,000 open source projects** and support over **100,000 users**
- Pay a *small team* of dedicated full-time staff

Many advertising models involve tracking users around the internet, selling their data, and privacy intrusion in general. Instead of doing that, we built an *Ethical Advertising* model that respects user privacy.

We recognize that advertising is not for everyone. You may *opt out of paid advertising* – you will still see *community ads* – or you can go ad-free by becoming a Gold Member or a Supporter of Read the Docs.

1.32.1 Ethical Ads

Read the Docs is a large, free web service. There is one proven business model to support this kind of site: **Advertising**. We are building the advertising model we want to exist, and we're calling it **Ethical Ads**.

Ethical Ads respect users while providing value to advertisers. We don't track you, sell your data, or anything

else. We simply show ads to users, based on the content of the pages you look at. We also give 10% of our ad space to *community projects*, as our way of saying thanks to the open source community.

We talk a bit below about *our worldview on advertising*, if you want to know more.

Important: Are you a marketer? Learn more about how you can connect with the millions of developers who Read the Docs each month.

Feedback

We're a community, and we value your feedback. If you ever want to reach out about this effort, feel free to shoot us an email.

You can *opt out* of having paid ads on your projects, or seeing paid ads if you want. You will still see *community ads*, which we run for free that promote community projects.

Our Worldview

We're building the advertising model we want to exist:

- We don't track you
- We don't sell your data
- We host everything ourselves, no third-party scripts or images

We're doing newspaper advertising, on the internet. For a hundred years, newspapers put an ad on the page, some folks would see it, and advertisers would pay for this. This is our model.

So much ad tech has been built to track users. Following them across the web, from site to site, showing the same ads and gathering data about them. Then retailers sell your purchase data to try and attribute sales to advertising. Now there is an industry in doing fake ad clicks and other scams, which leads the ad industry to track you even more intrusively to know more about you. The current advertising industry is in a vicious downward spiral.

As developers, we understand the massive downsides of the current advertising industry. This includes malware, slow site performance, and huge databases of your personal data being sold to the highest bidder.

The trend in advertising is to have larger and larger ads. They should run before your content, they should take over the page, the bigger, weirder, or flashier the better.

We opt out

- We don't store personal information about you.
- We only keep track of views and clicks.
- We don't build a profile of your personality to sell ads against.
- We only show high quality ads from companies that are of interest to developers.

  We are running a single, small, unobtrusive ad on documentation pages. The products should be interesting to you. The ads won't flash or move.

  We run the ads we want to have on our site, in a way that makes us feel good.

Additional details

- We have additional documentation on the *technical details of our advertising* including our Do Not Track policy and our use of analytics.
- We have an advertising FAQ written for advertisers.
- We have gone into more detail about our views in our blog post about this topic.
- Eric Holscher, one of our co-founders talks a bit more about funding open source this way on his blog.

Join us

We're building the advertising model we want to exist. We hope that others will join us in this mission:

- **If you're a developer**, talk to your marketing folks about using advertising that respects your privacy.
- **If you're a marketer**, vote with your dollars and support us in building the ad model we want to exist. Get more information on what we offer.

Community Ads

There are a large number of projects, conferences, and initiatives that we care about in the software and open source ecosystems. A large number of them operate like we did in the past, with almost no income. Our new Community Ads program will highlight some of these projects each month.

There are a few qualifications for our Community Ads program:

- Your organization and the linked site should not be trying to entice visitors to buy a product or service. We make an exception for conferences around open source projects if they are run not for profit and soliciting donations for open source projects.
- A software project should have an OSI approved license.
- We will not run a community ad for an organization tied to one of our paid advertisers.

  We'll show 10% of our ad inventory each month to support initiatives that we care about. Send us an email to be considered for our Community Ads program.

Opting Out

We have added multiple ways to opt out of the advertising on Read the Docs.

Users can go ad-free by becoming a Gold Member or a Supporter.

Users can opt out of seeing paid advertisements on documentation pages:

- Go to the drop down user menu in the top right of the Read the Docs dashboard and clicking **Settings** (https://readthedocs.org/accounts/edit/).
- On the **Advertising** tab, you can deselect **See paid advertising**.

Project owners can also opt out of paid advertisements for their projects. You can change these options:

- Click on your **Project** page (`/projects/<slug>/`)
- Click the **Admin** dashboard link
- Choose the **Advertising** submenu on the left side
- Change your advertising settings

Project opt out options include:

- Supporting us financially by becoming a Gold Member. This will disable all ads from showing on your project's documentation.
- Supporting us with your time by contributing to the project.
- Moving to our paid product over at readthedocs.com.
- Opting out without doing any of the above. This will make us a little sad, but we understand not everyone has the means to contribute back.

1.32.2 Advertising Details

Read the Docs largely funds our operations and development through advertising. However, we aren't willing to compromise our values, document authors, or site visitors simply to make a bit more money. That's why we created our *ethical advertising* initiative.

We get a lot of inquiries about our approach to advertising which range from questions about our practices to requests to partner. The goal of this document is to shed light on the advertising industry, exactly what we do for advertising, and how what we do is different. If you have questions or comments, send us an email or open an issue on GitHub.

Other ad networks' targeting

Some ad networks build a database of user data in order to predict the types of ads that are likely to be clicked. In the advertising industry, this is called *behavioral targeting*. This can include data such as:

- sites a user has visited
- a user's search history
- ads, pages, or stories a user has clicked on in the past
- demographic information such as age, gender, or income level

Typically, getting a user's page visit history is accomplished by the use of trackers (sometimes called beacons or pixels). For example, if a site uses a tracker from an ad network and a user visits that site, the site can now target future advertising to that user – a known past visitor – with that network. This is called *retargeting*.

Other ad predictions are made by grouping similar users together based on user data using machine learning. Frequently this involves an advertiser uploading personal data on users (often past customers of the advertiser) to an ad network and telling the network to target similar users. The idea is that two users with similar demographic information and similar interests would like the same products. In ad tech, this is known as *lookalike audiences* or *similar audiences*.

Understandably, many people have concerns about these targeting techniques. The modern advertising industry has built enormous value by centralizing massive amounts of data on as many people as possible.

Our targeting details

Read the Docs doesn't use the above techniques. Instead, we target based solely upon:

- Details of the page where the advertisement is shown including:
  - The name, keywords, or programming language associated with the project being viewed
  - Content of the page (eg. H1, title, theme, etc.)
  - Whether the page is being viewed from a mobile device
- General geography
  - We allow advertisers to target ads to a list of countries or to exclude countries from their advertising. For ads targeting the USA, we also support targeting by state or by metro area (DMA specifically).
  - We geolocate a user's IP address to a country when a request is made.

  Read the Docs uses GeoLite2 data created by MaxMind.

Where ads are shown

We can place ads in:

- the sidebar navigation
- the footer of the page
- on search result pages
- a small footer fixed to the bottom of the viewport (mobile-only)
- on 404 pages (rare)

We show no more than one ad per page so you will never see both a sidebar ad and a footer ad on the same page.

Themes

Currently we show ads on:

- Sphinx theme provided by Read the Docs
- MkDocs theme provided by Read the Docs (beginning in May 2018)
- Alabaster Sphinx theme

This list will expand as we strive to put advertising on a larger percentage of the documentation served by us. However, we always give advance notice in our issue tracker and via email about showing ads where none were shown before.

Do Not Track Policy

Read the Docs supports Do Not Track (DNT) and respects users' tracking preferences. For more details, see the *Do Not Track section* of our privacy policy.

Analytics

Analytics are a sensitive enough issue that they require their own section. In the spirit of full transparency, Read the Docs uses Google Analytics (GA). We go into a bit of detail on our use of GA in our *Privacy Policy*.

GA is a contentious issue inside Read the Docs and in our community. Some users are very sensitive and privacy conscious to usage of GA. Some authors want their own analytics on their docs to see the usage their docs get. The developers at Read the Docs understand that different users have different priorities and we try to respect the different viewpoints as much as possible while also accomplishing our own goals.

We have taken steps to address some of the privacy concerns surrounding GA. These steps apply both to analytics collected by Read the Docs and when *authors enable analytics on their docs*.

- Users can opt-out of analytics by using the Do Not Track feature of their browser.
- Read the Docs instructs Google to anonymize IP addresses sent to them.
- The cookies set by GA expire in 30 days rather than the default 2 years.

Why we use analytics

Advertisers ask us questions that are easily answered with an analytics solution like "how many users do you have in Switzerland browsing Python docs?". We need to be able to easily get this data. We also use data from GA for some development decisions such as what browsers to support (or not) or how much usage a particular page or feature gets.

Alternatives

We are always exploring our options with respect to analytics. There are alternatives but none of them are without downsides. Some alternatives are:

- Run a different cloud analytics solution from a provider other than Google (eg. Parse.ly, Matomo Cloud, Adobe

Analytics). We priced a couple of these out based on our load and they are very expensive. They also just substitute one problem of data sharing with another.

- Send data to GA (or another cloud analytics provider) on the server side and strip or anonymize personal data such as IPs before sending them. This would be a complex solution and involve additional infrastructure, but it would have many advantages. It would result in a loss of data on "sessions" and new vs. returning visitors which are of limited value to us.
- Run a local JavaScript based analytics solution (eg. Matomo community). This involves additional infrastructure that needs to be always up. Frequently there are very large databases associated with this. Many of these solutions aren't built to handle Read the Docs' load.
- Run a local analytics solution based on web server log parsing. This has the same infrastructure problems as above while also not capturing all the data we want (without additional engineering) like the programming language of the docs being shown or whether the docs are built with Sphinx or something else.

1.32.3 Ad blocking

Ad blockers fulfill a legitimate need to mitigate the significant downsides of advertising from tracking across the internet, security implications of third-party code, and impacting the UX and performance of sites.

At Read the Docs, we specifically didn't want those things. That's why we built the our *Ethical Ad initiative* with only relevant, unobtrusive ads that respect your privacy and don't do creepy behavioral targeting.

Advertising is the single largest source of funding for Read the Docs. To keep our operations sustainable, we ask that you either *allow our Ethical Ads* or *go ad-free*.

Allowing Ethical Ads

If you use AdBlock or AdBlockPlus and you allow acceptable ads or privacy-friendly acceptable ads then you're all set. Advertising on Read the Docs complies with both of these programs.

If you prefer not to allow acceptable ads but would consider allowing ads that benefit open source, please consider subscribing to either the wider **Open Source Ads list** or simply the **Read the Docs Ads list**.

- Setup for AdBlock
- Setup for AdBlockPlus

- Setup for uBlock Origin

Note: Because of the way Read the Docs is structured where docs are hosted on many different domains, adding a normal ad block exception will only allow that single domain not Read the Docs as a whole.

Going ad-free

Users can go completely ad-free by becoming a Gold Member or a Supporter. Thank you for supporting Read the Docs.

Statistics and data

It can be really hard to find good data on ad blocking. In the spirit of transparency, here is the data we have on ad blocking at Read the Docs.

- 32% of Read the Docs users use an ad blocker
- Of those, a little over 50% allow acceptable ads
- Read the Docs users running ad blockers click on ads at about the same rate as those not running an ad blocker.
- Comparing with our server logs, roughly 28% of our hits did not register a Google Analytics (GA) pageview due to an ad blocker, privacy plugin, disabling JavaScript, or another reason.
- Of users who do not block GA, about 6% opt out of analytics on Read the Docs by enabling *Do Not Track*.

1.33 Sponsors of Read the Docs

Running Read the Docs isn't free, and the site wouldn't be where it is today without generous support of our sponsors. Below is a list of all the folks who have helped the site financially, in order of the date they first started supporting us.

1.33.1 Current sponsors

- Microsoft Azure - They cover all of our hosting expenses every month. This is a pretty large sum of money, averaging around $3,000/mo, and we are really grateful to have them as a sponsor.
- Cloudflare - Cloudflare is providing us with an enterprise plan of their SSL for SaaS Providers product that enables us to provide SSL certificates for custom domains.

- You? (Email us at rev@readthedocs.org for more info)

1.33.2 Past sponsors

- Python Software Foundation
- Revsys
- Mozilla Web Dev
- Django Software Foundation
- Lab305
- Twilio
- Rackspace
- Mozilla

1.33.3 Sponsorship Information

As part of increasing sustainability, Read the Docs is testing out promoting sponsors on documentation pages. We have more information about this in our blog post about this effort.

Sponsor Us

Contact us at rev@readthedocs.org for more information on sponsoring Read the Docs.

1.34 Read the Docs Open Source Philosophy

Read the Docs is Open Source software. We have licensed the code base as MIT, which provides almost no restrictions on the use of the code.

However, as a project there are things that we care about more than others. We built Read the Docs to support documentation in the Open Source community. The code is open for people to contribute to, so that they may build features into https://readthedocs.org that they want. We also believe sharing the code openly is a valuable learning tool, especially for demonstrating how to collaborate and maintain an enormous website.

1.34.1 Official Support

The time of the core developers of Read the Docs is limited. We provide official support for the following things:

- Local development on the Python code base
- Usage of https://readthedocs.org for Open Source projects

- Bug fixes in the code base, as it applies to running it on https://readthedocs.org

1.34.2 Unsupported

There are use cases that we don't support, because it doesn't further our goal of promoting documentation in the Open Source Community.

We do not support:

- Specific usage of Sphinx and Mkdocs, that don't affect our hosting
- Custom installations of Read the Docs at your company
- Installation of Read the Docs on other platforms
- Any installation issues outside of the Read the Docs Python Code

1.34.3 Rationale

Read the Docs was founded to improve documentation in the Open Source Community. We fully recognize and allow the code to be used for internal installs at companies, but we will not spend our time supporting it. Our time is limited, and we want to spend it on the mission that we set out to originally support.

If you feel strongly about installing Read the Docs internal to a company, we will happily link to third party resources on this topic. Please open an issue with a proposal if you want to take on this task.

1.35 The Story of Read the Docs

Documenting projects is hard, hosting them shouldn't be. Read the Docs was created to make hosting documentation simple.

Read the Docs was started with a couple main goals in mind. The first goal was to encourage people to write documentation, by removing the barrier of entry to hosting. The other goal was to create a central platform for people to find documentation. Having a shared platform for all documentation allows for innovation at the platform level, allowing work to be done once and benefit everyone.

Documentation matters, but its often overlooked. We think that we can help a documentation culture flourish. Great projects, such as Django and SQLAlchemy, and projects from companies like Mozilla, are already using Read the Docs to serve their documentation to the world.

The site has grown quite a bit over the past year. Our look back at 2013 shows some numbers that show our progress.

The job isn't anywhere near done yet, but it's a great honor to be able to have such an impact already.

We plan to keep building a great experience for people hosting their docs with us, and for users of the documentation that we host.

1.36 Policy for Abandoned Projects

This policy describes the process by which a Read the Docs project name may be changed.

1.36.1 Rationale

Conflict between the current use of the name and a different suggested use of the same name occasionally arise. This document aims to provide general guidelines for solving the most typical cases of such conflicts.

1.36.2 Specification

The main idea behind this policy is that Read the Docs serves the community. Every user is invited to upload content under the Terms of Use, understanding that it is at the sole risk of the user.

While Read the Docs is not a backup service, the core team of Read the Docs does their best to keep that content accessible indefinitely in its published form. However, in certain edge cases the greater community's needs might outweigh the individual's expectation of ownership of a project name.

The use cases covered by this policy are:

Abandoned projects Renaming a project so that the original project name can be used by a different project

Active projects Resolving disputes over a name

1.36.3 Implementation

Reachability

The user of Read the Docs is solely responsible for being reachable by the core team for matters concerning projects that the user owns. In every case where contacting the user is necessary, the core team will try to do so at least three times, using the following means of contact:

- E-mail address on file in the user's profile

- E-mail addresses found in the given project's documentation
- E-mail address on the project's home page

The core team will stop trying to reach the user after six weeks and the user will be considered *unreachable*.

Abandoned projects

A project is considered *abandoned* when ALL of the following are met:

- Owner is unreachable (see *Reachability*)
- The project has no proper documentation being served (no successful builds) or does not have any releases within the past twelve months
- No activity from the owner on the project's home page (or no home page found).

All other projects are considered *active*.

Renaming of an abandoned project

Projects are never renamed solely on the basis of abandonment.

An *abandoned* project can be renamed (by appending `-abandoned` and a uniquifying integer if needed) for purposes of reusing the name when ALL of the following are met:

- The project has been determined *abandoned* by the rules described above
- The candidate is able to demonstrate their own failed attempts to contact the existing owner
- The candidate is able to demonstrate that the project suggested to reuse the name already exists and meets notability requirements
- The candidate is able to demonstrate why a fork under a different name is not an acceptable workaround
- The project has fewer than 100 monthly pageviews
- The core team does not have any additional reservations.

