
Read the Docs Documentation

Release 4.0.3

Eric Holscher, Charlie Leifer, Bobby Grace

Mar 27, 2020

1	First steps	3
1.1	Getting Started with Sphinx	3
1.2	Getting Started with MkDocs	5
1.3	Importing Your Documentation	5
2	Getting started with Read the Docs	9
2.1	Read the Docs features	9
2.2	Configuration File	11
2.3	Webhooks	23
2.4	Badges	26
2.5	Custom Domains	27
2.6	Connecting Your Account	29
2.7	Build Process	30
2.8	Versions	33
2.9	Support	34
2.10	Frequently Asked Questions	35
3	Advanced features of Read the Docs	41
3.1	Subprojects	41
3.2	Single Version Documentation	42
3.3	Privacy Levels	42
3.4	Localization of Documentation	43
3.5	User-defined Redirects	44
3.6	Automatic Redirects	46
3.7	Automation Rules	48
3.8	Guides	50
3.9	Public API	79
4	The Read the Docs project and organization	107
4.1	Contributing to Read the Docs	107
4.2	Roadmap	152
4.3	Google Summer of Code	154
4.4	Code of Conduct	157
4.5	Security	158
4.6	Privacy Policy	160
4.7	Read the Docs Terms of Service	165
4.8	DMCA Takedown Policy	176

4.9	Policy for Abandoned Projects	178
4.10	Changelog	179
4.11	Read the Docs Team	238
4.12	Read the Docs Open Source Philosophy	240
4.13	The Story of Read the Docs	241
4.14	Advertising	241
4.15	Sponsors of Read the Docs	247
4.16	Read the Docs for Business	248
4.17	Info about custom installs	251
HTTP Routing Table		257
Index		259

[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.

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](#).

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`:

```
extensions = ['recommonmark']
```

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](#)

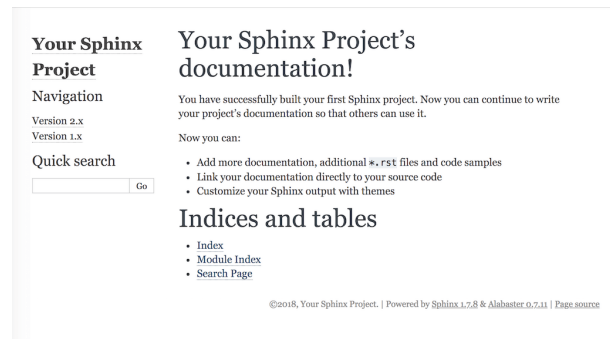


Fig. 1: Your Sphinx project is built

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*.

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](#)

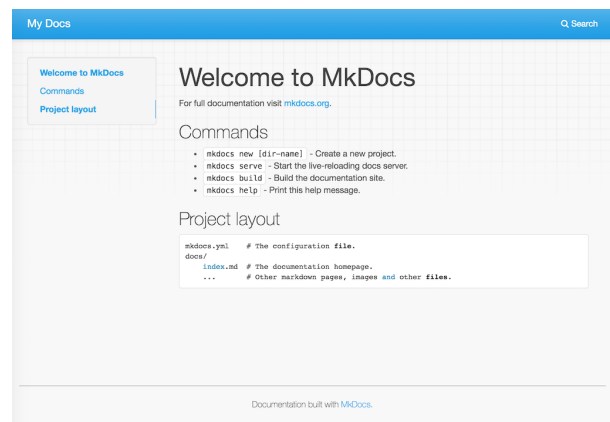


Fig. 2: Your MkDocs project is built

1.3 Importing Your Documentation

To import a public documentation repository, visit your [Read the Docs dashboard](#) and click **Import**. For private

repositories, please use *Read the Docs for Business*.

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**.

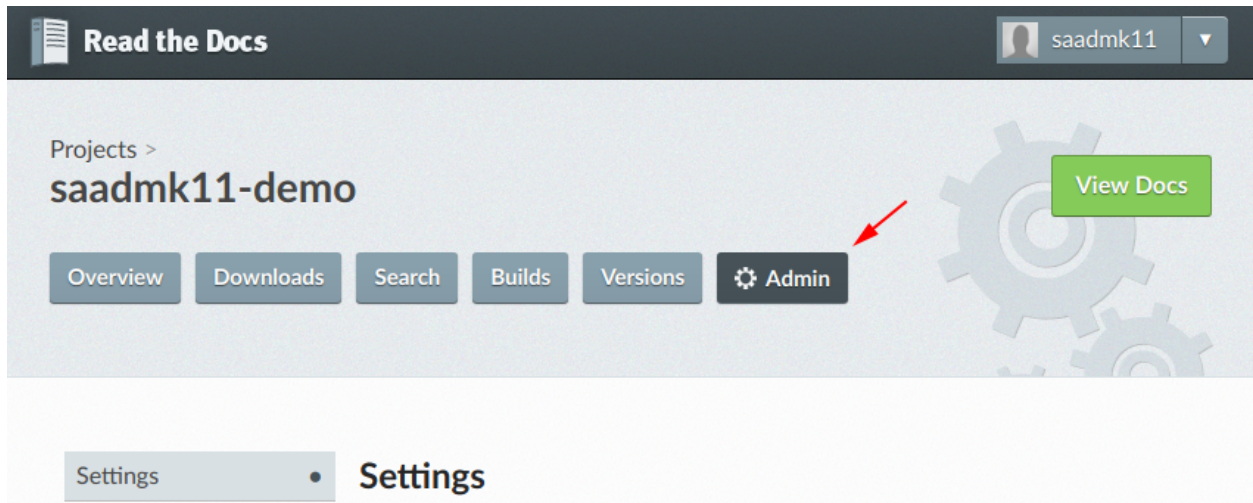
Once your project is created, you'll need to manually configure the repository webhook if you would like to have new changes trigger builds for your project on Read the Docs. Go to your project's *Admin > Integrations* page to configure a new webhook, or see *our steps for webhook creation* for more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

Note: The Admin page can be found at `https://readthedocs.org/dashboard/<project-slug>/edit/`. You can access all of the project settings from the admin page sidebar.

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 [Configuration File](#) docs for more details.

It is also important to note that the default version of Sphinx is `v1.8.5`. If choosing to build your documentation other than this, it must be specified in a `requirements.txt` file.

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.

Getting started with Read the Docs

Learn more about configuring your automated documentation builds and some of the core features of Read the Docs.

- **Overview of core features:** [Read the Docs features](#)
- **Configure your documentation:** [Configuration reference](#) | [Webhooks](#) | [Badges](#) | [Custom domains](#)
- **Connecting with GitHub, BitBucket, or GitLab:** [Connecting your account](#)
- **Read the Docs build and versioning process:** [Build process](#) | [Handling multiple docs versions](#)
- **Troubleshooting:** [Support](#) | [Frequently asked questions](#)

2.1 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.

2.1.1 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 [Version Control System Integration](#).

2.1.2 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.

2.1.3 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.

2.1.4 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.

2.1.5 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.rtd.org/>).

2.1.6 Version Control Support Matrix

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

2.1.7 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.

2.1.8 Search

We provide full-text search across all of the pages of documentation hosted on our site. This uses Elasticsearch as the search backend. We hope to be integrating this into the site more fully in the future.

2.1.9 Alternate Domains

We provide support for custom domains, subdomains, and a shorturl for your project as well. This is outlined in the [Custom Domains](#) section.

2.2 Configuration File

In addition to using the admin panel of your project to configure your project, you can use a configuration file in the root of your project. The configuration file can be named as:

- `readthedocs.yml`
- `readthedocs.yaml`
- `.readthedocs.yml`
- `.readthedocs.yaml`

The main advantages of using a configuration file over the web interface are:

- Some settings are only available using a configuration file
- The settings are per version rather than per project.
- The settings live in your VCS.
- Reproducible build environments over time.

Tip: Using a configuration file is the recommended way of using Read the Docs.

2.2.1 Configuration File V2

Read the Docs supports configuring your documentation builds with a YAML file. *The Read the Docs file* must be in the root directory of your project.

Below is an example YAML file which may require some changes for your project's configuration:

```
# .readthedocs.yml
# Read the Docs configuration file
# See https://docs.readthedocs.io/en/stable/config-file/v2.html for details

# Required
version: 2

# Build documentation in the docs/ directory with Sphinx
sphinx:
  configuration: docs/conf.py

# Build documentation with MkDocs
#mkdocs:
#  configuration: mkdocs.yml

# Optionally build your docs in additional formats such as PDF and ePub
formats: all

# Optionally set the version of Python and requirements required to build your docs
python:
  version: 3.7
  install:
    - requirements: docs/requirements.txt
```

Supported settings

Note: The presence of any other key that isn't documented here will make the build to fail. This is to avoid typos and provide feedback on invalid configurations.

version

Required `true`

Example:

```
version: 2
```

Warning: If you don't provide the version, `v/` will be used.

formats

Formats of the documentation to be built.

Type `list`

Options `htmlzip, pdf, epub`

Default `[]`

Example:

```
# Default
formats: []
```

```
# Build PDF & ePub
formats:
  - epub
  - pdf
```

Note: You can use the `all` keyword to indicate all formats.

```
# Build all formats
formats: all
```

Note: PDF/epub/htmlzip output is not supported when using MkDocs

python

Configuration of the Python environment to be used. Example:


```
python:
  version: 3.7
  install:
    - requirements: docs/requirements.txt
    - method: pip
      path: .
      extra_requirements:
        - docs
    - method: setuptools
      path: another/package
  system_packages: true
```

python.version

The Python version (this depends on *build.image*).

Type number

Default 3

Warning: If you are using a *Conda* environment to manage the build, this setting will not have any effect, as the Python version is managed by Conda.

python.install

List of installation methods of packages and requirements. You can have several of the following methods.

Type list

Default []

Requirements file

Install packages from a requirements file.

The path to the requirements file, relative to the root of the project.

Key requirements

Type path

Required true

Example:

```
python:
  version: 3.7
  install:
    - requirements: docs/requirements.txt
    - requirements: requirements.txt
```

Warning: If you are using a *Conda* environment to manage the build, this setting will not have any effect. Instead add the extra requirements to the `environment` file of Conda.

Packages

Install the project using `python setup.py install` or `pip install`.

The path to the package, relative to the root of the project.

Key `path`

Type `path`

Required `true`

The installation method.

Key `method`

Options `pip, setuptools`

Default `pip`

[Extra requirements](#) section to install in addition to the [package dependencies](#).

Warning: You need to install your project with `pip` to use `extra_requirements`.

Key `extra_requirements`

Type `list`

Default `[]`

Example:

```
python:
  version: 3.7
  install:
    - method: pip
      path: .
      extra_requirements:
        - docs
    - method: setuptools
      path: package
```

With the previous settings, Read the Docs will execute the next commands:

```
$ pip install .[docs]
$ python package/setup.py install
```

`python.system_packages`

Give the virtual environment access to the global site-packages directory.

Type `bool`

Default `false`

Depending on the [build.image](#), Read the Docs includes some libraries like `scipy`, `numpy`, etc. That you can access to them by enabling this option. See [The build environment](#) for more details.

Warning: If you are using a *Conda* environment to manage the build, this setting will not have any effect, since the virtual environment creation is managed by Conda.

conda

Configuration for Conda support. Example:

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

conda.environment

The path to the Conda environment file, relative to the root of the project.

Type path

Required true

build

Configuration for the documentation build process. Example:

```
build:
  image: latest

python:
  version: 3.7
```

build.image

The Docker image used for building the docs.

Type string

Options stable, latest

Default latest

Each image support different Python versions and has different packages installed, as defined here:

- **stable:** 2, 2.7, 3, 3.5, 3.6, 3.7, pypy3.5
- **latest:** 2, 2.7, 3, 3.5, 3.6, 3.7, 3.8, pypy3.5

sphinx

Configuration for Sphinx documentation (this is the default documentation type). Example:

```
sphinx:
  builder: html
  configuration: conf.py
  fail_on_warning: true
```

sphinx.builder

The builder type for the Sphinx documentation.

Type `string`

Options `html, dirhtml, singlehtml`

Default `html`

Note: The `htmldir` builder option was renamed to `dirhtml` to use the same name as sphinx. Configurations using the old name will continue working.

sphinx.configuration

The path to the `conf.py` file, relative to the root of the project.

Type `path`

Default `null`

If the value is `null`, Read the Docs will try to find a `conf.py` file in your project.

sphinx.fail_on_warning

Turn warnings into errors. This means that the build stops at the first warning and exits with exit status 1.

Type `bool`

Default `false`

mkdocs

Configuration for Mkdocs documentation. Example:

```
mkdocs:
  configuration: mkdocs.yml
  fail_on_warning: false
```

mkdocs.configuration

The path to the `mkdocs.yml` file, relative to the root of the project.

Type `path`

Default `null`

If the value is `null`, Read the Docs will try to find a `mkdocs.yml` file in your project.

mkdocs.fail_on_warning

Turn warnings into errors. This means that the build stops at the first warning and exits with exit status 1.

Type bool

Default false

submodules

VCS submodules configuration.

Note: Only Git is supported at the moment.

Note: You can't use `include` and `exclude` settings for submodules at the same time.

Example:

```
submodules:
  include:
    - one
    - two
  recursive: true
```

submodules.include

List of submodules to be included.

Type list

Default []

Note: You can use the `all` keyword to include all submodules.

```
submodules:
  include: all
```

submodules.exclude

List of submodules to be excluded.

Type list

Default []

Note: You can use the `all` keyword to exclude all submodules. This is the same as `include: []`.

```
submodules:
  exclude: all
```

submodules.recursive

Do a recursive clone of the submodules.

Type `bool`

Default `false`

Note: This is ignored if there aren't submodules to clone.

Schema

You can see the complete schema [here](#).