Name conflict resolution for active projects

The core team of Read the Docs are not arbiters in disputes around *active* projects. The core team recommends users to get in touch with each other and solve the issue by respectful communication.

1.36.4 Prior art

The Python Package Index (PyPI) policy for claiming abandoned packages (PEP-0541) heavily influenced this policy.

1.37 DMCA Takedown Policy

These are the guidelines that Read the Docs follows when handling DMCA takedown requests and takedown counter requests. If you are a copyright holder wishing to submit a takedown request, or an author that has been notified of a takedown request, please familiarize yourself with *our process*. You will be asked to confirm that you have reviewed information if you submit a request or counter request.

We aim to keep this entire process as transparent as possible. Our process is modeled after GitHub's DMCA takedown process, which we appreciate for its focus on transparency and fairness. All requests and counter requests will be posted to this page below, in the *Request Archive*. These requests will be redacted to remove all identifying information, except for Read the Docs user and project names.

1.37.1 Takedown Process

Here are the steps the Read the Docs will follow in the takedown request process:

Copyright holder submits a request This request, if valid, will be posted publicly on this page, *down below*. The author affected by the takedown request will be notified with a link to the takedown request.

For more information on submitting a takedown request, see: *Submitting a Request*

Author is contacted The author of the content in question will be asked to make changes to the content specified in the takedown request. The author will have 24 hours to make these changes. The copyright holder will be notified if and when this process begins

Author acknowledges changes have been made The author must notify Read the Docs that changes have been made within 24 hours of receiving a takedown request. If the author does not respond to this request, the default action will be to disable the Read the Docs project and remove any hosted versions

Copyright holder review If the author has made changes, the copyright holder will be notified of these changes. If the changes are sufficient, no further action is required, though

copyright holders are welcome to submit a formal retraction. If the changes are not sufficient, the author's changes can be rejected. If the takedown request requires alteration, a new request must be submitted. If Read the Docs does not receive a review response from the copyright holder within 2 weeks, the default action at this step is to assume the takedown request has been retracted.

Content may be disabled If the author does not respond to a request for changes, or if the copyright holder has rejected the author's changes during the review process, the documentation project in question will be disabled.

Author submits a counter request If the author believes their content was disabled as a result of a mistake, a counter request may be submitted. It would be advised that authors seek legal council before continuing. If the submitted counter request is sufficiently detailed, this counter will also be added to *this page*. The copyright holder will be notified, with a link to this counter request.

For more information on submitting a counter request, see: *Submitting a Counter*

Copyright holder may file legal action At this point, if the copyright holder wishes to keep the offending content disabled, the copyright holder must file for legal action ordering the author refrain from infringing activities on Read the Docs. The copyright holder will have 2 weeks to supply Read the Docs with a copy of a valid legal complaint against the author. The default action here, if the copyright holder does not respond to this request, is to re-enable the author's project.

Submitting a Request

Your request must:

Acknowledge this process You must first acknowledge you are familiar with our DMCA takedown request process. If you do not acknowledge that you are familiar with our process, you will be instructed to review this information.

Identify the infringing content You should list URLs to each piece of infringing content. If you allege that the entire project is infringing on copyrights you hold, please specify the entire project as infringing.

Identify infringement resolution You will need to specify what a user must do in order to avoid having the rest of their content disabled. Be as specific as possible with this. Specify if this means adding attribution, identify specific files or content that should be removed, or if you allege the entire project is infringing, your should be specific as to why it is infringing.

Include your contact information Include your name, email, physical address, and phone number.

Include your signature This can be a physical or electronic signature.

Please complete this `takedown request template` and send it to: support@readthedocs.com

Submitting a Counter

Your counter request must:

Acknowledge this process You must first acknowledge you are familiar with our DMCA takedown request process. If you do not acknowledge that you are familiar with our process, you will be instructed to review this information.

Identify the infringing content that was removed Specify URLs in the original takedown request that you wish to challenge.

Include your contact information Include your name, email, physical address, and phone number.

Include your signature This can be a physical or electronic signature.

Requests can be submitted to: support@readthedocs.com

1.37.2 Request Archive

Currently, Read the Docs has not received any takedown requests.

1.38 Changelog

1.38.1 Version 2.8.3

Date December 05, 2018

- @nutann3: Update "install Sphinx" URL (#4959)
- @humitos: Pin redis to the current stable and compatible version (#4956)
- @humitos: Properly set LANG environment variables (#4954)
- @humitos: Adapt code to remove and ignore warnings (#4953)
- @stsewd: Shallow git clone (#4939)
- @stsewd: Install latest version of pip (#4938)
- @stsewd: Fix svn update (#4933)
- @ericholscher: Release 2.8.2 (#4931)
- @stsewd: Remove repeated and dead code (#4929)

- @stsewd: Remove deprecated sudo from travis (#4919)
- @dojutsu-user: Validate profile form fields (#4910)
- @davidfischer: Calculate actual ad views (#4885)
- @stsewd: Sync versions when creating/deleting versions (#4876)
- @dojutsu-user: Remove unused project model fields (#4870)
- @humitos: All package updates (#4792)
- @humitos: Support git unicode branches (#4433)

1.38.2 Version 2.8.2

Date November 28, 2018

- @stsewd: Use .exists in queryset (#4927)
- @stsewd: Don't rmtree symlink (#4925)
- @stsewd: Delete tags with same commit (#4915)
- @safwanrahman: Tuning Elasticsearch for search improvements (#4909)
- @edmondchuc: Fixed some typos. (#4906)
- @humitos: Upgrade stripe Python package to the latest version (#4904)
- @humitos: Retry on API failure when connecting from builders (#4902)
- @stsewd: Separate update and checkout steps (#4901)
- @humitos: Expose environment variables from database into build commands (#4894)
- @ericholscher: Use python to expand the cwd instead of environment variables (#4882)
- @humitos: Call Celery worker properly (#4881)
- @dojutsu-user: Disable django.security.DisallowedHost from logging (#4879)
- @dojutsu-user: Remove 'Sphinx Template Changes' From Docs (#4878)
- @ericholscher: Unbreak the admin on ImportedFile by using raw_id_fields (#4874)
- @stsewd: Check if latest exists before updating identifier (#4873)
- @ericholscher: Release 2.8.1 (#4872)
- @dojutsu-user: Update django-guardian settings (#4871)
- @dojutsu-user: Change 'VerisionLockedTimeout' to 'VersionLockedError' in comment. (#4859)
- @stsewd: Hide "edit on" when the version is a tag (#4851)
- @stsewd: Delete untracked tags on fetch (#4811)
- @humitos: Appropiate logging when a LockTimeout for VCS is reached (#4804)
- @stsewd: Remove support for multiple configurations in one file (#4800)
- @stsewd: Pipfile support (schema) (#4782)
- @stsewd: Save config on build model (#4749)
- @invinciblycool: Redirect to build detail post manual build (#4622)
- @davidfischer: Enable timezone support and set timezone to UTC (#4545)

- @chirathr: Webhook notification URL size validation check (#3680)

1.38.3 Version 2.8.1

Date November 06, 2018

- @ericholscher: Fix migration name on modified date migration (#4867)
- @dojutsu-user: Change 'VerisionLockedTimeout' to 'VersionLockedError' in comment. (#4859)
- @stsewd: Fix rtd config file (#4857)
- @ericholscher: Shorten project name to match slug length (#4856)
- @stsewd: Generic message for parser error of config file (#4853)
- @stsewd: Use $HOME as CWD for virtualenv creation (#4852)
- @stsewd: Hide "edit on" when the version is a tag (#4851)
- @ericholscher: Add modified_date to ImportedFile. (#4850)
- @ericholscher: Use raw_id_fields so that the Feature admin loads (#4849)
- @stsewd: Allow to change project's VCS (#4845)
- @benjaoming: Version compare warning text (#4842)
- @dojutsu-user: Make form for adopting project a choice field (#4841)
- @humitos: Do not send notification on VersionLockedError (#4839)
- @stsewd: Start testing config v2 on our project (#4838)
- @ericholscher: Add all migrations that are missing from model changes (#4837)
- @ericholscher: Add docstring to DrfJsonSerializer so we know why it's there (#4836)
- @ericholscher: Show the project's slug in the dashboard (#4834)
- @humitos: Avoid infinite redirection (#4833)
- @ericholscher: Allow filtering builds by commit. (#4831)
- @dojutsu-user: Add 'Branding' under the 'Business Info' section and 'Guidelines' on 'Design Docs' (#4830)
- @davidfischer: Migrate old passwords without "set_unusable_password" (#4829)
- @humitos: Do not import the Celery worker when running the Django app (#4824)
- @damianz5: Fix for jQuery in doc-embed call (#4819)
- @invinciblycool: Add MkDocsYAMLParseError (#4814)
- @stsewd: Delete untracked tags on fetch (#4811)
- @stsewd: Don't activate version on build (#4810)
- @humitos: Feature flag to make `readthedocs` theme default on MkDocs docs (#4802)
- @ericholscher: Allow use of `file://` urls in repos during development. (#4801)
- @ericholscher: Release 2.7.2 (#4796)
- @dojutsu-user: Raise 404 at SubdomainMiddleware if the project does not exist. (#4795)
- @dojutsu-user: Add help_text in the form for adopting a project (#4781)
- @humitos: Add VAT ID field for Gold User (#4776)

- @sriks123: Remove logic around finding config file inside directories (#4755)
- @dojutsu-user: Improve unexpected error message when build fails (#4754)
- @stsewd: Don't build latest on webhook if it is deactivated (#4733)
- @dojutsu-user: Change the way of using login_required decorator (#4723)
- @invinciblycool: Remove unused views and their translations. (#4632)
- @invinciblycool: Redirect to build detail post manual build (#4622)
- @anubhavsinha98: Issue #4551 Changed mock docks to use sphinx (#4569)
- @xrmx: search: mark more strings for translation (#4438)
- @Alig1493: Fix for issue #4092: Remove unused field from Project model (#4431)
- @mashrikt: Remove pytest _describe (#4429)
- @xrmx: static: use modern getJSON callbacks (#4382)
- @jaraco: Script for creating a project (#4370)
- @xrmx: make it easier to use a different default theme (#4278)
- @humitos: Document alternate domains for business site (#4271)
- @xrmx: restapi/client: don't use DRF parser for parsing (#4160)
- @julienmalard: New languages (#3759)
- @stsewd: Improve installation guide (#3631)
- @stsewd: Allow to hide version warning (#3595)
- @Alig1493: [Fixed #872] Filter Builds according to commit (#3544)
- @stsewd: Make slug field a valid DNS label (#3464)

1.38.4 Version 2.8.0

Date October 30, 2018

Major change is an upgrade to Django 1.11.

- @humitos: Cleanup old code (remove old_div) (#4817)
- @humitos: Remove unnecessary migration (#4806)
- @humitos: Feature flag to make `readthedocs` theme default on MkDocs docs (#4802)
- @stsewd: Add codecov badge (#4799)
- @humitos: Pin missing dependency for the MkDocs guide compatibility (#4798)
- @ericholscher: Release 2.7.2 (#4796)
- @humitos: Do not log as error a webhook with an invalid branch name (#4779)
- @ericholscher: Run travis on release branches (#4763)
- @ericholscher: Remove Eric & Anthony from ADMINS & MANAGERS settings (#4762)
- @stsewd: Don't use RequestsContext (#4759)
- @davidfischer: Django 1.11 upgrade (#4750)
- @stsewd: Dropdown to select Advanced Settings (#4710)

- @stsewd: Remove hardcoded constant from config module (#4704)
- @stsewd: Update tastypie (#4325)
- @stsewd: Update to Django 1.10 (#4319)

1.38.5 Version 2.7.2

Date October 23, 2018

- @humitos: Validate the slug generated is valid before importing a project (#4780)
- @humitos: Do not log as error a webhook with an invalid branch name (#4779)
- @ericholscher: Add an index page to our design docs. (#4775)
- @dojutsu-user: Remove /embed API endpoint (#4771)
- @stsewd: Upgrade logs from debug on middleware (#4769)
- @humitos: Link to SSL for Custom Domains fixed (#4766)
- @ericholscher: Remove Eric & Anthony from ADMINS & MANAGERS settings (#4762)
- @humitos: Do not re-raise the exception if the one that we are checking (#4761)
- @humitos: Do not fail when unlinking an non-existing path (#4760)
- @humitos: Allow to extend the DomainForm from outside (#4752)
- @davidfischer: Fixes an OSX issue with the test suite (#4748)
- @humitos: Use Docker time limit for max lock age (#4747)
- @xyNNN: Fixed link of PagerDuty (#4744)
- @davidfischer: Make storage syncers extend from a base class (#4742)
- @ericholscher: Revert "Upgrade theme media to 0.4.2" (#4735)
- @ericholscher: Upgrade theme media to 0.4.2 (#4734)
- @stsewd: Extend install option from config file (v2, schema only) (#4732)
- @stsewd: Remove /cname endpoint (#4731)
- @ericholscher: Fix get_vcs_repo by moving it to the Mixin (#4727)
- @humitos: Guide explaining how to keep compatibility with mkdocs (#4726)
- @ericholscher: Release 2.7.1 (#4725)
- @dojutsu-user: Fix the form for adopting a project (#4721)
- @ericholscher: Remove logging verbosity on syncer failure (#4717)
- @humitos: Lint requirement file for py2 (#4712)
- @davidfischer: Improve the getting started docs (#4676)
- @stsewd: Strict validation in configuration file (v2 only) (#4607)
- @stsewd: Run coverage on travis (#4605)

1.38.6 Version 2.7.1

Date October 04, 2018

- @ericholscher: Revert "Merge pull request #4636 from rtfd/search_upgrade" (#4716)
- @ericholscher: Reduce the logging we do on CNAME 404 (#4715)
- @davidfischer: Minor redirect admin improvements (#4709)
- @humitos: Define the doc_search reverse URL from inside the __init__ on test (#4703)
- @ericholscher: Revert "auto refresh false" (#4701)
- @browniebroke: Remove unused package nilsimsa (#4697)
- @stsewd: Fix broken url on sphinx projects (#4696)
- @safwanrahman: Tuning elasticsearch shard and replica (#4689)
- @ericholscher: Fix bug where we were not indexing Sphinx HTMLDir projects (#4685)
- @ericholscher: Fix the queryset used in chunking (#4683)
- @ericholscher: Fix python 2 syntax for getting first key in search index update (#4682)
- @ericholscher: Release 2.7.0 (#4681)
- @davidfischer: Increase footer ad text size (#4678)
- @davidfischer: Fix broken docs links (#4677)
- @ericholscher: Remove search autosync from tests so local tests work (#4675)
- @stsewd: Refactor tasks into decorators (#4666)
- @stsewd: Clean up logging (#4665)
- @davidfischer: Ad customization docs (#4659)
- @davidfischer: Fix a typo in the privacy policy (#4658)
- @stsewd: Refactor PublicTask into a decorator task (#4656)
- @stsewd: Remove -r option from update_repos command (#4653)
- @davidfischer: Create an explicit ad placement (#4647)
- @agjohnson: Use collectstatic on `media/`, without collecting user files (#4502)
- @stsewd: Implement submodules key from v2 config (#4493)
- @stsewd: Implement mkdocs key from v2 config (#4486)
- @agjohnson: Add docs on our roadmap process (#4469)
- @humitos: Send notifications when generic/unhandled failures (#3864)
- @stsewd: Use relative path for docroot on mkdocs (#3525)

1.38.7 Version 2.7.0

Date September 29, 2018

Reverted, do not use

1.38.8 Version 2.6.6

Date September 25, 2018

- @davidfischer: Fix a markdown test error (#4663)
- @davidfischer: Ad customization docs (#4659)
- @davidfischer: Fix a typo in the privacy policy (#4658)
- @agjohnson: Put search step back into project build task (#4655)
- @davidfischer: Create an explicit ad placement (#4647)
- @stsewd: Fix some typos in docs and code (#4646)
- @stsewd: Downgrade celery (#4644)
- @stsewd: Downgrade django-taggit (#4639)
- @safwanrahman: [Fix #4247] deleting old search code (#4635)
- @stsewd: Add change versions slug to faq (#4633)
- @stsewd: Pin sphinx to a compatible version (#4631)
- @davidfischer: Make ads more obvious that they are ads (#4628)
- @agjohnson: Change mentions of "CNAME" -> custom domain (#4627)
- @invinciblycool: Use validate_dict for more accurate error messages (#4617)
- @safwanrahman: fixing the indexing (#4615)
- @humitos: Update our sponsors to mention Azure (#4614)
- @agjohnson: Add cwd to subprocess calls (#4611)
- @agjohnson: Make restapi URL additions conditional (#4609)
- @agjohnson: Ability to use supervisor from python 2.7 and still run Python 3 (#4606)
- @humitos: Return 404 for inactive versions and allow redirects on them (#4599)
- @davidfischer: Fixes an issue with duplicate gold subscriptions (#4597)
- @davidfischer: Fix ad block nag project issue (#4596)
- @humitos: Run all our tests with Python 3.6 on Travis (#4592)
- @humitos: Sanitize command output when running under DockerBuildEnvironment (#4591)
- @humitos: Force resolver to use PUBLIC_DOMAIN over HTTPS if not Domain.https (#4579)
- @davidfischer: Updates and simplification for mkdocs (#4556)
- @humitos: Docs for hidding "On ..." section from versions menu (#4547)
- @stsewd: Implement sphinx key from v2 config (#4482)
- @safwanrahman: [Fix #4268] Adding Documentation for upgraded Search (#4467)
- @humitos: Upgrade all packages using pur (#4318)
- @humitos: Clean CC sensible data on Gold subscriptions (#4291)
- @stsewd: Update docs to match the new triague guidelines (#4260)
- @xrmx: Make the STABLE and LATEST constants overridable (#4099)
- @stsewd: Use str to get the exception message (#3912)