Migrating from v1

Changes

- The version setting is required. See [version](#).
- The default value of the [formats](#) setting has changed to `[]` and it doesn't include the values from the web interface.
- The top setting `requirements_file` was moved to `python.install` and we don't try to find a requirements file if the option isn't present. See [Requirements file](#).
- The setting `conda.file` was renamed to `conda.environment`. See [conda.environment](#).
- The `build.image` setting now only has two options: `latest` (default) and `stable`. See [build.image](#).
- The settings `python.setup_py_install` and `python.pip_install` were replaced by `python.install`. And now it accepts a path to the package. See [Packages](#).
- The setting `python.use_system_site_packages` was renamed to `python.system_packages`. See [python.system_packages](#).
- The build will fail if there are invalid keys (strict mode).

Warning: Some values from the web interface are no longer respected, please see [Migrating from the web interface](#) if you have settings there.

New settings

- [sphinx](#)
- [mkdocs](#)
- [submodules](#)
- [python.install](#)

Migrating from the web interface

This should be pretty straightforward, just go to the *Admin > Advanced settings*, and find their respective setting in [here](#).

Not all settings in the web interface are per version, but are per project. These settings aren't supported via the configuration file.

- Name
- Repository URL
- Repository type
- Language
- Programming language
- Project homepage
- Tags
- Single version
- Default branch
- Default version
- Show versions warning
- Privacy level
- Analytics code

2.2.2 Configuration File V1

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

Warning: Version 1 shouldn't be used. The version 2 of the configuration file is now available. See the [new features](#) and [how to migrate from v1](#).

Here is an example of what this file looks like:

```
# .readthedocs.yml

build:
  image: latest

python:
  version: 3.6
  setup_py_install: true
```

Supported settings

version

- Default: 1

```
version: 1
```

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:
  - 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: `latest`
- Options: `stable`, `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:

- `stable`: 2, 2.7, 3, 3.5, 3.6, 3.7, pypy3.5
- `latest`: 2, 2.7, 3, 3.5, 3.6, 3.7, 3.8, pypy3.5

```
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: `3.7`
- Options: 2, 2.7, 3, 3.5, 3.6, 3.7, 3.8, pypy3.5

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 3.7. See [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',
```

(continues on next page)

(continued from previous page)

```
'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]
```

2.3 Webhooks

The primary method that Read the Docs uses to detect changes to your documentation and versions 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.

2.3.1 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.

2.3.2 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, go to *Admin > Integrations > Add integration* 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: `https://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.

Note: If your account is connected to the provider, we'll try to setup the webhook automatically. If something fails, you can still setup the webhook manually.

GitHub

- Go to the *Settings* page for your project
- Click *Webhooks > Add webhook*
- For **Payload URL**, use the URL of the integration on Read the Docs, found on the project's *Admin > Integrations* page. You may need to prepend *https://* to the URL.
- For **Content type**, both *application/json* and *application/x-www-form-urlencoded* work
- Leave the **Secrets** field blank
- Select **Let me select individual events**, and mark **Branch or tag creation**, **Branch or tag deletion** and **Pushes** events
- Ensure **Active** is enabled; it is by default
- Finish by clicking **Add webhook**. You may be prompted to enter your GitHub password to confirm your action.

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.

GitHub will emit an initial HTTP request (`X-GitHub-Event: ping`) upon creating the webhook and you may notice that the Read the Docs responds with `{"detail": "Unhandled webhook event"}` – this is normal and expected. Push changes to your repository and webhooks will work from this point.

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

Bitbucket

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

GitLab

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

2.3.3 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, which can be found on the project's **Integrations** dashboard page (*Admin > Integrations*).

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 found on the project's **Integrations** dashboard page (*Admin > Integrations*).

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.

2.3.4 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.

2.3.5 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.

2.3.6 Payload validation

If your project was imported through a connected account, we create a secret for every integration that offers a way to verify that a webhook request is legitimate. Currently, [GitHub](#) and [GitLab](#) offer a way to check this.

2.3.7 Troubleshooting

My project isn't automatically building

If your project isn't automatically building, you can check your integration on Read the Docs to see the payload sent to our servers. If there is no recent activity on your Read the Docs project webhook integration, then it's likely that your VCS provider is not configured correctly. If there is payload information on your Read the Docs project, you might need to verify that your versions are configured to build correctly.

Either way, it may help to either resync your webhook integration (see [Resyncing webhooks](#) for information on this process), or set up an entirely new webhook integration.

I was warned I shouldn't use GitHub Services

Last year, GitHub announced that effective Jan 31st, 2019, GitHub Services will stop working¹. This means GitHub will stop sending notifications to Read the Docs for projects configured with the `ReadTheDocs` GitHub Service. If your project has been configured on Read the Docs for a long time, you are most likely still using this service to automatically build your project on Read the Docs.

In order for your project to continue automatically building, you will need to configure your GitHub repository with a new webhook. You can use either a connected GitHub account and a *GitHub webhook integration* on your Read the Docs project, or you can use a *generic webhook integration* without a connected account.

I was warned that my project won't automatically build after April 1st

In addition to *no longer supporting GitHub Services*, we have decided to no longer support several other legacy incoming webhook endpoints that were used before we introduced project webhook integrations. When we introduced our webhook integrations, we added several features and improved security for incoming webhooks and these features were not added to our legacy incoming webhooks. New projects have not been able to use our legacy incoming webhooks since, however if you have a project that has been established for a while, you may still be using these endpoints.

After March 1st, 2019, we will stop accepting incoming webhook notifications for these legacy incoming webhooks. Your project will need to be reconfigured and have a webhook integration configured, pointing to a new webhook with your VCS provider.

In particular, the incoming webhook URLs that will be removed are:

- `https://readthedocs.org/build`
- `https://readthedocs.org/bitbucket`
- `https://readthedocs.org/github` (as noted *above*)
- `https://readthedocs.org/gitlab`

In order to establish a new project webhook integration, *follow the directions for your VCS provider*

2.4 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.

2.4.1 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 <https://developer.github.com/changes/2018-04-25-github-services-deprecation/>

2.4.2 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	

2.4.3 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/
```

2.5 Custom Domains

Note: These directions are for projects hosted on our community site. If you want to setup a custom domain on [Read the Docs for Business](#), please read our *commercial documentation*.

Read the Docs supports a number of custom domains for your convenience. Shorter URLs make everyone happy, and we like making people happy!

2.5.1 Subdomain Support

Every project has a subdomain that is available to serve its documentation. If you go to `<slug>.readthedocs.io`, it should show you the latest version of documentation. A good example is <https://pip.readthedocs.io>

Note: If you have an old project that has an underscore (`_`) in the name, it will use a subdomain with a hyphen (`-`). [RFC 1035](#) has more information on valid subdomains.

2.5.2 Custom Domain Support

You can also host your documentation from your own domain by adding a domain to your project:

- Add a CNAME record in your DNS that points the domain to: `readthedocs.io`
- Add a project domain in the *Domains* project admin page for your project.

Note: We don't currently support pointing subdomains or naked domains to a project using A records. It's best to point a subdomain, `docs.example.com` for example, using a CNAME record.

Using `pip` as an example, <https://pip.pypa.io> resolves, but is hosted on our infrastructure.

As another example, `fabric`'s dig record looks like this:

```
-> dig docs.fabfile.org
...
;; ANSWER SECTION:
docs.fabfile.org. 7200 IN CNAME readthedocs.io.
```

2.5.3 Custom Domain SSL

By default, when you setup a custom domain to host documentation at Read the Docs, we will attempt to provision a domain validated SSL certificate for the domain. This service is generously provided by Cloudflare.

After configuring your custom domain on Read the Docs, you can see the status of the certificate on the domain page in your project admin dashboard (*Domains > Edit Domain*).

If your domain has configured CAA records, please do not forget to include Cloudflare CAA entries, see their [Certification Authority Authorization \(CAA\) FAQ](#).

Note: Some older setups configured a CNAME record pointing to `readthedocs.org` or another variation. While these continue to resolve, they do not yet allow us to acquire SSL certificates for those domains. Point the CNAME to `readthedocs.io`, with no subdomain, and re-request a certificate by saving the domain in the project admin (*Domains > Edit Domain*).

If you change the CNAME record, the SSL certificate issuance can take about one hour.

Important: Due to a limitation, a domain cannot be proxied on Cloudflare to another Cloudflare account that also proxies. This results in a “CNAME Cross-User Banned” error. In order to do SSL termination, we must proxy this connection. If you don't want us to do SSL termination for your domain – **which means you are responsible for the SSL certificate** – then set your CNAME to `cloudflare-to-cloudflare.readthedocs.io` instead of `readthedocs.io`.

For more details, see this [previous issue](#).

2.5.4 Proxy SSL

If you would prefer to do your own SSL termination on a server you own and control, you can do that although the setup is a bit more complex.

Broadly, the steps are:

- Have a server listening on 443 that you control
- Procure an SSL certificate for your domain and provision it and the private key on your server.

- Add a domain that you wish to point at Read the Docs
- Enable proxying to us, with a custom X-RTD-SLUG header

An example nginx configuration for pip would look like:

```
server {
    server_name pip.pypa.io;
    location / {
        proxy_pass https://pip.readthedocs.io:443;
        proxy_set_header Host $http_host;
        proxy_set_header X-Forwarded-Proto https;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Scheme $scheme;
        proxy_set_header X-RTD-SLUG pip;
        proxy_connect_timeout 10s;
        proxy_read_timeout 20s;
    }
}
```

2.5.5 rtdf.org

You can also use `rtdf.io` and `rtdf.org` for short URLs for Read the Docs. For example, <https://pip.rtdf.io> redirects to its documentation page. Any use of `rtdf.io` or `rtdf.org` will simply be redirected to `readthedocs.io`.

2.6 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.

2.6.1 Permissions for connected accounts

Read the Docs does not generally ask for write permission to your repositories' code (with one exception detailed below) and since we only connect to public repositories we don't need special permissions to read them. However, we do need permissions for authorizing your account so that you can login to Read the Docs with your connected account credentials and to setup *Webhooks* which allow us to build your documentation on every change to your repository.

GitHub

Read the Docs requests the following permissions (more precisely, [OAuth scopes](#)) when connecting your Read the Docs account to GitHub.

Read access to your email address (`user:email`) We ask for this so you can create a Read the Docs account and login with your GitHub credentials.

Administering webhooks (`admin:repo_hook`) We ask for this so we can create webhooks on your repositories when you import them into Read the Docs. This allows us to build the docs when you push new commits.

Read access to your organizations (`read:org`) We ask for this so we know which organizations you have access to. This allows you to filter repositories by organization when importing repositories.

Repository status (`repo:status`) Repository statuses allow Read the Docs to report the status (eg. passed, failed, pending) of pull requests to GitHub. This is used for a feature currently in beta testing that builds documentation on each pull request similar to a continuous integration service.

Note: *Read the Docs for Business* asks for one additional permission (`repo`) to allow access to private repositories and to allow us to setup SSH keys to clone your private repositories. Unfortunately, this is the permission for read/write control of the repository but there isn't a more granular permission that only allows setting up SSH keys for read access.

Bitbucket

For similar reasons to those above for GitHub, we request permissions for:

- Reading your account information including your email address
- Read access to your team memberships
- Read access to your repositories
- Read and write access to webhooks

GitLab

Like the others, we request permissions for:

- Reading your account information (`read_user`)
- API access (`api`) which is needed to create webhooks in GitLab

2.7 Build Process

Files: [tasks.py](#) - [doc_builder/](#)

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

- 15 minutes of CPU
- 1GB of RAM memory

We can increase build limits on a per-project basis, sending an email to support@readthedocs.org providing a good reason why your documentation needs more resources. If your business is hitting build limits hosting documentation on Read the Docs, please consider *Read the Docs for Business* which has much higher build 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:latest` docker image as our default image.

2.7.1 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.

2.7.2 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 project's configuration.

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

At this point, if you need extra requirements, or even install your own package in the virtual environment to build your documentation, you can use a [Configuration File](#).

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)
config = get_config(project)
if config.python.install.method.setuptools:
    run('python setup.py install')
if config.python.install.method.pip:
    run('pip install .')
if config.python.install.requirement:
    run('pip install -r %s' % config.python.install.requirement)
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`.

2.7.3 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.

2.7.4 The build environment

The build process is executed inside Docker containers, by default the image used is `readthedocs/build:latest`, 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](#).

More details on software installed in images could be found by browsing specific branch in [rtfd/readthedocs-docker-images](#) repository.

2.7.5 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)
```

2.7.6 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.

2.7.7 Build environment

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

Table 1: :header-rows: 1

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 slug of the project which is being built	my-example-project
READTHEDOCS_LANGUAGE	The RTD language slug of the project which is being built	en

Tip: In case extra environment variables are needed to the build process (like secrets, tokens, etc), you can add them going to *Admin > Environment Variables* in your project. See *I Need Secrets (or Environment Variables) in my Build*.

2.8 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. If you want a custom `stable` version, create either a tag or branch in your project with that name.

2.8.1 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.

2.8.2 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.

2.8.3 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.

2.8.4 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 (*Admin > Advanced Settings*).

2.9 Support

2.9.1 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?”

2.9.2 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

Specific Requests

If you need a specific request for your project or account, like more resources, change of the project's slug or username. Send an email to support@readthedocs.org.

2.9.3 Commercial Support

We offer commercial support with [Read the Docs for Business](#) and we have a dedicated team that responds to support requests during business hours.

For consulting services around documentation systems, you can [contact us](#) or read more at <https://readthedocs.com/services/#open-source-support>.

2.10 Frequently Asked Questions

2.10.1 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](#).

2.10.2 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](#).

If that isn't enough, you can request the change sending an email to support@readthedocs.org.

2.10.3 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 request the change sending an email to support@readthedocs.org.

2.10.4 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).

2.10.5 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 %}
```

2.10.6 My project requires different settings than those available under Admin

Read the Docs offers some settings which can be used for a variety of purposes, such as to use the latest version of sphinx or pip. To enable these settings, please send an email to support@readthedocs.org and we will change the settings for the project. Read more about these settings [here](#).

2.10.7 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. 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.

2.10.8 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'
```

2.10.9 Deleting a stale or broken build environment

See *Wiping a Build Environment*.

2.10.10 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:

<https://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.

2.10.11 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.

2.10.12 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.