1.38.9 Version 2.6.5

Date August 29, 2018

- @stsewd: Tests for yaml file regex (#4587)
- @agjohnson: Respect user language when caching homepage (#4585)
- @humitos: Add start and termination to YAML file regex (#4584)
- @safwanrahman: [Fix #4576] Do not delete projects which have multiple users (#4577)

1.38.10 Version 2.6.4

Date August 29, 2018

- @stsewd: Update tests failing on master (#4575)
- @davidfischer: Add a flag to disable docsearch (#4570)
- @stsewd: Fix nested syntax in docs (#4567)
- @stsewd: Fix incorrect reraise (#4566)
- @davidfischer: Add a note about specifying the version of build tools (#4562)
- @davidfischer: Serve badges directly from local filesystem (#4561)
- @humitos: Build JSON artifacts in HTML builder (#4554)
- @humitos: Route task to proper queue (#4553)
- @humitos: Sanitize BuildCommand.output by removing NULL characters (#4552)
- @davidfischer: Fix changelog for 2.6.3 (#4548)
- @ericholscher: Remove hiredis (#4542)
- @davidfischer: Use the STATIC_URL for static files to avoid redirection (#4522)
- @stsewd: Update docs about build process (#4515)
- @StefanoChiodino: Allow for period as a prefix and yaml extension for config file (#4512)
- @AumitLeon: Update information on mkdocs build process (#4508)
- @humitos: Fix Exact Redirect to work properly when using $rest keyword (#4501)
- @humitos: Mark some BuildEnvironmentError exceptions as Warning and do not log them (#4495)
- @xrmx: projects: don't explode trying to update UpdateDocsTaskStep state (#4485)
- @humitos: Note with the developer flow to update our app translations (#4481)
- @humitos: Add `trimmed` to all multilines `blocktrans` tags (#4480)
- @humitos: Example and note with usage of trimmed option in blocktrans (#4479)
- @humitos: Update Transifex resources for our documentation (#4478)
- @humitos: Documentation for Manage Translations (#4470)
- @stsewd: Port https://github.com/rtfd/readthedocs-build/pull/38/ (#4461)
- @stsewd: Match v1 config interface to new one (#4456)
- @humitos: Skip tags that point to blob objects instead of commits (#4442)

- @stsewd: Document python.use_system_site_packages option (#4422)
- @humitos: More tips about how to reduce resources usage (#4419)
- @xrmx: projects: user in ProjectQuerySetBase.for_admin_user is mandatory (#4417)

1.38.11 Version 2.6.3

Date August 18, 2018

Release to Azure!

- @davidfischer: Add Sponsors list to footer (#4424)
- @stsewd: Cache node_modules to speed up CI (#4484)
- @xrmx: templates: mark missing string for translation on project edit (#4518)
- @ericholscher: Performance improvement: cache version listing on the homepage (#4526)
- @agjohnson: Remove mailgun from our dependencies (#4531)
- @davidfischer: Improved ad block detection (#4532)
- @agjohnson: Revert "Remove SelectiveFileSystemFolder finder workaround" (#4533)
- @davidfischer: Slight clarification on turning off ads for a project (#4534)
- @davidfischer: Fix the sponsor image paths (#4535)
- @agjohnson: Update build assets (#4537)

1.38.12 Version 2.6.2

Date August 14, 2018

- @davidfischer: Custom domain clarifications (#4514)
- @trein: Use single quote throughout the file (#4513)
- @davidfischer: Support ads on pallets themes (#4499)
- @davidfischer: Only use HostHeaderSSLAdapter for SSL/HTTPS connections (#4498)
- @keflavich: Very minor English correction (#4497)
- @davidfischer: All static media is run through "collectstatic" (#4489)
- @humitos: Fix reST structure (#4488)
- @nijel: Document expected delay on CNAME change and need for CAA (#4487)
- @davidfischer: Allow enforcing HTTPS for custom domains (#4483)
- @davidfischer: Add some details around community ad qualifications (#4436)
- @davidfischer: Updates to manifest storage (#4430)
- @davidfischer: Update alt domains docs with SSL (#4425)
- @agjohnson: Add SNI support for API HTTPS endpoint (#4423)
- @davidfischer: API v1 cleanup (#4415)
- @davidfischer: Allow filtering versions by active (#4414)
- @mlncn: Fix broken link (#4410)

- @safwanrahman: [Fix #4407] Port Project Search for Elasticsearch 6.x (#4408)
- @davidfischer: Add client ID to Google Analytics requests (#4404)
- @xrmx: projects: fix filtering in projects_tag_detail (#4398)
- @davidfischer: Fix a proxy model bug related to ad-free (#4390)
- @humitos: Release 2.6.1 (#4389)
- @davidfischer: Do not access database from builds to check ad-free (#4387)
- @humitos: Adapt YAML config integration tests (#4385)
- @stsewd: Set full `source_file` path for default configuration (#4379)
- @humitos: Make `get_version` usable from a specified path (#4376)
- @humitos: More tags when logging errors to Sentry (#4375)
- @humitos: Check for 'options' in update_repos command (#4373)
- @safwanrahman: [Fix #4333] Implement asynchronous search reindex functionality using celery (#4368)
- @stsewd: V2 of the configuration file (#4355)
- @davidfischer: Remove the UID from the GA measurement protocol (#4347)
- @humitos: Mount `pip_cache_path` in Docker container (#3556)
- @agjohnson: Show subprojects in search results (#1866)

1.38.13 Version 2.6.1

Date July 17, 2018

- @davidfischer: Do not access database from builds to check ad-free (#4387)
- @humitos: Adapt YAML config integration tests (#4385)
- @stsewd: Set full `source_file` path for default configuration (#4379)
- @humitos: More tags when logging errors to Sentry (#4375)

1.38.14 Version 2.6.0

Date July 16, 2018

- Adds initial support for HTTPS on custom domains
- @stsewd: Revert "projects: serve badge with same protocol as site" (#4353)
- @davidfischer: Do not overwrite sphinx context variables feature (#4349)
- @stsewd: Calrify docs about how rtd select the stable version (#4348)
- @davidfischer: Remove the UID from the GA measurement protocol (#4347)
- @stsewd: Fix error in command (#4345)

- @davidfischer: Improvements for the build/version admin (#4344)
- @safwanrahman: [Fix #4265] Porting frontend docsearch to work with new API (#4340)
- @ktdreyer: fix spelling of "demonstrating" (#4336)
- @davidfischer: Warning about theme context implementation status (#4335)
- @Blendify: Docs: Let Theme Choose Pygments Theme (#4331)
- @davidfischer: Disable the ad block nag for ad-free projects (#4329)
- @safwanrahman: [fix #4265] Port Document search API for Elasticsearch 6.x (#4309)
- @stsewd: Refactor configuration object to class based (#4298)

1.38.15 Version 2.5.3

Date July 05, 2018

- @xrmx: Do less work in querysets (#4322)
- @stsewd: Fix deprecations in management commands (#4321)
- @davidfischer: Add a flag for marking a project ad-free (#4313)
- @davidfischer: Use "npm run lint" from tox (#4312)
- @davidfischer: Fix issues building static assets (#4311)
- @humitos: Use PATHs to call clear_artifacts (#4296)
- @safwanrahman: [Fix #2457] Implement exact match search (#4292)
- @davidfischer: API filtering improvements (#4285)
- @annegentle: Remove self-referencing links for webhooks docs (#4283)
- @safwanrahman: [Fix #2328 #2013] Refresh search index and test for case insensitive search (#4277)
- @xrmx: doc_builder: clarify sphinx backend append_conf docstring (#4276)
- @davidfischer: Add documentation for APIv2 (#4274)
- @humitos: Wrap notifications HTML code into a block (#4273)
- @stsewd: Move config.py from rtd build (#4272)
- @ericholscher: Fix our use of `--use-wheel` in pip. (#4269)
- @agjohnson: Revert "Merge pull request #4206 from FlorianKuckelkorn/fix/pip-breaking-change" (#4261)
- @humitos: Fix triggering a build for a skipped project (#4255)
- @stsewd: Update default sphinx version (#4250)
- @stsewd: Move config module from rtd-build repo (#4242)
- @davidfischer: Allow staying logged in for longer (#4236)

- @safwanrahman: Upgrade Elasticsearch to version 6.x (#4211)
- @humitos: Make tests extensible from corporate site (#4095)
- @stsewd: `stable` version stuck on a specific commit (#3913)

1.38.16 Version 2.5.2

Date June 18, 2018

- @davidfischer: Add a page detailing ad blocking (#4244)
- @xrmx: projects: serve badge with same protocol as site (#4228)
- @FlorianKuckelkorn: Fixed breaking change in pip 10.0.0b1 (2018-03-31) (#4206)
- @StefanoChiodino: Document that readthedocs file can now have yaml extension (#4129)
- @humitos: Downgrade docker to 3.1.3 because of timeouts in EXEC call (#4241)
- @stsewd: Move parser tests from rtd-build repo (#4225)
- @humitos: Handle revoked oauth permissions by the user (#4074)
- @humitos: Allow to hook the initial build from outside (#4033)

1.38.17 Version 2.5.1

Date June 14, 2018

- @stsewd: Add feature to build json with html in the same build (#4229)
- @davidfischer: Prioritize ads based on content (#4224)
- @mostaszewski: #4170 - Link the version in the footer to the changelog (#4217)
- @Jmennius: Add provision_elasticsearch command (#4216)
- @SuriyaaKudoIsc: Use the latest YouTube share URL (#4209)
- @davidfischer: Allow staff to trigger project builds (#4207)
- @davidfischer: Use autosectionlabel in the privacy policy (#4204)
- @davidfischer: These links weren't correct after #3632 (#4203)
- @davidfischer: Release 2.5.0 (#4200)
- @ericholscher: Fix Build: Convert md to rst in docs (#4199)
- @ericholscher: Updates to #3850 to fix merge conflict (#4198)
- @ericholscher: Build on top of #3881 and put docs in custom_installs. (#4196)
- @davidfischer: Increase the max theme version (#4195)
- @ericholscher: Remove maxcdn reqs (#4194)
- @ericholscher: Add missing gitignore item for ES testing (#4193)
- @xrmx: fabfile: update i18n helpers (#4189)
- @xrmx: Update italian locale (#4188)
- @xrmx: locale: update and build the english translation (#4187)
- @humitos: Upgrade celery to avoid AtributeError:async (#4185)

- @stsewd: Prepare code for custo mkdocs.yaml location (#4184)
- @agjohnson: Updates to our triage guidelines (#4180)
- @davidfischer: Server side analytics (#4131)
- @humitos: Upgrade packages with pur (#4124)
- @stsewd: Fix resync remote repos (#4113)
- @stsewd: Add schema for configuration file with yamale (#4084)
- @davidfischer: Ad block nag to urge people to whitelist (#4037)
- @benjaoming: Add Mexican Spanish as a project language (#3588)

1.38.18 Version 2.5.0

Date June 06, 2018

- @ericholscher: Fix Build: Convert md to rst in docs (#4199)
- @ericholscher: Remove maxcdn reqs (#4194)
- @ericholscher: Add missing gitignore item for ES testing (#4193)
- @xrmx: fabfile: update i18n helpers (#4189)
- @xrmx: Update italian locale (#4188)
- @xrmx: locale: update and build the english translation (#4187)
- @safwanrahman: Test for search functionality (#4116)
- @davidfischer: Update mkdocs to the latest (#4041)
- @davidfischer: Ad block nag to urge people to whitelist (#4037)
- @davidfischer: Decouple the theme JS from readthedocs.org (#3968)
- @xrmx: tests: fixup url tests in test_privacy_urls (#3966)
- @fenilgandhi: Add support for different badge styles (#3632)
- @benjaoming: Add Mexican Spanish as a project language (#3588)
- @stsewd: Wrap versions' list to look more consistent (#3445)
- @agjohnson: Move CDN code to external abstraction (#2091)

1.38.19 Version 2.4.0

Date May 31, 2018

- This fixes assets that were generated against old dependencies in 2.3.14
- @agjohnson: Fix issues with search javascript (#4176)
- @stsewd: Use anonymous refs in CHANGELOG (#4173)
- @stsewd: Fix some warnings on docs (#4172)
- @davidfischer: Update the privacy policy date (#4171)

- @davidfischer: Note about state and metro ad targeting (#4169)
- @ericholscher: Add another guide around fixing memory usage. (#4168)
- @stsewd: Download raw build log (#3585)
- @stsewd: Add "edit" and "view docs" buttons to subproject list (#3572)
- @kennethlarsen: Remove outline reset to bring back outline (#3512)

1.38.20 Version 2.3.14

Date May 30, 2018

- @ericholscher: Remove CSS override that doesn't exist. (#4165)
- @davidfischer: Include a DMCA request template (#4164)
- @davidfischer: No CSRF cookie for docs pages (#4153)
- @davidfischer: Small footer rework (#4150)
- @stsewd: Fix prospector dependencies (#4149)
- @ericholscher: Remove deploy directory which is unused. (#4147)
- @stsewd: Use autosectionlabel extension (#4146)
- @davidfischer: Add Intercom to the privacy policy (#4145)
- @humitos: Minimum refactor to decide_if_cors (#4143)
- @stsewd: Ignore migrations from coverage report (#4141)
- @stsewd: 5xx status in old webhooks (#4139)
- @davidfischer: Fix with Lato Bold font (#4138)
- @davidfischer: Release 2.3.13 (#4137)
- @davidfischer: Build static assets (#4136)
- @xrmx: oauth/services: correct error handling in paginate (#4134)
- @xrmx: oauth/services: don't abuse log.exception (#4133)
- @cedk: Use quiet mode to retrieve branches from mercurial (#4114)
- @humitos: Add `has_valid_clone` and `has_valid_webhook` to ProjectAdminSerializer (#4107)
- @stsewd: Put the rtd extension to the beginning of the list (#4054)
- @stsewd: Use gitpython for tags (#4052)
- @davidfischer: Do Not Track support (#4046)
- @stsewd: Set urlconf to None after changing SUBDOMAIN setting (#4032)
- @humitos: Fix /404/ testing page (#3976)
- @xrmx: Fix some tests with postgres (#3958)
- @xrmx: Fixup DJANGO_SETTINGS_SKIP_LOCAL in tests (#3899)

- @xrmx: templates: mark a few more strings for translations (#3869)
- @ze: Make search bar in dashboard have a more clear message. (#3844)
- @varunotelli: Pointed users to Python3.6 (#3817)
- @stsewd: [RDY] Fix tests for environment (#3764)
- @ajatprabha: Ticket #3694: rename owners to maintainers (#3703)
- @SanketDG: Refactor to replace old logging to avoid mangling (#3677)
- @stsewd: Add rstcheck to CI (#3624)
- @techtonik: Update Git on prod (#3615)
- @stsewd: Allow to hide version warning (#3595)
- @cclauss: Modernize Python 2 code to get ready for Python 3 (#3514)
- @stsewd: Consistent version format (#3504)

1.38.21 Version 2.3.13

Date May 23, 2018

- @davidfischer: Build static assets (#4136)
- @stsewd: Don't sync _static dir for search builder (#4120)
- @davidfischer: Use the latest Lato release (#4093)
- @davidfischer: Update Gold Member marketing (#4063)
- @davidfischer: Fix missing fonts (#4060)
- @stsewd: Additional validation when changing the project language (#3790)
- @stsewd: Improve yaml config docs (#3685)

1.38.22 Version 2.3.12

Date May 21, 2018

- @stsewd: Remove Django deprecation warning (#4112)
- @davidfischer: Display feature flags in the admin (#4108)
- @humitos: Set valid clone in project instance inside the version object also (#4105)
- @davidfischer: Use the latest theme version in the default builder (#4096)
- @humitos: Use next field to redirect user when login is done by social (#4083)
- @humitos: Update the `documentation_type` when it's set to 'auto' (#4080)
- @brainwane: Update link to license in philosophy document (#4059)
- @agjohnson: Update local assets for theme to 0.3.1 tag (#4047)
- @stsewd: Fix unbalanced div (#4044)
- @stsewd: Remove haystack from code base (#4039)
- @davidfischer: Subdomains use HTTPS if settings specify (#3987)
- @davidfischer: Draft Privacy Policy (#3978)

- @humitos: Allow import Gitlab repo manually and set a webhook automatically (#3934)
- @davidfischer: Enable ads on the readthedocs mkdocs theme (#3922)
- @bansalnitish: Fixes #2953 - Url resolved with special characters (#3725)
- @Jigar3: Deleted bookmarks app (#3663)

1.38.23 Version 2.3.11

Date May 01, 2018

- @agjohnson: Update local assets for theme to 0.3.1 tag (#4047)
- @stsewd: Fix unbalanced div (#4044)
- @stsewd: Remove haystack from code base (#4039)
- @stsewd: Remove dead code from api v1 (#4038)
- @humitos: Bump sphinx default version to 1.7.4 (#4035)
- @davidfischer: Detail where ads are shown (#4031)
- @ericholscher: Make email verification optional for dev (#4024)
- @davidfischer: Support sign in and sign up with GH/GL/BB (#4022)
- @agjohnson: Remove old varnish purge utility function (#4019)
- @agjohnson: Remove build queue length warning on build list page (#4018)
- @stsewd: Don't check order on assertQuerysetEqual on tests for subprojects (#4016)
- @stsewd: Tests for view docs api response (#4014)
- @davidfischer: MkDocs projects use RTD's analytics privacy improvements (#4013)
- @humitos: Release 2.3.10 (#4009)
- @davidfischer: Remove typekit fonts (#3982)
- @stsewd: Move dynamic-fixture to testing requirements (#3956)
- @stsewd: Fix view docs link (#3882)
- @stsewd: [WIP] Remove comments app (#3802)
- @Jigar3: Deleted bookmarks app (#3663)

1.38.24 Version 2.3.10

Date April 24, 2018

- @humitos: Downgrade docker to 3.1.3 (#4003)

1.38.25 Version 2.3.9

Date April 20, 2018

- @agjohnson: Fix recursion problem more generally (#3989)

1.38.26 Version 2.3.8

Date April 20, 2018

- @agjohnson: Give TaskStep class knowledge of the underlying task (#3983)
- @humitos: Resolve domain when a project is a translation of itself (#3981)

1.38.27 Version 2.3.7

Date April 19, 2018

- @humitos: Fix server_error_500 path on single version (#3975)
- @davidfischer: Fix bookmark app lint failures (#3969)
- @humitos: Use latest setuptools (39.0.1) by default on build process (#3967)
- @ericholscher: Fix exact redirects. (#3965)
- @humitos: Make `resolve_domain` work when a project is subproject of itself (#3962)
- @humitos: Remove django-celery-beat and use the default scheduler (#3959)
- @xrmx: Fix some tests with postgres (#3958)
- @davidfischer: Add advertising details docs (#3955)
- @humitos: Use pur to upgrade python packages (#3953)
- @ze: Make adjustments to Projects page (#3948)
- @davidfischer: Small change to Chinese language names (#3947)
- @agjohnson: Don't share state in build task (#3946)
- @davidfischer: Fixed footer ad width fix (#3944)
- @humitos: Allow extend Translation and Subproject form logic from corporate (#3937)
- @humitos: Resync valid webhook for project manually imported (#3935)
- @humitos: Resync webhooks from Admin (#3933)
- @humitos: Fix attribute order call (#3930)
- @humitos: Mention RTD in the Project URL of the issue template (#3928)
- @davidfischer: Correctly report mkdocs theme name (#3920)
- @xrmx: Fixup DJANGO_SETTINGS_SKIP_LOCAL in tests (#3899)
- @davidfischer: Show an adblock admonition in the dev console (#3894)
- @stsewd: Fix view docs link (#3882)
- @xrmx: templates: mark a few more strings for translations (#3869)
- @ze: Update quickstart from README (#3847)
- @vidartf: Fix page redirect preview (#3811)
- @stsewd: [RDY] Fix requirements file lookup (#3800)
- @aasis21: Documentation for build notifications using webhooks. (#3671)
- @mashrikt: [#2967] Scheduled tasks for cleaning up messages (#3604)

- @stsewd: Show URLS for exact redirect (#3593)
- @marcelstoer: Doc builder template should check for mkdocs_page_input_path before using it (#3536)
- @Code0x58: Document creation of slumber user (#3461)

1.38.28 Version 2.3.6

Date April 05, 2018

- @agjohnson: Drop readthedocs- prefix to submodule (#3916)
- @agjohnson: This fixes two bugs apparent in nesting of translations in subprojects (#3909)
- @humitos: Use new django celery beat scheduler (#3908)
- @humitos: Use a proper default for `docker` attribute on UpdateDocsTask (#3907)
- @davidfischer: Handle errors from publish_parts (#3905)
- @agjohnson: Drop pdbpp from testing requirements (#3904)
- @stsewd: Little improve on sync_versions (#3902)
- @humitos: Save Docker image data in JSON file only for DockerBuildEnvironment (#3897)
- @davidfischer: Single analytics file for all builders (#3896)
- @humitos: Organize logging levels (#3893)

1.38.29 Version 2.3.5

Date April 05, 2018

- @agjohnson: Drop pdbpp from testing requirements (#3904)
- @agjohnson: Resolve subproject correctly in the case of single version (#3901)
- @davidfischer: Fixed footer ads again (#3895)
- @davidfischer: Fix an Alabaster ad positioning issue (#3889)
- @humitos: Save Docker image hash in RTD environment.json file (#3880)
- @agjohnson: Add ref links for easier intersphinx on yaml config page (#3877)
- @rajujha373: Typo correction in docs/features.rst (#3872)
- @gaborbernat: add description for tox tasks (#3868)
- @davidfischer: Another CORS hotfix for the sustainability API (#3862)
- @agjohnson: Fix up some of the logic around repo and submodule URLs (#3860)
- @davidfischer: Fix linting errors in tests (#3855)
- @agjohnson: Use gitpython to find a commit reference (#3843)
- @davidfischer: Remove pinned CSS Select version (#3813)
- @davidfischer: Use JSONP for sustainability API (#3789)
- @rajujha373: #3718: Added date to changelog (#3788)
- @xrmx: tests: mock test_conf_file_not_found filesystem access (#3740)

1.38.30 Version 2.3.4

- Release for static assets

1.38.31 Version 2.3.3

- @davidfischer: Fix linting errors in tests (#3855)
- @humitos: Fix linting issues (#3838)
- @humitos: Update instance and model when `record_as_success` (#3831)
- @ericholscher: Reorder GSOC projects, and note priority order (#3823)
- @agjohnson: Add temporary method for skipping submodule checkout (#3821)
- @davidfischer: Remove pinned CSS Select version (#3813)
- @humitos: Use readthedocs-common to share linting files accross different repos (#3808)
- @davidfischer: Use JSONP for sustainability API (#3789)
- @humitos: Define useful celery beat task for development (#3762)
- @humitos: Auto-generate conf.py compatible with Py2 and Py3 (#3745)
- @humitos: Task to remove orphan symlinks (#3543)
- @stsewd: Fix regex for public bitbucket repo (#3533)
- @humitos: Documentation for RTD context sent to the Sphinx theme (#3490)
- @stsewd: Show link to docs on a build (#3446)

1.38.32 Version 2.3.2

This version adds a hotfix branch that adds model validation to the repository URL to ensure strange URL patterns can't be used.

1.38.33 Version 2.3.1

- @humitos: Update instance and model when `record_as_success` (#3831)
- @agjohnson: Bump docker -> 3.1.3 (#3828)
- @Doug-AWS: Pip install note for Windows (#3827)
- @himanshutejwani12: Update index.rst (#3824)
- @ericholscher: Reorder GSOC projects, and note priority order (#3823)
- @agjohnson: Autolint cleanup for #3821 (#3822)
- @agjohnson: Add temporary method for skipping submodule checkout (#3821)
- @stsewd: Pin astroid to fix linter issue on travis (#3816)
- @varunotelli: Update install.rst dropped the Python 2.7 only part (#3814)
- @xrmx: Update machine field when activating a version from project_version_detail (#3797)
- @humitos: Allow members of "Admin" Team to wipe version envs (#3791)

- @ericholscher: Add sustainability api to CORS (#3782)
- @durwasa-chakraborty: Fixed a grammatical error (#3780)
- @humitos: Trying to solve the end line character for a font file (#3776)
- @stsewd: Fix tox env for coverage (#3772)
- @bansalnitish: Added eslint rules (#3768)
- @davidfischer: Use sustainability api for advertising (#3747)
- @davidfischer: Add a sustainability API (#3672)
- @humitos: Upgrade django-pagination to a "maintained" fork (#3666)
- @humitos: Project updated when subproject modified (#3649)
- @davidfischer: Anonymize IP addresses for Google Analytics (#3626)
- @humitos: Improve "Sharing" docs (#3472)
- @humitos: Upgrade docker-py to its latest version (docker==3.1.1) (#3243)
- @humitos: Upgrade all packages using `pur` tool (#2916)
- @rixx: Fix page redirect preview (#2711)

1.38.34 Version 2.3.0

Warning: Version 2.3.0 includes a security fix for project translations. See security-2.3.0 for more information

- @stsewd: Fix tox env for coverage (#3772)
- @humitos: Try to fix end of file (#3761)
- @berkerpeksag: Fix indentation in docs/faq.rst (#3758)
- @stsewd: Check for http protocol before urlize (#3755)
- @rajujha373: #3741: replaced Go Crazy text with Search (#3752)
- @humitos: Log in the proper place and add the image name used (#3750)
- @shubham76: Changed 'Submit' text on buttons with something more meaningful (#3749)
- @agjohnson: Fix tests for Git submodule (#3737)
- @bansalnitish: Add eslint rules and fix errors (#3726)
- @davidfischer: Prevent bots indexing promos (#3719)
- @agjohnson: Add argument to skip errorlist through knockout on common form (#3704)
- @ajatprabha: Fixed #3701: added closing tag for div element (#3702)
- @bansalnitish: Fixes internal reference (#3695)
- @humitos: Always record the git branch command as success (#3693)
- @ericholscher: Show the project slug in the project admin (to make it more explicit what project is what) (#3681)
- @humitos: Upgrade django-taggit to 0.22.2 (#3667)

- @stsewd: Check for submodules (#3661)
- @agjohnson: Hotfix for adding logging call back into project sync task (#3657)
- @agjohnson: Fix issue with missing setting in oauth SyncRepo task (#3656)
- @ericholscher: Remove error logging that isn't an error. (#3650)
- @humitos: Project updated when subproject modified (#3649)
- @aasis21: formatting buttons in edit project text editor (#3633)
- @humitos: Filter by my own repositories at Import Remote Project (#3548)
- @funkyHat: check for matching alias before subproject slug (#2787)

1.38.35 Version 2.2.1

Version `2.2.1` is a bug fix release for the several issues
found in production during the `2.2.0` release.

- @agjohnson: Hotfix for adding logging call back into project sync task (#3657)
- @agjohnson: Fix issue with missing setting in oauth SyncRepo task (#3656)
- @humitos: Tests for build notifications (#3654)
- @humitos: Send proper context to celery email notification task (#3653)
- @ericholscher: Remove error logging that isn't an error. (#3650)
- @davidfischer: Update RTD security docs (#3641)
- @humitos: Ability to override the creation of the Celery App (#3623)

1.38.36 Version 2.2.0

- @humitos: Tests for build notifications (#3654)
- @humitos: Send proper context to celery email notification task (#3653)
- @xrmx: Update django-formtools to 2.1 (#3648)
- @xrmx: Update Django to 1.9.13 (#3647)
- @davidfischer: Fix a 500 when searching for files with API v1 (#3645)
- @davidfischer: Update RTD security docs (#3641)
- @humitos: Fix SVN initialization for command logging (#3638)
- @humitos: Ability to override the creation of the Celery App (#3623)
- @humitos: Update the operations team (#3621)
- @mohitkyadav: Add venv to .gitignore (#3620)
- @stsewd: Remove hardcoded copyright year (#3616)
- @stsewd: Improve installation steps (#3614)
- @stsewd: Update GSOC (#3607)
- @Jigar3: Updated AUTHORS.rst (#3601)
- @stsewd: Pin less to latest compatible version (#3597)

- @Angeles4four: Grammar correction (#3596)
- @davidfischer: Fix an unclosed tag (#3592)
- @aaksarin: add missed fontawesome-webfont.woff2 (#3589)
- @davidfischer: Force a specific ad to be displayed (#3584)
- @stsewd: Docs about preference for tags over branches (#3582)
- @davidfischer: Rework homepage (#3579)
- @stsewd: Don't allow to create a subproject of a project itself (#3571)
- @davidfischer: Fix for build screen in firefox (#3569)
- @humitos: Style using pre-commit (#3560)
- @humitos: Use DRF 3.1 `pagination_class` (#3559)
- @davidfischer: Analytics fixes (#3558)
- @davidfischer: Upgrade requests version (#3557)
- @humitos: Mount `pip_cache_path` in Docker container (#3556)
- @ericholscher: Add a number of new ideas for GSOC (#3552)
- @humitos: Fix Travis lint issue (#3551)
- @davidfischer: Send custom dimensions for mkdocs (#3550)
- @davidfischer: Promo contrast improvements (#3549)
- @humitos: Allow git tags with `/` in the name and properly slugify (#3545)
- @humitos: Allow to import public repositories on corporate site (#3537)
- @humitos: Log `git checkout` and expose to users (#3520)
- @stsewd: Update docs (#3498)
- @davidfischer: Switch to universal analytics (#3495)
- @stsewd: Move Mercurial dependency to pip.txt (#3488)
- @agjohnson: Add docs on removing edit button (#3479)
- @davidfischer: Convert default dev cache to local memory (#3477)
- @agjohnson: Fix lint error (#3402)
- @techtonik: Fix Edit links if version is referenced by annotated tag (#3302)
- @jaraco: Fixed build results page on firefox (part two) (#2630)

1.38.37 Version 2.1.6

- @davidfischer: Promo contrast improvements (#3549)
- @humitos: Refactor run command outside a Build and Environment (#3542)
- @AnatoliyURL: Project in the local read the docs don't see tags. (#3534)
- @malarzm: searchtools.js missing init() call (#3532)
- @johanneskoester: Build failed without details (#3531)
- @danielmitterdorfer: "Edit on Github" points to non-existing commit (#3530)

- @lk-geimfari: No such file or directory: 'docs/requirements.txt' (#3529)
- @stsewd: Fix Good First Issue link (#3522)
- @Blendify: Remove RTD Theme workaround (#3519)
- @stsewd: Move project description to the top (#3510)
- @davidfischer: Switch to universal analytics (#3495)
- @davidfischer: Convert default dev cache to local memory (#3477)
- @nlgranger: Github service: cannot unlink after deleting account (#3374)
- @andrewgodwin: "stable" appearing to track future release branches (#3268)
- @skddc: Add JSDoc to docs build environment (#3069)
- @chummels: RTD building old "stable" docs instead of "latest" when auto-triggered from recent push (#2351)
- @cajus: Builds get stuck in "Cloning" state (#2047)
- @gossi: Cannot delete subproject (#1341)
- @gigster99: extension problem (#1059)

1.38.38 Version 2.1.5

- @ericholscher: Add GSOC 2018 page (#3518)
- @stsewd: Move project description to the top (#3510)
- @RichardLitt: Docs: Rename "Good First Bug" to "Good First Issue" (#3505)
- @stsewd: Fix regex for getting project and user (#3501)
- @ericholscher: Check to make sure changes exist in BitBucket pushes (#3480)
- @andrewgodwin: "stable" appearing to track future release branches (#3268)
- @cdeil: No module named pip in conda build (#2827)
- @Yaseenh: building project does not generate new pdf with changes in it (#2758)
- @chummels: RTD building old "stable" docs instead of "latest" when auto-triggered from recent push (#2351)
- @KeithWoods: GitHub edit link is aggressively stripped (#1788)

1.38.39 Version 2.1.4

- @davidfischer: Add programming language to API/READTHEDOCS_DATA (#3499)
- @ericholscher: Remove our mkdocs search override (#3496)
- @humitos: Better style (#3494)
- @humitos: Update README.rst (#3492)
- @davidfischer: Small formatting change to the Alabaster footer (#3491)
- @matsen: Fixing "reseting" misspelling. (#3487)
- @ericholscher: Add David to dev team listing (#3485)
- @ericholscher: Check to make sure changes exist in BitBucket pushes (#3480)
- @ericholscher: Use semvar for readthedocs-build to make bumping easier (#3475)