2.10.13 I want to use the Read the Docs theme locally

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

2.10.14 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.

2.10.15 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.

2.10.16 How do I support multiple languages of documentation?

See the section on *Localization of Documentation*.

2.10.17 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.

Note: To use these extensions you need to specify the dependencies on your project by following this [guide](#).

2.10.18 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.

2.10.19 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/readthedocs/readthedocs.org/commits/rel>

2.10.20 How can I avoid search results having a deprecated version of my docs?

If readers search something related to your docs in Google, it will probably return the most relevant version of your documentation. It may happen that this version is already deprecated and you want to stop Google indexing it as a result, and start suggesting the latest (or newer) one.

To accomplish this, you can add a `robots.txt` file to your documentation's root so it ends up served at the root URL of your project (for example, <https://yourproject.readthedocs.io/robots.txt>).

Minimal example of `robots.txt`

```
User-agent: *  
Disallow: /en/deprecated-version/  
Disallow: /en/2.0/
```

Note: See [Google's docs](#) for its full syntax.

This file has to be served as is under `/robots.txt`.

Setup

The `robots.txt` file will be served from the **default version** of your Project. This is because the `robots.txt` file is served at the top-level of your domain, so we must choose a version to find the file in. The **default version** is the best place to look for it.

Sphinx and Mkdocs both have different ways of outputting static files in the build:

Sphinx

Sphinx uses `html_extra_path` option to add static files to the output. You need to create a `robots.txt` file and put it under the path defined in `html_extra_path`.

MkDocs

MkDocs needs the `robots.txt` to be at the directory defined at `docs_dir` config.

Advanced features of Read the Docs

Read the Docs offers many advanced features and options. Learn more about these integrations and how you can get the most out of your documentation and Read the Docs.

- **Advanced project configuration:** *Subprojects* | *Single version docs* | *Privacy levels*
- **Multi-language documentation:** *Translations and localization*
- **Redirects:** *User defined redirects* | *Automatic redirects*
- **Versions** *Automation rules*
- **Topic specific guides:** *How-to guides*
- **Extending Read the Docs:** *REST API*

3.1 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/>

3.1.1 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.

3.1.2 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: <https://docs.example.com/en/latest/> and <https://docs.example.com/projects/bar/en/latest/>

3.1.3 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.

3.2 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>

3.2.1 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.

3.2.2 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.

3.3 Privacy Levels

Note: For private documentation or docs from private repositories, use *Read the Docs for Business*.

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

3.3.1 Understanding the Privacy Levels

Level	Detail	Listing	Search	Viewing
Private	No	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.

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.

3.4 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

3.4.1 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.

3.4.2 Project with multiple translations

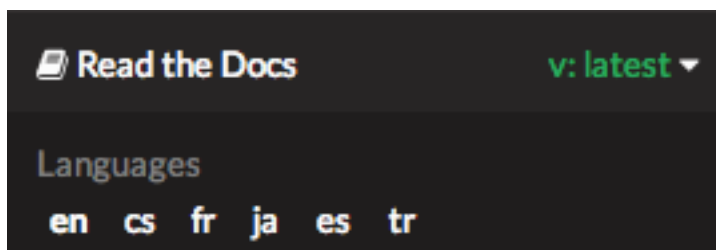
This situation is a bit more complicated. To support this, you will have one parent project and a number of projects marked as translations of that parent. Let's use `phpmyadmin` as an example.

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 [Custom 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.

3.5 User-defined Redirects

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

3.5.1 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.

3.5.2 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 `https://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 `https://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 `https://docs.example.com/dev/install.html` to a new URL of `https://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.

3.5.3 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.

3.6 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.

3.6.1 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.

3.6.2 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.

3.6.3 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
```

(continues on next page)

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.

3.7 Automation Rules

Automation rules allow project maintainers to automate actions on new branches and tags on repositories.

3.7.1 Creating an automation rule

1. Go to your project dashboard
2. Click *Admin > Automation Rules*
3. Click on *Add Rule*
4. Fill in the fields
5. Click *Save*

3.7.2 How do they work?

When a new tag or branch is pushed to your repository, Read the Docs creates a new version.

All rules are evaluated for this version, in the order they are listed. If the version matches the version type and the pattern in the rule, the specified action is performed on that version.

Note: Versions can match multiple rules, and all matching actions will be performed on the version.

3.7.3 Predefined matches

Automation rules support several predefined version matches:

- **Any version:** All new versions will match the rule.
- **SemVer versions:** All new versions that follow [semantic versioning](#) will match the rule.

3.7.4 User defined matches

If none of the above predefined matches meet your use case, you can use a **Custom match**.

The custom match should be a valid [Python regular expression](#). Each new version will be tested against this regular expression.

3.7.5 Actions

When a rule matches a new version, the specified action is performed on that version. Currently, the following actions are available:

- **Activate version:** Activates and builds the version.
- **Set version as default:** Sets the version as default, i.e. the version of your project that / redirects to. See more in [Root URL](#). It also activates and builds the version.

Note: If your versions follow [PEP 440](#), Read the Docs activates and builds the version if it's greater than the current stable version. The stable version is also automatically updated at the same time. See more in [Versions](#).

3.7.6 Order

The order your rules are listed in *Admin > Automation Rules* matters. Each action will be performed in that order, so first rules have a higher priority.

You can change the order using the up and down arrow buttons.

Note: New rules are added at the end (lower priority).

3.7.7 Examples

Activate all new tags

- Match: `Any version`
- Version type: `Tag`
- Action: `Activate version`

Activate only new branches that belong to the 1.x release

- Custom match: `^1\.\d+$`
- Version type: `Branch`
- Action: `Activate version`

Set as default new tags that have the `-stable` or `-release` suffix

- Custom match: `-(stable|release)$`
- Version type: `Tag`
- Action: `Set version as default`

Note: You can also create two rules: one to match `-stable` and other to match `-release`.

Activate all new tags and branches that start with `v` or `V`

- Custom match: `^[vV]`
- Version type: Tag
- Action: `Activate version`
- Custom match: `^[vV]`
- Version type: Branch
- Action: `Activate version`

Activate all new tags that don't contain the `-nightly` suffix

- Custom match: `.*(?<!\-nightly)$`
- Version type: Tag
- Action: `Activate version`

3.8 Guides

These guides will help walk you through specific use cases related to Read the Docs itself, documentation tools like Sphinx and MkDocs and how to write successful documentation.

3.8.1 Sphinx & MkDocs how-to guides

These guides will help you get the most out of your documentation authoring tool whether that is Sphinx or MkDocs.

Adding Custom CSS or JavaScript to Sphinx Documentation

Adding additional CSS or JavaScript files to your Sphinx documentation can let you customize the look and feel of your docs or add additional functionality. For example, with a small snippet of CSS, your documentation could use a custom font or have a different background color.

If your custom stylesheet is `_static/css/custom.css`, you can add that CSS file to the documentation using the Sphinx option `html_css_files`:

```
## conf.py

# These folders are copied to the documentation's HTML output
html_static_path = ['_static']

# These paths are either relative to html_static_path
# or fully qualified paths (eg. https://...)
html_css_files = [
    'css/custom.css',
]
```

A similar approach can be used to add JavaScript files:

```
html_js_files = [
    'js/custom.js',
]
```

Note: The Sphinx HTML options `html_css_files` and `html_js_files` were added in Sphinx 1.8. Unless you have a good reason to use an older version, you are strongly encouraged to upgrade. Sphinx is almost entirely backwards compatible.

Overriding or replacing a theme’s stylesheet

The above approach is preferred for adding additional stylesheets or JavaScript, but it is also possible to completely replace a Sphinx theme’s stylesheet with your own stylesheet.

If your replacement stylesheet exists at `_static/css/yourtheme.css`, you can replace your theme’s CSS file by setting `html_style` in your `conf.py`:

```
## conf.py

html_style = 'css/yourtheme.css'
```

If you only need to override a few styles on the theme, you can include the theme’s normal CSS using the CSS `@import` rule.