- @davidfischer: Add programming languages (#3471)
- @humitos: Remove TEMPLATE_LOADERS since it's the default (#3469)
- @Code0x58: Minor virtualenv upgrade (#3463)
- @humitos: Remove invite only message (#3456)
- @maxirus: Adding to Install Docs (#3455)
- @stsewd: Fix a little typo (#3448)
- @stsewd: Better autogenerated index file (#3447)
- @stsewd: Better help text for privacy level (#3444)
- @msyriac: Broken link URL changed fixes #3442 (#3443)
- @ericholscher: Fix git (#3441)
- @ericholscher: Properly slugify the alias on Project Relationships. (#3440)
- @stsewd: Don't show "build ideas" to unprivileged users (#3439)
- @Blendify: Docs: Point Theme docs to new website (#3438)
- @humitos: Do not use double quotes on git command with –format option (#3437)
- @ericholscher: Hack in a fix for missing version slug deploy that went out a while back (#3433)
- @humitos: Check versions used to create the venv and auto-wipe (#3432)
- @ericholscher: Upgrade psycopg2 (#3429)
- @humitos: Fix "Edit in Github" link (#3427)
- @ericholscher: Add celery theme to supported ad options (#3425)
- @humitos: Link to version detail page from build detail page (#3418)
- @humitos: Move wipe button to version detail page (#3417)
- @humitos: Show/Hide "See paid advertising" checkbox depending on USE_PROMOS (#3412)
- @benjaoming: Strip well-known version component origin/ from remote version (#3377)
- @humitos: Remove warnings from code (#3372)
- @ericholscher: Add docker image from the YAML config integration (#3339)
- @humitos: Show proper error to user when conf.py is not found (#3326)
- @humitos: Simple task to finish inactive builds (#3312)
- @techtonik: Fix Edit links if version is referenced by annotated tag (#3302)
- @Riyuzakii: changed from html to css (#2699)

1.38.40 Version 2.1.3

date Dec 21, 2017

- @ericholscher: Upgrade psycopg2 (#3429)
- @humitos: Fix "Edit in Github" link (#3427)
- @ericholscher: Add celery theme to supported ad options (#3425)
- @ericholscher: Only build travis push builds on master. (#3421)

- @ericholscher: Add concept of dashboard analytics code (#3420)
- @humitos: Use default avatar for User/Orgs in OAuth services (#3419)
- @humitos: Link to version detail page from build detail page (#3418)
- @humitos: Move wipe button to version detail page (#3417)
- @bieagrathara: 019 497 8360 (#3416)
- @bieagrathara: rew (#3415)
- @tony: lint prospector task failing (#3414)
- @humitos: Remove extra 's' (#3413)
- @humitos: Show/Hide "See paid advertising" checkbox depending on USE_PROMOS (#3412)
- @accraze: Removing talks about RTD page (#3410)
- @humitos: Pin pylint to 1.7.5 and fix docstring styling (#3408)
- @agjohnson: Update style and copy on abandonment docs (#3406)
- @agjohnson: Update changelog more consistently (#3405)
- @agjohnson: Update prerelease invoke command to call with explicit path (#3404)
- @ericholscher: Fix changelog command (#3403)
- @agjohnson: Fix lint error (#3402)
- @julienmalard: Recent builds are missing translated languages links (#3401)
- @stsewd: Remove copyright application (#3400)
- @humitos: Show connect buttons for installed apps only (#3394)
- @agjohnson: Fix display of build advice (#3390)
- @agjohnson: Don't display the build suggestions div if there are no suggestions (#3389)
- @ericholscher: Pass more data into the redirects. (#3388)
- @ericholscher: Fix issue where you couldn't edit your canonical domain. (#3387)
- @benjaoming: Strip well-known version component origin/ from remote version (#3377)
- @humitos: Remove warnings from code (#3372)
- @JavaDevVictoria: Updated python.setup_py_install to be true (#3357)
- @humitos: Use default avatars for GitLab/GitHub/Bitbucket integrations (users/organizations) (#3353)
- @jonrkarr: Error in YAML configuration docs: default value for `python.setup_py_install` should be `true` (#3334)
- @humitos: Show proper error to user when conf.py is not found (#3326)
- @MikeHart85: Badges aren't updating due to being cached on GitHub. (#3323)
- @humitos: Simple task to finish inactive builds (#3312)
- @techtonik: Fix Edit links if version is referenced by annotated tag (#3302)
- @humitos: Remove/Update talks about RTD page (#3283)
- @gawel: Regain pyquery project ownership (#3281)
- @dialex: Build passed but I can't see the documentation (maze screen) (#3246)

- @makixx: Account is inactive (#3241)
- @agjohnson: Cleanup misreported failed builds (#3230)
- @cokelaer: links to github are broken (#3203)
- @agjohnson: Remove copyright application (#3199)
- @shacharoo: Unable to register after deleting my account (#3189)
- @gtalarico: 3 week old Build Stuck Cloning (#3126)
- @agjohnson: Regressions with conf.py and error reporting (#2963)
- @agjohnson: Can't edit canonical domain (#2922)
- @virtuald: Documentation stuck in 'cloning' state (#2795)
- @Riyuzakii: changed from html to css (#2699)
- @tjanez: Support specifying 'python setup.py build_sphinx' as an alternative build command (#1857)
- @bdarnell: Broken edit links (#1637)

1.38.41 Version 2.1.2

- @agjohnson: Update changelog more consistently (#3405)
- @agjohnson: Update prerelease invoke command to call with explicit path (#3404)
- @agjohnson: Fix lint error (#3402)
- @stsewd: Remove copyright application (#3400)
- @humitos: Show connect buttons for installed apps only (#3394)
- @agjohnson: Don't display the build suggestions div if there are no suggestions (#3389)
- @jonrkarr: Error in YAML configuration docs: default value for `python.setup_py_install` should be `true` (#3334)
- @humitos: Simple task to finish inactive builds (#3312)
- @agjohnson: Cleanup misreported failed builds (#3230)
- @agjohnson: Remove copyright application (#3199)

1.38.42 Version 2.1.1

Release information missing

1.38.43 Version 2.1.0

- @ericholscher: Revert "Merge pull request #3336 from rtfd/use-active-for-stable" (#3368)
- @agjohnson: Revert "Do not split before first argument (#3333)" (#3366)
- @ericholscher: Remove pitch from ethical ads page, point folks to actual pitch page. (#3365)
- @agjohnson: Add changelog and changelog automation (#3364)
- @ericholscher: Fix mkdocs search. (#3361)

- @ericholscher: Email sending: Allow kwargs for other options (#3355)
- @ericholscher: Try and get folks to put more tags. (#3350)
- @ericholscher: Suggest wiping your environment to folks with bad build outcomes. (#3347)
- @humitos: GitLab Integration (#3327)
- @jimfulton: Draft policy for claiming existing project names. (#3314)
- @agjohnson: More logic changes to error reporting, cleanup (#3310)
- @safwanrahman: [Fix #3182] Better user deletion (#3214)
- @ericholscher: Better User deletion (#3182)
- @RichardLitt: Add `Needed: replication` label (#3138)
- @josejrobles: Replaced usage of deprecated function get_fields_with_model with new … (#3052)
- @ericholscher: Don't delete the subprojects directory on sync of superproject (#3042)
- @andrew: Pass query string when redirecting, fixes #2595 (#3001)
- @saily: Add GitLab repo sync and webhook support (#1870)
- @destroyerofbuilds: Setup GitLab Web Hook on Project Import (#1443)
- @takotuesday: Add GitLab Provider from django-allauth (#1441)

1.38.44 Version 2.0

- @ericholscher: Email sending: Allow kwargs for other options (#3355)
- @ericholscher: Try and get folks to put more tags. (#3350)
- @ericholscher: Small changes to email sending to enable from email (#3349)
- @dplanella: Duplicate TOC entries (#3345)
- @ericholscher: Small tweaks to ethical ads page (#3344)
- @agjohnson: Fix python usage around oauth pagination (#3342)
- @tony: Fix isort link (#3340)
- @ericholscher: Change stable version switching to respect `active` (#3336)
- @ericholscher: Allow superusers to pass admin & member tests for projects (#3335)
- @humitos: Do not split before first argument (#3333)
- @humitos: Update docs for pre-commit (auto linting) (#3332)
- @humitos: Take preferece of tags over branches when selecting the stable version (#3331)
- @humitos: Add prospector as a pre-commit hook (#3328)
- @andrewgodwin: "stable" appearing to track future release branches (#3268)
- @humitos: Config files for auto linting (#3264)
- @mekrip: Build is not working (#3223)
- @skddc: Add JSDoc to docs build environment (#3069)
- @jakirkham: Specifying conda version used (#2076)
- @agjohnson: Document code style guidelines (#1475)

1.38.45 Previous releases

Starting with version `2.0`, we will be incrementing the Read the Docs version based on semantic versioning principles, and will be automating the update of our changelog.

Below are some historical changes from when we have tried to add information here in the past

July 23, 2015

- Django 1.8 Support Merged

Code Notes

- Updated Django from `1.6.11` to `1.8.3`.
- Removed South and ported the South migrations to Django's migration framework.
- Updated django-celery from `3.0.23` to `3.1.26` as django-celery 3.0.x does not support Django 1.8.
- Updated Celery from `3.0.24` to `3.1.18` because we had to update django-celery. We need to test this extensively and might need to think about using the new Celery API directly and dropping django-celery. See release notes: http://docs.celeryproject.org/en/latest/whatsnew-3.1.html
- Updated tastypie from `0.11.1` to current master (commit `1e1aff3dd4dcd21669e9c68bd7681253b286b856`) as 0.11.x is not compatible with Django 1.8. No surprises expected but we should ask for a proper release, see release notes: https://github.com/django-tastypie/django-tastypie/blob/master/docs/release_notes/v0.12.0.rst
- Updated django-oauth from `0.16.1` to `0.21.0`. No surprises expected, see release notes in the docs and finer grained in the repo
- Updated django-guardian from `1.2.0` to `1.3.0` to gain Django 1.8 support. No surprises expected, see release notes: https://github.com/lukaszb/django-guardian/blob/devel/CHANGES
- Using `django-formtools` instead of removed `django.contrib.formtools` now. Based on the Django release notes, these modules are the same except of the package name.
- Updated pytest-django from `2.6.2` to `2.8.0`. No tests required, but running the testsuite :smile:
- Updated psycopg2 from 2.4 to 2.4.6 as 2.4.5 is required by Django 1.8. No trouble expected as Django is the layer between us and psycopg2. Also it's only a minor version up-

grade. Release notes: http://initd.org/psycopg/docs/news.html#what-s-new-in-psycopg-2-4-6

- Added `django.setup()` to `conf.py` to load django properly for doc builds.
- Added migrations for all apps with models in the `readthedocs/` directory

Deployment Notes

After you have updated the code and installed the new dependencies, you need to run these commands on the server:

```
python manage.py migrate contenttypes
python␣
↪manage.py migrate projects 0002 --fake
python manage.py migrate --fake-initial
```

Locally I had trouble in a test environment that pip did not update to the specified commit of tastypie. It might be required to use `pip install -U -r requirements/deploy.txt` during deployment.

Development Update Notes

The readthedocs developers need to execute these commands when switching to this branch (or when this got merged into master):

- **Before updating** please make sure that all migrations are applied:

  ```
  python manage.py syncdb
  python manage.py migrate
  ```

- Update the codebase: `git pull`
- You need to update the requirements with `pip install -r requirements.txt`
- Now you need to fake the initial migrations:

  ```
  python manage.py migrate contenttypes
  python␣
  ↪manage.py migrate projects 0002 --fake
  python manage.py migrate --fake-initial
  ```

1.39 Installation

Here is a step by step guide on how to install Read the Docs. It will get you to a point of having a local running instance.

1.39.1 Requirements

First, obtain Python 3.6 and virtualenv if you do not already have them. Using a virtual environment is strongly recommended, since it will help you to avoid clutter in your system-wide libraries.

Additionally Read the Docs depends on:

- Git (version >=2.17.0)
- Mercurial (only if you need to work with mercurial repositories)
- Pip (version >1.5)
- Redis
- Elasticsearch (only if you want full support for searching inside the site)

– Ubuntu users could install this package by following *Enabling Elasticsearch on the local server*.

Note: If you plan to import Python 2 projects to your RTD, then you'll need to install Python 2 with virtualenv in your system as well.

In order to get all the dependencies successfully installed, you need these libraries.

Mac OS

If you are having trouble on OS X Mavericks (or possibly other versions of OS X) with building `lxml`, you probably might need to use Homebrew to `brew install libxml2`, and invoke the install with:

```
CFLAGS=-
↪I/usr/local/opt/libxml2/include/libxml2 \
LDFLAGS=-L/usr/local/opt/libxml2/lib \
pip install -r requirements.txt
```

Ubuntu

Install:

```
sudo apt-get install build-essential
sudo apt-get install␣
↪python-dev python-pip python-setuptools
sudo apt-get install␣
↪libxml2-dev libxslt1-dev zlib1g-dev
```

If you don't have redis installed yet, you can do it with:

```
sudo apt-get install redis-server
```

CentOS/RHEL 7

Install:

```
sudo yum install python-devel␣
↪python-pip libxml2-devel libxslt-devel
```

Other OS

On other operating systems no further dependencies are required, or you need to find the proper equivalent libraries.

1.39.2 Get and run Read the Docs

Clone the repository somewhere on your disk and enter to the repository:

```
git clone https:/
↪/github.com/rtfd/readthedocs.org.git
cd readthedocs.org
```

Create a virtual environment and activate it:

```
virtualenv venv
source venv/bin/activate
```

Next, install the dependencies using `pip` (make sure you are inside of the virtual environment):

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

This may take a while, so go grab a beverage. When it's done, build the database:

```
python manage.py migrate
```

Then create a superuser account for Django:

```
python manage.py createsuperuser
```

Now let's properly generate the static assets:

```
python manage.py collectstatic
```

Now you can optionally load a couple users and test projects:

```
python manage.py loaddata test_data
```

Note: If you do not opt to install test data, you'll need to create an account for API use and set `SLUMBER_USERNAME` and `SLUMBER_PASSWORD` in order for everything to work properly. This can be done by using `createsuperuser`, then attempting a manual login to create an `EmailAddress` entry for the user, then you can use `shell_plus` to update the object with `primary=True`, `verified=True`.

Finally, you're ready to start the web server:

```
python manage.py runserver
```

Visit http://127.0.0.1:8000/ in your browser to see how it looks; you can use the admin interface via http://127.0.0.1:8000/admin (logging in with the superuser account you just created).

For builds to properly work as expected, it is necessary the port you're serving on (i.e. `python manage.py runserver 0.0.0.0:8080`) match the port defined in `PRODUCTION_DOMAIN`. You can use `readthedocs/settings/local_settings.py` to modify this (by default, it's `localhost:8000`).

While the web server is running, you can build the documentation for the latest version of any project using the `update_repos` command. For example to update the `pip` repo:

```
python manage.py update_repos pip
```

Note: If you have problems building successfully a project, probably is because some missing libraries for `pdf` and `epub` generation. You can uncheck this on the advanced settings of your project.

1.39.3 What's available

After registering with the site (or creating yourself a superuser account), you will be able to log in and view the dashboard.

Importing your docs

One of the goals of readthedocs.org is to make it easy for any open source developer to get high quality hosted docs with great visibility! Simply provide us with the clone URL to your repo, we'll pull your code, extract your docs, and build them!

We make available a post-commit webhook that can be configured to update the docs whenever you commit to your repo. See our *Importing Your Documentation* page to learn more.

1.39.4 Further steps

By now you can trigger builds on your local environment, to encapsulate the build process inside a Docker container, see *Build Environments*.

For building this documentation, see *Building and Contributing to Documentation*.

And for setting up for the front end development, see *Front End Development*.

1.40 Architecture

Read the Docs is architected to be highly available. A lot of projects host their documentation with us, so we have built the site so that it shouldn't go down. The load balancer is the only real single point of failure currently. This means mainly that if the network to the load balancer goes down, we have issues.

1.40.1 Diagram

(continues on next page)

(continued from previous page)

```
↪                    |                    |
↪                    +--------------------+
```

1.41 Testing

Before contributing to Read the Docs, make sure your patch passes our test suite and your code style passes our code linting suite.

Read the Docs uses Tox to execute testing and linting procedures. Tox is the only dependency you need to run linting or our test suite, the remainder of our requirements will be installed by Tox into environment specific virtualenv paths. Before testing, make sure you have Tox installed:

```
pip install tox
```

To run the full test and lint suite against your changes, simply run Tox. Tox should return without any errors. You can run Tox against all of our environments by running:

```
tox
```

In order to run all test including the search tests, include `"'--including-search'"` argument:

```
tox "'--including-search'"
```

To target a specific environment:

```
tox -e py27
```

The `tox` configuration has the following environments configured. You can target a single environment to limit the test suite:

```
py27
    Run our test suite using Python 2.7

lint
    Run code linting using `Prospector`_
↪. This currently runs `pylint`_,
    `pyflakes`_
↪, `pep8`_ and other linting tools.

docs
    Test␣
↪documentation compilation with Sphinx.
```

1.41.1 Continuous Integration

The RTD test suite is exercised by Travis CI on every push to our repo at GitHub. You can check out the current build status: https://travis-ci.org/rtfd/readthedocs.org

1.42 Building and Contributing to Documentation

As one might expect, the documentation for Read the Docs is built using Sphinx and hosted on Read the Docs. The docs are kept in the `docs/` directory at the top of the source tree.

You can build the docs by installing `Sphinx` and running:

```
# in the docs directory
make html
```

Please follow these guidelines when updating our docs. Let us know if you have any questions or something isn't clear.

1.42.1 The brand

We are called **Read the Docs**. The *the* is not capitalized.

We do however use the acronym **RTD**.

1.42.2 Titles

For page titles, or Heading1 as they are sometimes called, we use title-case.

If the page includes multiple sub-headings (H2, H3), we usually use sentence-case unless the titles include terminology that is supposed to be capitalized.

1.42.3 Content

- Do not break the content across multiple lines at 80 characters, but rather break them on semantic meaning (e.g. periods or commas). Read more about this here.
- If you are cross-referencing to a different page within our website, use the `doc` directive and not a hyperlink.

1.43 Front End Development

1.43.1 Background

Note: Consider this the canonical resource for contributing Javascript and CSS. We are currently in the process of modernizing our front end development procedures. You will see a lot of different styles around the code base for front end JavaScript and CSS.

Our modern front end development stack includes the following tools:

- Gulp
- Bower
- Browserify
- Debowerify
- And soon, LESS

We use the following libraries:

- Knockout
- jQuery
- Several jQuery plugins

Previously, JavaScript development has been done in monolithic files or inside templates. jQuery was added as a global object via an include in the base template to an external source. There are no standards currently to JavaScript libraries, this aims to solve that.

The requirements for modernizing our front end code are:

- Code should be modular and testable. One-off chunks of JavaScript in templates or in large monolithic files are not easily testable. We currently have no JavaScript tests.
- Reduce code duplication.
- Easy JavaScript dependency management.

Modularizing code with Browserify is a good first step. In this development workflow, major dependencies commonly used across JavaScript includes are installed with Bower for testing, and vendorized as standalone libraries via Gulp and Browserify. This way, we can easily test our JavaScript libraries against jQuery/etc, and have the flexibility of modularizing our code. See *JavaScript Bundles* for more information on what and how we are bundling.

To ease deployment and contributions, bundled JavaScript is checked into the repository for now. This ensures new contributors don't need an additional front end stack just for making changes to our Python code base. In the future, this may change, so that assets are compiled before deployment, however as our front end assets are in a state of flux, it's easier to keep absolute sources checked in.

1.43.2 Getting Started

You will need a working version of Node and NPM to get started. We won't cover that here, as it varies from platform to platform.

To install these tools and dependencies:

```
npm install
```

Next, install front end dependencies:

```
bower install
```

The sources for our bundles are found in the per-application path `static-src`, which has the same directory structure as `static`. Files in `static-src` are compiled to `static` for static file collection in Django. Don't edit files in `static` directly, unless you are sure there isn't a source file that will compile over your changes.

To test changes while developing, which will watch source files for changes and compile as necessary, you can run Gulp with our development target:

```
npm run dev
```

Once you are satisfied with your changes, finalize the bundles (this will minify library sources):

```
npm run build
```

If you updated any of our vendor libraries, compile those:

```
npm run vendor
```

Make sure to check in both files under `static` and `static-src`.

Note: We run Gulp through an `npm` script in order to ensure that the correct version from `package.json` is used.

1.43.3 Making Changes

If you are creating a new library, or a new library entry point, make sure to define the application source file in `gulpfile.js`, this is not handled automatically right now.

If you are bringing in a new vendor library, make sure to define the bundles you are going to create in `gulpfile.js` as well.

Tests should be included per-application, in a path called `tests`, under the `static-src/js` path you are work-

ing in. Currently, we still need a test runner that accumulates these files.

1.43.4 Deployment

If merging several branches with JavaScript changes, it's important to do a final post-merge bundle. Follow the steps above to rebundle the libraries, and check in any changed libraries.

1.43.5 JavaScript Bundles

There are several components to our bundling scheme:

Vendor library We repackage these using Browserify, Bower, and Debowerify to make these libraries available by a `require` statement. Vendor libraries are packaged separately from our JavaScript libraries, because we use the vendor libraries in multiple locations. Libraries bundled this way with Browserify are available to our libraries via `require` and will back down to finding the object on the global `window` scope.

Vendor libraries should only include libraries we are commonly reusing. This currently includes `jQuery` and `Knockout`. These modules will be excluded from libraries by special includes in our `gulpfile.js`.

Minor third party libraries These libraries are maybe used in one or two locations. They are installed via Bower and included in the output library file. Because we aren't reusing them commonly, they don't require a separate bundle or separate include. Examples here would include jQuery plugins used on one off forms, such as jQuery Payments.

Our libraries These libraries are bundled up excluding vendor libraries ignored by rules in our `gulpfile.js`. These files should be organized by function and can be split up into multiple files per application.

Entry points to libraries must be defined in `gulpfile.js` for now. We don't have a defined directory structure that would make it easy to imply the entry point to an application library.

1.44 Build Environments

Read the Docs uses container virtualization to encapsulate documentation build processes. Each build spins up a new virtual machine using our base image, which is an image with the minimum necessary components required to build documentation. Virtual machines are limiting in CPU time

and memory, which aims to reduce excessive usage of build resources.

1.44.1 Setup

Build environments use Docker to handle container virtualization. To perform any development on the Docker build system, you will need to set up Docker on your host system. Setup of Docker will vary by system, and so is out of the scope of this documentation.

Once you have Docker set up, you will need to pull down our build image. These images are found on our Docker Hub repository, the source comes from our container image repo.

To get started using Docker for build environments, you'll need to pull down at least one build image. For example, to pull down our latest image:

```
docker pull readthedocs/build:latest
```

The default image used by our build servers is `readthedocs/build:2.0`. This would be a good place to start testing as the `latest` version could operate differently. See `DOCKER_IMAGE` below for setting this configuration option.

After this image is downloaded, you can update your settings to use the new image – see *Configuration*.

1.44.2 Configuration

There are several settings used to configure usage of virtual machines:

DOCKER_ENABLE True/False value used to enable the Docker build environment.

Default: `False`

DOCKER_LIMITS A dictionary of limits to virtual machines. These limits include:

time An integer representing the total allowed time limit (in seconds) of build processes. This time limit affects the parent process to the virtual machine and will force a virtual machine to die if a build is still running after the allotted time expires.

memory The maximum memory allocated to the virtual machine. If this limit is hit, build processes will be automatically killed. Examples: '200m' for 200MB of total memory, or '2g' for 2GB of total memory.

Default: `None`

DOCKER_IMAGE Tag of a Docker image to use as a base image.

Default: `readthedocs/build:2.0`

DOCKER_SOCKET URI of the socket to connect to the Docker daemon. Examples include: `unix:///var/run/docker.sock` and `tcp://127.0.0.1:2375`.

Default: `None`

DOCKER_VERSION Version of the API to use for the Docker API client.

Default: `None`

1.45 How we use symlinks

Read the Docs stays highly available by serving all documentation pages out of nginx. This means that they never hit our Python layer, meaning that they never hit our database. This reduces the total number of servers to serve a request to 1, each of which is redundant.

1.45.1 Nginx

We handle a couple of different types of requests in nginx:

- Requests to a `readthedocs.io` subdomain
- Requests to a custom domain

1.45.2 Subdomains

For subdomains, this is a simple lookup of the project slug, using the subdomain portion of the request's hostname. This doesn't require symlinks, but it shows the basic logic that we need to replicate.

When a user navigates to `http://pip.readthedocs.org/en/latest/`, we know that they want the pip documentation. So we simply serve them the documentation:

```
location ~ ^/en/(.+)/(.*) {
    alias /home/docs/checkouts/readthedocs.
↪org/user_builds/$domain/rtd-builds/$1/$2;
    error_page 404 = @fallback;
    error_page 500 = @fallback;
}

location @fallback {
    proxy_pass http://127.0.0.1:8888;
    proxy_set_header Host $host;

    proxy_set_header X-Real-IP $remote_addr;
```

(continues on next page)

(continued from previous page)

```
    proxy_set_header X-
↪Forwarded-For $proxy_add_x_forwarded_for;
    add_header X-Deity Asgard;
}
```

Note: The `@fallback` directive is hit when we don't find the proper file. This will cause things to hit the Python backend, so that proper action can be taken.

1.45.3 Custom domains

Custom domains add a bit of difficulty, because at the nginx layer we don't know what documentation to serve. When someone requests `http://docs.fabfile.org/en/latest/`, we can't look at the URL to know to serve the `fabric` docs.

This is where symlinks come in. When someone requests `http://docs.fabfile.org/en/latest/` the first time, it hits the Python layer. In that Python layer we record that `docs.fabfile.org` points at `fabric`. When we build the `fabric` docs, we create a symlink for all domains that have pointed at `fabric` before.

So, when we get a request for `docs.fabfile.org` in the future, we will be able to serve it directly from nginx. In this example, $host would be `docs.fabfile.org`:

```
location␣
↪~ ^/en/(?P<doc_version>.+)/(?P<path>.*) {
    alias /home/docs/checkouts/readthedocs.
↪org/cnames/$host/$doc_version/$path;
    error_page 404 = @fallback;
    error_page 500 = @fallback;
}
```

Notice that nowhere in the above path is the project's slug mentioned. It is simply there in the symlink in the cnames directory, and the docs are served from there.

1.46 Interesting Settings

1.46.1 SLUMBER_USERNAME

Default: `test`

The username to use when connecting to the Read the Docs API. Used for hitting the API while building the docs.

1.46.2 SLUMBER_PASSWORD

Default: `test`

The password to use when connecting to the Read the Docs API. Used for hitting the API while building the docs.

1.46.3 USE_SUBDOMAIN

Default: `False`

Whether to use subdomains in URLs on the site, or the Django-served content. When used in production, this should be `True`, as Nginx will serve this content. During development and other possible deployments, this might be `False`.

1.46.4 PRODUCTION_DOMAIN

Default: `localhost:8000`

This is the domain that gets linked to throughout the site when used in production. It depends on `USE_SUBDOMAIN`, otherwise it isn't used.

1.46.5 MULTIPLE_APP_SERVERS

Default: `None`

This is a list of application servers that built documentation is copied to. This allows you to run an independent build server, and then have it rsync your built documentation across multiple front end documentation/app servers.

1.46.6 DEFAULT_PRIVACY_LEVEL

Default: `public`

What privacy projects default to having. Generally set to `public`. Also acts as a proxy setting for blocking certain historically insecure options, like serving generated artifacts directly from the media server.

1.46.7 INDEX_ONLY_LATEST

Default: `None`

In search, only index the `latest` version of a Project.

1.46.8 DOCUMENT_PYQUERY_PATH

Default: `None`

The Pyquery path to an HTML element that is the root of your document. This is used for making sure we are only searching the main content of a document.

1.46.9 USE_PIP_INSTALL

Default: `None`

Whether to use `pip install .` or `python setup.py install` when installing packages into the Virtualenv. Default is to use `python setup.py install`.

1.46.10 PUBLIC_DOMAIN

Default: `None`

A special domain for serving public documentation. If set, public docs will be linked here instead of the `PRODUCTION_DOMAIN`.

1.46.11 PUBLIC_DOMAIN_USES_HTTPS

Default: `False`

If `True` and `PUBLIC_DOMAIN` is set, that domain will default to serving public documentation over HTTPS. By default, documentation is served over HTTP.

1.46.12 ALLOW_ADMIN

Default: `None`

Whether to include `django.contrib.admin` in the URL's.

1.47 Internationalization

This document covers the details regarding internationalization and localization that are applied in Read the Docs. The guidelines described are mostly based on Kitsune's localization documentation.

As with most of the Django applications out there, Read the Docs' i18n/l10n framework is based on GNU gettext. Crowd-sourced localization is optionally available at Transifex.

For more information about the general ideas, look at this document: http://www.gnu.org/software/gettext/manual/html_node/Concepts.html

1.47.1 Making Strings Localizable

Making strings in templates localizable is exceptionally easy. Making strings in Python localizable is a little more complicated. The short answer, though, is to just wrap the string in `_()`.

Interpolation

A string is often a combination of a fixed string and something changing, for example, `Welcome, James` is a combination of the fixed part `Welcome,`, and the changing part `James`. The naive solution is to localize the first part and then follow it with the name:

```
_('Welcome, ') + username
```

This is **wrong!**

In some locales, the word order may be different. Use Python string formatting to interpolate the changing part into the string:

```
_('Welcome, {name}').format(name=username)
```

Python gives you a lot of ways to interpolate strings. The best way is to use Py3k formatting and kwargs. That's the clearest for localizers.

Localization Comments

Sometimes, it can help localizers to describe where a string comes from, particularly if it can be difficult to find in the interface, or is not very self-descriptive (e.g. very short strings). If you immediately precede the string with a comment that starts with `Translators:`, the comment will be added to the PO file, and visible to localizers.

Example:

```
DEFAULT_THEME_CHOICES = (
    # Translators:␣
↪This is a name of a Sphinx theme.
    (THEME_DEFAULT, _('Default')),
    # Translators:␣
↪This is a name of a Sphinx theme.
    (THEME_SPHINX, _('Sphinx Docs')),
    # Translators:␣
↪This is a name of a Sphinx theme.
    (THEME_TRADITIONAL, _('Traditional')),
```

(continues on next page)

(continued from previous page)

```
    # Translators:␣
↪This is a name of a Sphinx theme.
    (THEME_NATURE, _('Nature')),
    # Translators:␣
↪This is a name of a Sphinx theme.
    (THEME_HAIKU, _('Haiku')),
)
```

Adding Context with msgctxt

Strings may be the same in English, but different in other languages. English, for example, has no grammatical gender, and sometimes the noun and verb forms of a word are identical.

To make it possible to localize these correctly, we can add "context" (known in gettext as *msgctxt*) to differentiate two otherwise identical strings. Django provides a `pgettext()` function for this.

For example, the string *Search* may be a noun or a verb in English. In a heading, it may be considered a noun, but on a button, it may be a verb. It's appropriate to add a context (like *button*) to one of them.

Generally, we should only add context if we are sure the strings aren't used in the same way, or if localizers ask us to.

Example:

```
from␣
↪django.utils.translation import pgettext

month = pgettext("text for␣
↪the search button on the form", "Search")
```

Plurals

You have 1 new messages grates on discerning ears. Fortunately, gettext gives us a way to fix that in English *and* other locales, the `ngettext()` function:

```
ngettext('singular␣
↪sentence', 'plural sentence', count)
```

A more realistic example might be:

```
ngettext('Found {count} result.',
         'Found {count} results',
        ␣
↪ len(results)).format(count=len(results))
```

This method takes three arguments because English only needs three, i.e., zero is considered "plural" for English. Other languages may have different plural rules, and require different phrases for, say 0, 1, 2-3, 4-10, >10. That's absolutely fine, and gettext makes it possible.

1.47.2 Strings in Templates

When putting new text into a template, all you need to do is wrap it in a `{% trans %}` template tag:

```
<h1>{% trans "Heading" %}</h1>
```

Context can be added, too:

```
<h1>{% trans␣
↪"Heading" context "section name" %}</h1>
```

Comments for translators need to precede the internationalized text and must start with the `Translators:` keyword.:

```
{# Translators: This heading is␣
↪displayed in the user's profile page #}
<h1>{% trans "Heading" %}</h1>
```

To interpolate, you need to use the alternative and more verbose `{% blocktrans %}` template tag — it's actually a block:

```
{% blocktrans␣
↪%}Welcome, {{ name }}!{% endblocktrans %}
```

Note that the `{{ name }}` variable needs to exist in the template context.

In some situations, it's desirable to evaluate template expressions such as filters or accessing object attributes. You can't do that within the `{% blocktrans %}` block, so you need to bind the expression to a local variable first:

```
{% blocktrans trimmed with revision.
↪created_date|timesince as timesince %}
{{ revision }} {{ timesince }} ago
{% endblocktrans %}

{% blocktrans with project.name as name␣
↪%}Delete {{ name }}?{% endblocktrans %}
```

`{% blocktrans %}` also provides pluralization. For that you need to bind a counter with the name `count` and provide a plural translation after the `{% plural %}` tag:

```
{% blocktrans trimmed with amount=article.
↪price count years=i.length %}
That will cost $ {{ amount }} per year.
```

(continues on next page)

(continued from previous page)

```
{% plural %}
That will cost␣
↪$ {{ amount }} per {{ years }} years.
{% endblocktrans %}
```

Note: The previous multi-lines examples also use the `trimmed` option. This removes newline characters and replaces any whitespace at the beginning and end of a line, helping translators when translating these strings.