```
/** css/yourtheme.css */

/* This line is theme specific - it includes the base theme CSS */
@import '../alabaster.css'; /* for Alabaster */
/*@import 'theme.css';      /* for the Read the Docs theme */

body {
    /* ... */
}
```

Use a Custom 404 Not Found Page on my Project

If you want your project to use a custom page for not found pages instead of the “Maze Found” default one, you can put a `404.html` at the top level of your project’s HTML output.

When a 404 is returned, Read the Docs checks if there is a `404.html` in the root of your project’s output and uses it if it exists.

As the `404.html` will be returned for all the URLs where the real page was not found, all its resources URLs and links must be absolute (starting with a `/`), otherwise they will not work when a user clicks on them.

In case you don’t want to deal with these links manually, or you want to use the same style as your theme, you can use the [sphinx-notfound-page](#) extension.

Using `sphinx-notfound-page` extension

The `sphinx-notfound-page` extension automatically creates the proper URLs for your 404 page. Once the extension is installed, it will generate the default 404 page for your project. See its [documentation](#) for how to install and custom it.

Link to Other Projects' Documentation With Intersphinx

You may be familiar with using the `:ref: role` to link to any location of your docs. It helps you to keep all links within your docs up to date and warns you if a reference target moves or changes so you can ensure that your docs don't have broken cross-references.

Sometimes you may need to link to a specific section of another project's documentation. While you could just hyperlink directly, there is a better way. **Intersphinx** allows you to use all **cross-reference roles** from Sphinx with objects in other projects. That is, you could use the `:ref:` role to link to sections of other documentation projects. Sphinx will ensure that your cross-references to the other project exist and will raise a warning if they are deleted or changed so you can keep your docs up to date.

Note: You can also use Sphinx's `linkcheck` builder to check for broken links. By default it will also check the validity of `#anchors` in links.

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

See all the [options for the linkcheck builder](#).

Using Intersphinx

To use Intersphinx you need to add it to the list of extensions in your `conf.py` file.

```
# conf.py file

extensions = [
    'sphinx.ext.intersphinx',
]
```

And use the `intersphinx_mapping` configuration to indicate the name and link of the projects you want to use.

```
# conf.py file

intersphinx_mapping = {
    'sphinx': ('https://www.sphinx-doc.org/en/master/', None),
}
```

Now we can use the `sphinx` name with a cross-reference role:

```
- :ref:`sphinx:ref-role`
- :ref:`:ref: role <sphinx:ref-role>`
- :doc:`sphinx:usage/extensions/intersphinx`
- :doc:`Intersphinx <sphinx:usage/extensions/intersphinx>`
```

Result:

- [Cross-referencing arbitrary locations](#)
- `:ref: role`
- `sphinx.ext.intersphinx` – Link to other projects' documentation
- **Intersphinx**

Note: You can get the targets used in Intersphinx by inspecting the source file of the project or using [this utility](#) provided by Intersphinx:


```
$ python -msphinx.ext.intersphinx https://www.sphinx-doc.org/en/master/objects.inv
```

Intersphinx in Read the Docs

You can use Intersphinx to link to subprojects, translations, another version or any other project hosted in Read the Docs. For example:

```
# conf.py file

intersphinx_mapping = {
    # Links to "v2" version of the "docs" project.
    'docs-v2': ('https://docs.readthedocs.io/en/v2', None),
    # Links to the French translation of the "docs" project.
    'docs-fr': ('https://docs.readthedocs.io/fr/latest', None),
    # Links to the "apis" subproject of the "docs" project.
    'sub-apis': ('https://docs.readthedocs.io/projects/apis/en/latest', None),
}
```

Intersphinx with private projects

If you are using *Read the Docs for Business*, Intersphinx will not be able to fetch the inventory file from private docs.

Intersphinx supports [URLs with Basic Authorization](#), which Read the Docs supports *using a token*. You need to generate a token for each project you want to use with Intersphinx.

1. Go the project you want to use with Intersphinx
2. Click *Admin > Sharing*
3. Select HTTP Header Token
4. Set an expiration date long enough to use the token when building your project
5. Click on Share!.

Now we can add the link to the private project with the token like:

```
# conf.py file

intersphinx_mapping = {
    # Links to a private project named "docs"
    'docs': ('https://<token-for-docs>:@readthedocs-docs.readthedocs-hosted.com/en/
↪latest', None),
    # Links to the private French translation of the "docs" project
    'docs': ('https://<token-for-fr-translation>:@readthedocs-docs.readthedocs-hosted.
↪com/fr/latest', None),
    # Links to the private "apis" subproject of the "docs" project
    'docs': ('https://<token-for-apis>:@readthedocs-docs.readthedocs-hosted.com/
↪projects/apis/en/latest', None),
}
```

Note: Sphinx will strip the token from the URLs when generating the links.

You can use your tokens with environment variables, so you don't have to hard code them in your `conf.py` file. See *I Need Secrets (or Environment Variables) in my Build* to use environment variables inside Read the Docs.

For example, if you create an environment variable named `RTD_TOKEN_DOCS` with the token from the “docs” project. You can use it like this:

```
# conf.py file

import os
RTD_TOKEN_DOCS = os.environ.get('RTD_TOKEN_DOCS')

intersphinx_mapping = {
    # Links to a private project named "docs"
    'docs': (f'https://{RTD_TOKEN_DOCS}@readthedocs-docs.readthedocs-hosted.com/en/
↪latest', None),
}
```

Note: Another way of using Intersphinx with private projects is to download the inventory file and keep it in sync when the project changes. The inventory file is by default located at `objects.inv`, for example `https://readthedocs-docs.readthedocs-hosted.com/en/latest/objects.inv`.

```
# conf.py file

intersphinx_mapping = {
    # Links to a private project named "docs" using a local inventory file.
    'docs': ('https://readthedocs-docs.readthedocs-hosted.com/en/latest', 'path/to/
↪local/objects.inv'),
}
```

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."
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
```

This command will generate a file at `.tx/config` with all the information needed by the `transifex-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
```

2. 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.

Building With Pre-1.0 Versions 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 an older 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
```

More detailed information about how to specify dependencies can be found [here](#).