1.47.3 Strings in Python

Note: Whenever you are adding a string in Python, ask yourself if it really needs to be there, or if it should be in the template. Keep logic and presentation separate!

Strings in Python are more complex for two reasons:

1. We need to make sure we're always using Unicode strings and the Unicode-friendly versions of the functions.
2. If you use the `ugettext()` function in the wrong place, the string may end up in the wrong locale!

Here's how you might localize a string in a view:

```
from django.
↪utils.translation import ugettext as _

def my_view(request):
    if request.user.is_superuser:
        msg = _(u'Oh hi, staff!')
    else:
        msg = _(u'You are not staff!')
```

Interpolation is done through normal Python string formatting:

```
msg = _(u'Oh, hi, {user}
↪').format(user=request.user.username)
```

Context information can be supplied by using the `pgettext()` function:

```
msg = pgettext('the context', 'Search')
```

Translator comments are normal one-line Python comments:

```
# Translators: A message to users.
msg = _(u'Oh, hi there!')
```

If you need to use plurals, import the `ungettext()` function:

```
from␣
↪django.utils.translation import ungettext

n = len(results)
msg = ungettext('Found {0} result
↪', 'Found {0} results', n).format(n)
```

Lazily Translated Strings

You can use `ugettext()` or `ungettext()` only in views or functions called from views. If the function will be evaluated when the module is loaded, then the string may end up in English or the locale of the last request!

Examples include strings in module-level code, arguments to functions in class definitions, strings in functions called from outside the context of a view. To internationalize these strings, you need to use the `_lazy` versions of the above methods, `ugettext_lazy()` and `ungettext_lazy()`. The result doesn't get translated until it is evaluated as a string, for example by being output or passed to `unicode()`:

```
from django.utils.
↪translation import ugettext_lazy as _

class UserProfileForm(forms.ModelForm):
    first_name = CharField(label=_
↪('First name'), required=False)
    last_name = CharField(label=_
↪('Last name'), required=False)
```

In case you want to provide context to a lazily-evaluated gettext string, you will need to use `pgettext_lazy()`.

1.47.4 Administrative Tasks

Updating Localization Files

To update the translation source files (eg if you changed or added translatable strings in the templates or Python code) you should run `python manage.py makemessages -l <language>` in the project's root directory (substitute `<language>` with a valid language code).

The updated files can now be localized in a PO editor or crowd-sourced online translation tool.

Compiling to MO

Gettext doesn't parse any text files, it reads a binary format for faster performance. To compile the latest PO files in

the repository, Django provides the `compilemessages` management command. For example, to compile all the available localizations, just run:

```
$ python manage.py compilemessages -a
```

You will need to do this every time you want to push updated translations to the live site.

Also, note that it's not a good idea to track MO files in version control, since they would need to be updated at the same pace PO files are updated, so it's silly and not worth it. They are ignored by `.gitignore`, but please make sure you don't forcibly add them to the repository.

Transifex Integration

To push updated translation source files to Transifex, run `tx push -s` (for English) or `tx push -t <language>` (for non-English).

To pull changes from Transifex, run `tx pull -a`. Note that Transifex does not compile the translation files, so you have to do this after the pull (see the *Compiling to MO* section).

For more information about the `tx` command, read the Transifex client's help pages.

Note: For the Read the Docs community site, we use a fabric script to follow this process:

1. Update files and push sources (English) to Transifex:

   ```
   $ fab i18n_push_source
   ```

2. Pull changes (new translations) from Transifex:

   ```
   $ fab i18n_pull
   ```

1.48 Overview of issue labels

Here is a full list of labels that we use in the GitHub issue tracker and what they stand for.

Accepted Issues with this label are issues that the core team has accepted on to the roadmap. The core team focuses on accepted bugs, features, and improvements that are on our immediate roadmap and will give priority to these issues. Pull requests could be delayed or closed if the pull request doesn't align with our current roadmap. An issue or pull request that has not been accepted should either eventually move to an accepted state, or should be closed. As an issue is accepted, we will find room for it on our roadmap, either

on an upcoming release (point release milestones), or on a future milestone project (named milestones).

Bug An issue describing unexpected or malicious behaviour of the readthedocs.org software. A Bug issue differs from an Improvement issue in that Bug issues are given priority on our roadmap. On release, these issues generally only warrant incrementing the patch level version.

Design Issues related to the UI of the readthedocs.org website.

Feature Issues that describe new features. Issues that do not describe new features, such as code cleanup or fixes that are not related to a bug, should probably be given the Improvement label instead. On release, issues with the Feature label warrant at least a minor version increase.

Good First Issue This label marks issues that are easy to get started with. The issue should be ideal for beginners to dive into the code base.

Priority: high Issues with this label should be resolved as quickly as possible.

Priority: low Issues with this label won't have the immediate focus of the core team.

Improvement An issue with this label is not a Bug nor a Feature. Code cleanup or small changes to existing features would likely have this label. The distinction for this label is that these issues have a lower priority on our roadmap compared to issues labeled Bug, and aren't implementing new features, such as a Feature issue might.

Needed: design decision Issues that need a design decision are blocked for development until a project leader clarifies the way in which the issue should be approached.

Needed: documentation If an issue involves creating or refining documentation, this label will be assigned.

Needed: more information This label indicates that a reply with more information is required from the bug reporter. If no response is given by the reporter, the issue is considered invalid after 2 weeks and will be closed. See the documentation about our *triage process* for more information.

Needed: patch This label indicates that a patch is required in order to resolve the issue. A fix should be proposed via a pull request on GitHub.

Needed: tests This label indicates that a better test coverage is required to resolve the issue. New tests should be proposed via a pull request on GitHub.

Needed: replication This label indicates that a bug has been reported, but has not been successfully replicated by another user or contributor yet.

Operations Issues that require changes in the server infrastructure.

PR: work in progress Pull Requests that are not complete yet. A final review is not possible yet, but every Pull Request is open for discussion.

PR: hotfix Pull request was applied directly to production after a release. These pull requests still need review to be merged into the next release.

Sprintable Sprintable are all issues that have the right amount of scope to be handled during a sprint. They are very focused and encapsulated.

Status: blocked The issue cannot be resolved until some other issue has been closed. See the issue's log for which issue is blocking this issue.

Status: stale A issue is stale if it there has been no activity on it for 90 days. Once a issue is determined to be stale, it will be closed after 2 weeks unless there is activity on the issue.

Support Questions that needs answering but do not require code changes or issues that only require a one time action on the server will have this label. See the documentation about our *triage process* for more information.

1.49 Security

Security is very important to us at Read the Docs. We are committed to responsible reporting and disclosure of security issues.

1.49.1 Reporting a security issue

If you believe you've discovered a security issue at Read the Docs, please contact us at **security@readthedocs.org** (optionally using our *PGP key*). We request that you please not publicly disclose the issue until it has been addressed by us.

You can expect:

- We will respond acknowledging your email typically within one business day.
- We will follow up if and when we have confirmed the issue with a timetable for the fix.
- We will notify you when the issue is fixed.
- We will add the issue to our *security issue archive*.

1.49.2 PGP key

You may use this PGP key to securely communicate with us and to verify signed messages you receive from us.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Comment: GPGTools - http://gpgtools.org

mQINBFqNz1ABEADvgtp3LT1pV5wuTyBPaKrbWBFj10eKyQ15wfgyc2RR6Ix5QnBo
6BcJ4fpgBhSwlngsrm0WU5kI/
 ↪jH7ySwzbDpYCRiLvGJx+pEYLuBBOSm6r5M1N+FV
xq3ShT4mHXhwPS1mKf9Xe+KlMdYa2e5TlBEr+TxGAmFFrOLjPxw6IDHgP3MVidr2
iHA2PAATl6H9ZYvNzLkI2sP7h0V1/
 ↪ADd43YpAK4yk6gdVjype5ez8lmoxDKNabMt
dSfdOup8zy/
 ↪fbC5KlxqrT9hHBkYfQWDLWXWcDW111q+ZvncujCrpONaY86bcQ3nn
QgkeWCwj254vvqsrygEU93reC2onWaROUKoLlX1/
 ↪1m2k2X3qze/hJRFZaljXVPKH
jV/
 ↪5q88EbjSUDgY5v9mdX8jhJAukx9HkOFdkMSh3RBgu1r+UPnCNd9K4T2nN0LBL
c9NTG0HW7Di5ivEVq74SqDIeiVeOrfY/
 ↪B6pRuUm/kNPcvZ+ZQPeNk6JUMqEemO9Q
h0VHSkgkhCPWPO9c9wWJz7O6y6vXgsFG7BZg7mTVOiKbdgneGo/
 ↪rKRvuBlQ7hCvP
PklwyRn90SJSgv7NF6HMm4TA1R9mzp+90oXjrDXARXmGTsPtcDXFv7xqpK1+Mfcn
ajEJYdIRNWVgx0E2RzHRipdG5MIQ5Plf4/
 ↪GasVHl71nMGY06oIu1T+44MQARAQAB
tFpSZWFkIHRoZSBEb2NzIFNlY3VyaXR5IFRlYW0gKGh0dHBzOi8vcmVhZHRoZWRv
Y3Mub3JnL3NlY3VyaXR5LykgPHNlY3VyaXR5QHJlYWR0aGVkb2NzLm9yZz6JAk4E
EwEIADgWIQRq+P453S2vjKvMbGn+75/
 ↪C3SHScQUCWo3PUAIbAwULCQgHAgYVCgkI
CwIEFgIDAQIeAQIXgAAKCRD+75/
 ↪C3SHScYMMD/4z0TN08oJ57Krg+UODXPsT9U3l
8fyKHhe6fJCTt5GQiWwBbkfa4M0YcxemIJGdgF1DpdSteWIL0tCwXbxHph+keYlg
z+EmF+W7OlnwbmtDx/Rj9VNdzf636DkMusTQzYEB/
 ↪+FdN4LtMVq7Al4CZ2Ca82F8
h0TLceh2bRgNjeWPuAMj7kS8tw3D9LmYA8d8Lv2c2jN7ba9p+QNKdSa4ErdJ0kbz
CSFcABPfc+LlYWFbm5j1ggzTONgR9R27mpAGMAtgSeAtxXLU0sQfLtCNaVkRyJ3C
s0awUvJCuq11YUPjz4HAcTWM4baAxK5LliEDOdaOlTK0q8T0sPP+SWt5JRL6/
 ↪Xc3
SwaXnVfzzZyeaSmRGEHmGQYBTB3WMUcH1RNH6uhNPCF4x3t0jOHWP7Eka4B9IdfE
cd+GDwqTKCHyddh8yUzTrmlSbdO7iuel6WVN0Xo1xzVrLUKpgDvB0UuPQXlxDLUc
WVrKv9rcyDVGVpDjQSQ4l191NDzlfzmDFkZ69Qe3E5Ir8oWBCMtHX3C99ocIcbR3
3mqOio2/QQCJzrMOWxgauF/
 ↪q4JMKQRj5Qq8US2U32xlPzB8O09z1e3kUOEy4gbwE
6LVMj6vxJqjV8/
 ↪4AOcocGgJKLLC9nqhf2sq5zA4TjI7lI25pgDDYozaLF2ss5nk3
t9hQmI5Q0MXGSsflAbkCDQRajc9QARAA30mNh2QaDVLcHEfJ9QKfqRfY3ddG6s6F
AHLR7mQ2tmO7ygzxgEcM7+7W+lmc9L+mZ5WutK5PIae/
 ↪MpAgOo220079aW3em2sz
+dIHdSE7CaajUahQaPqLY6W0bbZloGGDetPtOMKBTI1HtSNyKIsULsbyUA1SsEFn
aWtOof1MqqVQvYDwwwRj6T+DHtV17yO33v98k01Nx1SSThVY9wQ4MOZDBOAqWhie
iboDssrvtVZZihbQ9LM8TH/
 ↪l81auodBDpp96tgWguzjM4eyutaYZ6ZOLhfVUuEX+
gEqqJ7alXfDhh3NZUMHZ0SHVII7u7nqciTo7NS9rxBXfdGlKmC/
 ↪9Z3ehIXSpCnPY
JO42qMjPVM5/QDoeK9BWWX3rXmwnNzqK0D4L7zK/
 ↪cVnt2q9EqPCUJgOITJWEGc9G
crO0ni/
 ↪8M+BuhO/
 ↪4MeJJtrPtmq1b1BoeuYBzf1M7ARtnvtC5hLLrtxiy4UANlwSm
HFcIEt5UViwEqRuQWr5ZO3mwaJP2R/
 ↪foDHww7JYEqJ/GFI5RpT+3LWT5FXPC1QvU
```

(continues on next page)

(continued from previous page)