Upgrade your custom theme to be compatible with later MkDocs versions

It is possible that your build passes but your documentation doesn't look correct. This may be because newer MkDocs versions 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 were passing arguments to the extension by defining them between brackets (`toc(permalink=true)`) in your `mkdocs.yml` 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
```

Sphinx PDFs with Unicode

Sphinx offers different [LaTeX engines](#) that have better support for Unicode characters and non-European languages like Japanese or Chinese. By default Sphinx uses `pdflatex`, which does not have good support for Unicode characters and may make the PDF builder fail.

To build your documentation in PDF format, you need to configure Sphinx properly in your project's `conf.py`. Read the Docs will execute the proper commands depending on these settings. There are [several settings that can be defined](#) (all the ones starting with `latex_`), to modify Sphinx and Read the Docs behavior to make your documentation to build properly.

For docs that are not written in Chinese or Japanese, and if your build fails from a Unicode error, then try `xelatex` as the `latex_engine` instead of the default `pdflatex` in your `conf.py`:

```
latex_engine = 'xelatex'
```

When Read the Docs detects that your documentation is in Chinese or Japanese, it automatically adds some defaults for you.

For *Chinese* projects, it appends to your `conf.py` these settings:

```
latex_engine = 'xelatex'
latex_use_xindy = False
latex_elements = {
    'preamble': '\\usepackage[UTF8]{ctex}\n',
}
```

And for *Japanese* projects:

```
latex_engine = 'platex'
latex_use_xindy = False
```

Tip: You can always override these settings if you define them by yourself in your `conf.py` file.

Note: `xindy` is currently not supported by Read the Docs, but we plan to support it in the near future.

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 Sphinx Documentation](#) 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.

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](#).

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 %}
```

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 %}
```

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 %}
```

Additional variables

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

3.8.2 Read the Docs how-to guides

These guides will help you customize or tune aspects of Read the Docs.

Autobuild Documentation for Pull Requests

Read the Docs allows autobuilding documentation for pull/merge requests for GitHub or GitLab projects. This feature is currently available under a *Feature Flag*. So, you can enable this feature by sending us an [email](#) including your project URL.

Features

- **Build on Pull/Merge Request Event:** We create an external version and trigger a build for that version when we receive pull/merge request open event from the webhook. We also trigger a new build when a new commit has been pushed to the Pull/Merge Request.
- **Warning Banner for Pull/Merge Request Documentation:** While building documentation for pull/merge requests we add a warning banner at the top of those documentations to let the users know that this documentation was generated from pull/merge requests and is not the main documentation for the project.

- **Send Build Status Notification:** We send build status reports to the status API of the provider (e.g. GitHub, GitLab). When a build is triggered for a pull/merge request we send build pending notification with the build URL and after the build has finished we send success notification if the build succeeded without any error or failure notification if the build failed.

Fig. 1: GitHub Build Status Reporting for Pull Requests

Troubleshooting

After the feature is enabled on your project, you might hit one of these issues:

1. **Pull Requests builds are not triggering.** We only support GitHub and GitLab currently. You need to make sure that you Read the Docs account is connected with that providers social account. You can check this by going to your [profile settings](#).
2. **Build status is not being reported on your Pull/Merge Request.** You need to make sure that you have granted access to the Read the Docs [OAuth App](#) to your/organizations GitHub account. Also make sure your webhook is properly setup to handle events. You can setup or `re-sync` the webhook from your projects admin dashboard. Learn more about setting up webhooks from our [Webhook Documentation](#).

If you have tried all the above troubleshooting and still getting issues, please let us know by sending us an [email](#).

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 are triggered, successful or failed.

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 will be sent as POST request to your webhook URL:

```
{
  "name": "Read the Docs",
  "slug": "rtd",
  "build": {
    "id": 6321373,
    "commit": "e8dd17a3f1627dd206d721e4be08ae6766fda40",
    "state": "finished",
    "success": false,
    "date": "2017-02-15 20:35:54"
  }
}
```

You should now get notified on your webhook when your builds start and finish (failure/success)!

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 [Configuration File](#), see [formats](#).

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.system_packages` option to have access to them.

Requests more resources

If you still have problems building your documentation, we can increase build limits on a per-project basis, sending an email to support@readthedocs.org providing a good reason why your documentation needs more resources.

Technical Documentation Search Engine Optimization (SEO) Guide

This guide will help you optimize your documentation for search engines with the goal of increasing traffic to your docs. While you optimize your docs to make them more crawler friendly for search engine spiders, it's important to keep in mind that your ultimate goal is to make your docs more discoverable for your users. **You're trying to make sure that when a user types a question into a search engine that is answerable by your documentation, that your docs appear in the results.**

This guide isn't meant to be your only resource on SEO, and there's a lot of SEO topics not covered here. For additional reading, please see the [external resources](#) section.

While many of the topics here apply to all forms of technical documentation, this guide will focus on Sphinx, which is the most common documentation authoring tool on Read the Docs, as well as improvements provided by Read the Docs itself.

Table of contents

- [SEO Basics](#)
- [Optimizing your docs for search engine spiders](#)
 - [Avoid orphan pages](#)
 - [Avoid uncrawlable content](#)
 - [Redirects](#)
 - [Canonical URLs](#)
 - [Use a robots.txt file](#)
 - [Use a sitemap.xml file](#)
 - [Use meta tags](#)
- [Measure, iterate, & improve](#)
 - [Search engine feedback](#)
 - [Analytics tools](#)
- [External resources](#)

SEO Basics

Search engines like Google and Bing crawl through the internet following links in an attempt to understand and build an index of what various pages and sites are about. This is called “crawling” or “indexing”. When a person sends a query to a search engine, the search engine evaluates this index using a number of factors and attempts to return the results most likely to answer that person’s question.

How search engines “rank” sites based on a person’s query is part of their secret sauce. While some search engines publish the basics of their algorithms (see Google’s published details on PageRank), few search engines give all of the details in an attempt to prevent users from gaming the rankings with low value content which happens to rank well.

Both [Google](#) and [Bing](#) publish a set of guidelines to help make sites easier to understand for search engines and rank better. To summarize some of the most important aspects as they apply to technical documentation, your site should:

- Use descriptive and accurate titles in the HTML `<title>` tag. For Sphinx, the `<title>` comes from the first heading on the page.
- Ensure your URLs are descriptive. They are displayed in search results. Sphinx uses the source filename without the file extension as the URL.
- Make sure the words your readers would search for to find your site are actually included on your pages.
- Avoid low content pages or pages with very little original content.
- Avoid tactics that attempt to increase your search engine ranking without actually improving content.
- Google specifically [warns about automatically generated content](#) although this applies primarily to keyword stuffing and low value content. High quality documentation generated from source code (eg. auto generated API documentation) seems OK.

While both Google and Bing discuss site performance as an important factor in search result ranking, this guide is not going to discuss it in detail. Most technical documentation that uses Sphinx or Read the Docs generates static HTML and the performance is typically decent relative to most of the internet.

Optimizing your docs for search engine spiders

Once a crawler or spider finds your site, it will follow links and redirects in an attempt to find any and all pages on your site. While there are a few ways to guide the search engine in its crawl for example by using a [sitemap](#) or a [robots.txt file](#) which we’ll discuss shortly, the most important thing is making sure the spider can follow links on your site and get to all your pages.