```
sbewD+ZmLSfifhC0WUzF002eadgXNyXSZKAirM8+yELM4xZAs0pJVlKVFRnis0OL
Wxdzthp2gTg+agtMoz27belxVUEmRK9GDaXi9XtJSooSglt0xlTimgB40nDPniVB
4h5S/gHsg8cAEQEAAYkCNgQYAQgAIBYhBGr4/
↪jndLa+Mq8xsaf7vn8LdIdJxBQJa
jc9QAhsMAAoJEP7vn8LdIdJxwswP/
↪0oGlxUJZhDG8yCbTTTvxvKXd02AXw/GQKrq
ptrLEXbhko6TOuZolEWsRrc1ObMiky97CicqQthg22Kf1K7g2UNlPS4LFtTrPXKL
9iJMAgms0a0ul3cHqQh2XiuGc1bfDuGyNe/
↪nE5/uvgpjxg0hvvBH/5xuiaMkf+gZ
nJjF2ZcXm6a17MCuAcw/siox1/
↪PeXn0At/wzOWD9qONg+BI/QUynzcSMg/coBe7V
hUX1LU02n6laBwuQ6Q0KoD6CP43seYv3JaPyVP7+IkhtH/
↪RDm8q3vs0qLpEBrJIb
vBYBXLtyoGHxTkWueou0Ur1j2lLUMqnQkq5NAsckSfHtZEdPDy6T3NHMfVRmnXnW
m/GM3BDE7DFe5BBYb+vJS4/
↪JHNDoSpk+jNezaf3hdx9+fh2DIoL84fs1FRRAl3Od
6LWPAt3twOQLS0KsQh0GSIZ+zdJf3xvlZ4ixAaPB4iAF8bXYzvsODN3LRQIGhet2
NzjD41f5IrAlG/
↪qFiC6s/
↪YLj1DWanLw2nTzSi4x3v0Gc4DEXPebB3KvaNEmqoKGP
5aXa9IPbvzEVCX82qjeqCPYAsYVOBQnFEAcnkrQ76363oJTeTHxK7kgewS2YCVyy
7wVinR8eyrs+3AWrZ5Op817HgxGvAVDGOEK+1OX9g1wt+IdxX00s85/
↪T+Zk9RF6H
wtRaD9li
=LjIC
-----END PGP PUBLIC KEY BLOCK-----
```

1.49.3 Security issue archive

Release 2.3.0

Version 2.3.0 resolves a security issue with translations on our community hosting site that allowed users to modify the hosted path of a target project by adding it as a translation project of their own project. A check was added to ensure project ownership before adding the project as a translation.

In order to add a project as a translation now, users must now first be granted ownership in the translation project.

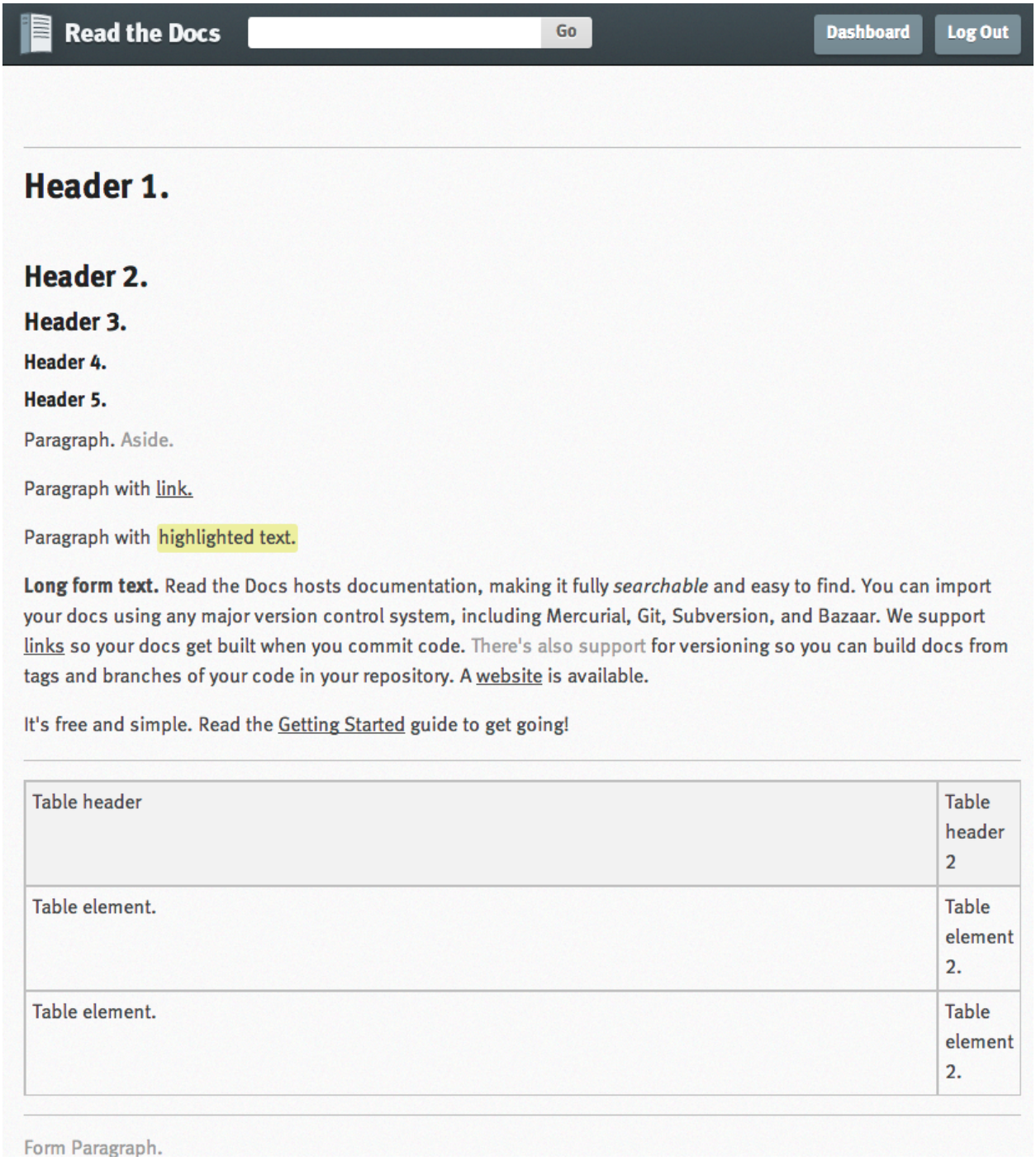
1.50 Designing Read the Docs

So you're thinking of contributing some of your time and design skills to Read the Docs? That's **awesome**. This document will lead you through a few features available to ease the process of working with Read the Doc's CSS and static assets.

To start, you should follow the *Installation* instructions to get a working copy of the Read the Docs repository locally.

1.50.1 Style Catalog

Once you have RTD running locally, you can open `http://localhost:8000/style-catalog/` for a quick overview of the currently available styles.



This way you can quickly get started writing HTML – or if you're modifying existing styles you can get a quick idea of how things will change site-wide.

1.50.2 Readthedocs.org Changes

Styles for the primary RTD site are located in `media/css` directory.

These styles only affect the primary site – **not** any of the generated documentation using the default RTD style.

1.50.3 Contributing

Contributions should follow the *Contributing to Read the Docs* guidelines where applicable – ideally you'll create a pull request against the Read the Docs GitHub project from your forked repo and include a brief description of what you added / removed / changed, as well as an attached image (you can just take a screenshot and drop it into the PR creation form) of the effects of your changes.

There's not a hard browser range, but your design changes should work reasonably well across all major browsers, IE8+ – that's not to say it needs to be pixel-perfect in older browsers! Just avoid making changes that render older browsers utterly unusable (or provide a sane fallback).

1.50.4 Brand Guidelines

Find our branding guidelines in our guidelines documentation: https://read-the-docs-guidelines.readthedocs-hosted.com.

1.51 Read the Docs Commercial Features

Note: These features are for our new commercial offering, readthedocs.com.

All of the other features outlined in these docs work on both sites. Things inside this section are specific to our commercial offering.

The largest feature that is different is that documentation on readthedocs.com is **private**. If you have private code that you want documentation for, this is our solution.

1.51.1 Custom Domains

Subdomain support

Once a project is imported under Read the Docs, by default it's hosted under a subdomain on one of our domains. If

you need a custom domain, continue on custom domain setup.

Custom domains

Projects can also be hosted under a custom domain. If you'd prefer to use your own domain name instead of our default hosting domain, you can still host with us.

We require two steps from your side:

- Add a CNAME record in your DNS that points to our servers `<organization-slug>.users.readthedocs.com`
- Set your project's Privacy Level to *Public* from **Project Admin > Advance Settings**.
- Add a Domain in the **Project Admin > Domains** page for your project.

Note: The domain that should be used is the actual subdomain that you want your docs served on. Generally it will be `docs.projectname.com`.

Custom domain SSL

We do support SSL for CNAMEs on our side. Please, contact our support team to setup it.

Note: SSL is required for *private* projects.

1.51.2 Organizations

Organizations allow you to segment who has access to what projects in your company. Your company will be represented as an Organization, let's use ACME Corporation as our example.

ACME has a few people inside their organization, some who need full access and some who just need access to one project.

Member Types

- **Owners** – Get full access to both view and edit the Organization and all Projects
- **Members** – Get access to a subset of the Organization projects
- **Teams** – Where you give members access to a set of projects.

The best way to think about this relationship is:

Owners will create *Teams* to assign permissions to all *Members*.

Team Types

You can create two types of Teams:

- **Admins** – These teams have full access to administer the projects in the team. They are allowed to change all of the settings, set notifications, and perform any action under the **Admin** tab.
- **Read Only** – These teams are only able to read and search inside the documents.

Example

ACME would set up *Owners* of their organization, for example Frank Roadrunner would be an owner. He has full access to the organization and all projects.

Wile E. Coyote is a contractor, and will just have access to the new project Road Builder.

Roadrunner would set up a *Team* called *Contractors*. That team would have *Read Only* access to the *Road Builder* project. Then he would add *Wile E. Coyote* to the team. This would give him access to just this one project inside the organization.

1.51.3 Sharing

Note: This feature only exists on our Business offering at readthedocs.com.

You can share your project with users outside of your company. There are two way to do this:

- by sending them a *secret link*,
- by giving them a *password*.

These methods will allow them to view a specific project inside your company.

Enabling

- Go into your *Project Admin* page and to the *Sharing* menu.
- Under the *Share with someone new* heading, select the way you prefer (secret link or password), add an expiration date and a *Description* so you remember who you're sharing it with.

- Click *Share!* to create.
- Get the info needed to share your documentation with other users:
  - If you have selected secret link, copy the link that is generated
  - In case of password, copy the link and password
- Give that information to the person who you want to give access.

Note: You can always revoke access in the same panel.

Effects

Secret Link

Once the person you send the link to clicks the link, they will have access to view your project.

Password

Once the person you send the link to clicks on the link, they will see a *Authorization required* page asking them for the password you generated. When the user enters the password, they will have access to view your project.

1.51.4 Analytics

Note: These features are still being developed, and aren't deployed yet.

Analytics lets you see *who* is viewing *which* documents. This allows you to understand how your documentation is being used, so you can focus on expanding and updating parts people are reading most.

Viewing

Each project page has a listing of the number of views that it has seen. You can click through here to inspect more information about who is viewing, and when they are looking at things.

You can also view your Analytics data in your documentation pages. There is a button in the Read the Docs flyout what will overlay analytics information. This will let you understand how users are using docs, in context of the actual documentation.

1.52 Info about custom installs

Read the Docs is open source, which means you can run your own version of it. There are many reasons to do this, the main one being if you want a private instance. If you have to keep everything behind a firewall or VPN, this is for you.

Warning: Read the Docs developers do not support custom installs of our software. These documents are maintained by the community, and might not be up to date.

1.52.1 Customizing your install

Read the Docs has a lot of *Interesting Settings* that help customize your install. This document will outline some of the more useful ways that these can be combined.

Have a local settings file

If you put a file named `local_settings.py` in the `readthedocs/settings` directory, it will override settings available in the base install.

Adding your own title to pages

This requires 2 parts of setup. First, you need to add a custom `TEMPLATE_DIRS` setting that points at your template overrides. Then, in those template overrides you have to insert your logo where the normal RTD logo goes.

Note: This works for any setting you wish to change.

Example `local_settings.py`:

```
import os

# Directory␣
↪that the project lives in, aka ../..
SITE_ROOT = '/'.join(os.
↪path.dirname(__file__).split('/')[0:-2])

TEMPLATE_DIRS = (
    "%s/var/custom_templates/" % SITE_
↪ROOT, # Your custom template directory,
↪ before the RTD one to override it.
    '%s/readthedocs/templates/
↪' % SITE_ROOT, # Default RTD template dir
)
```

Example `base.html` in your template overrides:

```
{% extends␣
↪"/home/docs/checkouts/readthedocs.
↪org/readthedocs/templates/base.html" %}
{% load i18n %}

{% block branding %}{
↪% trans "My sweet site" %} {% endblock %}
```

You can of course override any block in the template. If there is something that you would like to be able to customize, but isn't currently in a block, please submit an issue.

1.52.2 Enabling Elasticsearch on the local server

Read the Docs has been using Elasticsearch for indexing and searching. To enable this on your local installation, you need to install elasticsearch and run the Elastic server locally.

Installation has been mainly divided into following steps.

Installing Java

Elasticsearch requires Java 8 or later. Use Oracle official documentation. or opensource distribution like OpenJDK.

After installing java, verify the installation by,:

```
$ java -version
```

The result should be something like this:

```
openjdk version "1.8.0_151"
OpenJDK Runtime Environment (build␣
↪1.8.0_151-8u151-b12-0ubuntu0.16.04.2-b12)
OpenJDK 64-Bit␣
↪Server VM (build 25.151-b12, mixed mode)
```

Downloading and installing Elasticsearch

Elasticsearch can be downloaded directly from elastic.co. For Ubuntu, it's best to use the deb (Debian) package which will install everything you need to run Elasticsearch.

RTD currently uses elasticsearch 1.x which can be easily downloaded and installed from elastic.co.

Install the downloaded package by following command:

```
$ sudo apt install .{path-to-
↪downloaded-file}/elasticsearch-1.3.8.deb
```

Custom setup

You need the icu plugin:

```
$ elasticsearch/
↪bin/plugin -install elasticsearch/
↪elasticsearch-analysis-icu/2.3.0
```

Running Elasticsearch from command line

Elasticsearch is not started automatically after installation. How to start and stop Elasticsearch depends on whether your system uses SysV init or systemd (used by newer distributions). You can tell which is being used by running this command:

```
$ ps -p 1
```

Running Elasticsearch with SysV init

Use the `update-rc.d command` to configure Elasticsearch to start automatically when the system boots up:

```
$ sudo␣
↪update-rc.d elasticsearch defaults 95 10
```

Elasticsearch can be started and stopped using the service command:

```
$ sudo -i service elasticsearch start
$ sudo -i service elasticsearch stop
```

If Elasticsearch fails to start for any reason, it will print the reason for failure to STDOUT. Log files can be found in /var/log/elasticsearch/.

Running Elasticsearch with systemd

To configure Elasticsearch to start automatically when the system boots up, run the following commands:

```
$ sudo /bin/systemctl daemon-reload
$ sudo /bin/
↪systemctl enable elasticsearch.service
```

Elasticsearch can be started and stopped as follows:

```
$ sudo␣
↪systemctl start elasticsearch.service
$ sudo systemctl stop elasticsearch.service
```

To verify run:

```
$ curl http://localhost:9200
```

You should get something like:

```
{
    status: 200,
    name: "Amina Synge",
    version: {
        number: "1.3.8",
        build_hash:␣
↪"475733ee0837fba18c00c3ee76cd49a08755550c
↪",
    ␣
↪ build_timestamp: "2015-02-11T14:45:42Z",
        build_snapshot: false,
        lucene_version: "4.9"
    },
    tagline: "You Know, for Search"
}
```

Index the data available at RTD database

You need to create the indexes:

```
$ python manage.py provision_elasticsearch
```

In order to search through the RTD database, you need to index it into the elasticsearch index:

```
$ python manage.py reindex_elasticsearch
```

You are ready to go!

1.52.3 Local VM Install

Assumptions and Prerequisites

- Debian VM provisioned with python 2.7.x
- All python dependencies and setup tools are installed

```
$ sudo apt-get install python-setuptools
$ sudo apt-get install build-essential
$ sudo apt-get install python-dev
$ sudo apt-get install libevent-dev
$ sudo easy_install pip
```

- Git

```
$ sudo apt-get install git
```

- Git repo is `git.corp.company.com:git/docs/documentation.git`
- Source documents are in `../docs/source`
- Sphinx

```
$ sudo pip install sphinx
```

Note: Not using sudo may prevent access. "error: could not create '/usr/local/lib/python2.7/dist-packages/markupsafe': Permission denied"

Local RTD Setup

Install RTD

To host your documentation on a local RTD installation, set it up in your VM.

```
$ mkdir checkouts
$ cd checkouts
$ git clone https:/
↪/github.com/rtfd/readthedocs.org.git
$ cd readthedocs.org
$ sudo pip install -r requirements.txt
```

Possible Error and Resolution

Error: `error: command 'gcc' failed with exit status 1`

Resolution: Run the following commands.

```
$ sudo apt-get update
$ sudo apt-get install␣
↪python2.7-dev tk8.5 tcl8.5 tk8.5-dev␣
↪tcl8.5-dev libxml2-devel libxslt-devel
$ sudo apt-get␣
↪build-dep python-imaging --fix-missing
```

On Debian 8 (jessie) the command is slightly different

```
$ sudo apt-get update
$ sudo apt-get␣
↪install python2.7-dev tk8.5 tcl8.5 tk8.
↪5-dev tcl8.5-dev libxml2-dev libxslt-dev
$ sudo apt-get␣
↪build-dep python-imaging --fix-missing
```

Also don't forget to re-run the dependency installation

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

Configure the RTD Server and Superuser

1. Run the following commands.

```
$ ./manage.py migrate
$ ./manage.py createsuperuser
```

2. This will prompt you to create a superuser account for Django. Enter appropriate details. For example:

```
Username: monami.b
Email address: monami.b@email.com
Password: pa$$word
```

RTD Server Administration

Navigate to the `../checkouts/readthedocs.org` folder in your VM and run the following command.

```
$ .
↪/manage.py runserver [VM IP ADDRESS]:8000
$ curl -i http://[VM IP ADDRESS]:8000
```

You should now be able to log into the admin interface from any PC in your LAN at `http://[VM IP ADDRESS]:8000/admin` using the superuser account created in django.

Go to the dashboard at `http://[VM IP ADDRESS]:8000/dashboard` and follow these steps:

1. Point the repository to your corporate Git project where the documentation source is checked in. Example: `git.corp.company.com:/git/docs/documentation.git`.
2. Clone the documentation sources from Git in the VM.
3. Navigate to the root path for documentation.
4. Run the following Sphinx commands.

```
$ make html
```

This generates the HTML documentation site using the default Sphinx theme. Verify the output in your local documentation folder under `../build/html`

Possible Error and Resolution

Error: Couldn't access Git Corp from VM.

Resolution: The primary access may be set from your base PC/laptop. You will need to configure your RSA keys in the VM.

Workaround-1

1. In your machine, navigate to the `.ssh` folder.

```
$ cd .ssh/
$ cat id_rsa
```

2. Copy the entire Private Key.
3. Now, SSH to the VM.
4. Open the `id_rsa` file in the VM.

```
$ vim /home/<username>/.ssh/id_rsa
```

5. Paste the RSA key copied from your machine and save file (`Esc. :wq!`).

Workaround 2

SSH to the VM using the `-A` directive.

```
$ ssh document-vm -A
```

This provides all permissions for that particular remote session, which are revoked when you logout.

Build Documentation on Local RTD Instance

Log into `http://[VM IP ADDRESS]:[PORT]` using the django superuser creds and follow these steps.

For a new project

1. Select **<username> > Add Project** from the user menu.
2. Click **Manually Import Project**.
3. Provide the following information in the **Project Details** page:

- **Name**: Appropriate name for the documentation project. For example – API Docs Project
- **Repository URL**: URL to the documentation project. For example - git.corp.company.com:/git/docs/documentation.git
- **Repository Type**: Git

4. Select the **Edit advanced project options** checkbox.
5. Click **Next**.

For an existing project

1. Select **<username> > Projects** from the user menu.
2. Select the relevant project from the **Projects** list.
3. Select latest from the **Build a version** dropdown.

4. Click **Build**. This will take you to the Builds tab where the progress status is displayed. This may take some time.

Tips

- If the installation doesn't work on VM using your login/LDAP credentials, try running the operations as root (su).

HTTP Routing Table

/api

Index