Avoid orphan pages

Sphinx calls pages that don’t have links to them “orphans” and will throw a warning while building documentation that contains an orphan unless the warning is silenced with the [orphan directive](#):

```
$ make html
sphinx-build -b html -d _build/doctrees . _build/html
Running Sphinx v1.8.5
...
checking consistency... /path/to/file.rst: WARNING: document isn't included in any
↳ toctree
done
...
build finished with problems, 1 warning.
```

You can make all Sphinx warnings into errors during your build process by setting `SPHINXOPTS = -W --keep-going` in your Sphinx Makefile.

Avoid uncrawable content

While typically this isn't a problem with technical documentation, try to avoid content that is "hidden" from search engines. This includes content hidden in images or videos which the crawler may not understand. For example, if you do have a video in your docs, make sure the rest of that page describes the content of the video.

When using images, make sure to set the image alt text or set a caption on figures. For Sphinx, the image and figure directives support this:

```
.. image:: your-image.png
   :alt: A description of this image

.. figure:: your-image.png

   A caption for this figure
```

Redirects

Redirects tell search engines when content has moved. For example, if this guide moved from `guides/technical-docs-seo-guide.html` to `guides/sphinx-seo-guide.html`, there will be a time period where search engines will still have the old URL in their index and will still be showing it to users. This is why it is important to update your own links within your docs as well as redirecting. If the hostname moved from `docs.readthedocs.io` to `docs.readthedocs.org`, this would be even more important!

Read the Docs supports a few different kinds of *user defined redirects* that should cover all the different cases such as redirecting a certain page for all project versions, or redirecting one version to another.

Canonical URLs

Anytime very similar content is hosted at multiple URLs, it is pretty important to set a canonical URL. The canonical URL tells search engines where the original version your documentation is even if you have multiple versions on the internet (for example, incomplete translations or deprecated versions).

Read the Docs supports *setting the canonical URL* if you are using a *custom domain* under *Admin > Domains* in the Read the Docs dashboard.

Use a robots.txt file

A `robots.txt` file is readable by crawlers and lives at the root of your site (eg. <https://docs.readthedocs.io/robots.txt>). It tells search engines which pages to crawl or not to crawl and can allow you to control how a search engine crawls your site. For example, you may want to request that search engines *ignore unsupported versions of your documentation* while keeping those docs online in case people need them.

By default, Read the Docs serves a `robots.txt` for you. To customize this file, you can create a `robots.txt` file that is written to your documentation root on your default branch/version.

See the [Google's documentation on robots.txt](#) for additional details.

Use a sitemap.xml file

A sitemap is a file readable by crawlers that contains a list of pages and other files on your site and some metadata or relationships about them (eg. <https://docs.readthedocs.io/sitemap.xml>). A good sitemaps provides information like how frequently a page or file is updated or any alternate language versions of a page.

Read the Docs generates a sitemap for you that contains the last time your documentation was updated as well as links to active versions, subprojects, and translations your project has. We have a small separate guide on [sitemaps](#).

See the [Google docs on building a sitemap](#).

Use meta tags

Using a meta description allows you to customize how your pages look in search engine result pages.

Typically search engines will use the first few sentences of a page if no meta description is provided. In Sphinx, you can customize your meta description using the following RestructuredText:

```
.. meta::
    :description lang=en:
        Adding additional CSS or JavaScript files to your Sphinx documentation
        can let you customize the look and feel of your docs or add additional_
        ↪ functionality.
```

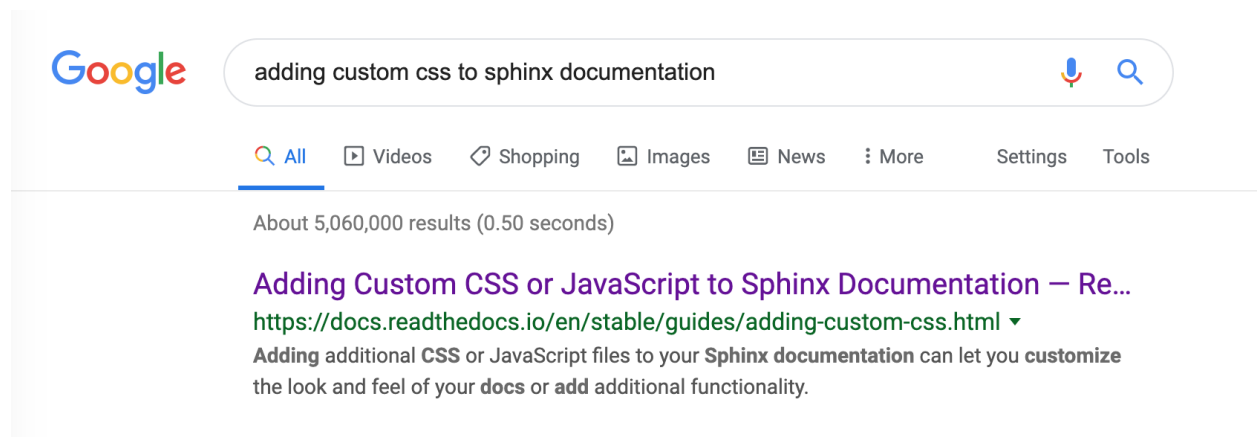


Fig. 2: Google search engine results showing a customized meta description

Moz.com, an authority on search engine optimization, makes the following suggestions for meta descriptions:

- Your meta description should have the most relevant content of the page. A searcher should know whether they've found the right page from the description.
- The meta description should be between 150-300 characters and it may be truncated down to around 150 characters in some situations.
- Meta descriptions are used for display but not for ranking.

Search engines don't always use your customized meta description if they think a snippet from the page is a better description.

Measure, iterate, & improve

Search engines (and soon, Read the Docs itself) can provide useful data that you can use to improve your docs' ranking on search engines.

Search engine feedback

[Google Search Console](#) and [Bing Webmaster Tools](#) are tools for webmasters to get feedback about the crawling of their sites (or docs in our case). Some of the most valuable feedback these provide include:

- Google and Bing will show pages that were previously indexed that now give a 404 (or more rarely a 500 or other status code). These will remain in the index for some time but will eventually be removed. This is a good opportunity to create a [redirect](#).
- These tools will show any crawl issues with your documentation.
- Search Console and Webmaster Tools will highlight security issues found or if Google or Bing took action against your site because they believe it is spammy.

Analytics tools

A tool like [Google Analytics](#) can give you feedback about the search terms people use to find your docs, your most popular pages, and lots of other useful data.

Search term feedback can be used to help you optimize content for certain keywords or for related keywords. For Sphinx documentation, or other technical documentation that has its own search features, analytics tools can also tell you the terms people search for within your site.

Knowing your popular pages can help you prioritize where to spend your SEO efforts. Optimizing your already popular pages can have a significant impact.

External resources

Here are a few additional resources to help you learn more about SEO and rank better with search engines.

- [Moz's beginners guide to SEO](#)
- [Google's Webmaster Guidelines](#)
- [Bing's Webmaster Guidelines](#)
- [Google's SEO Starter Guide](#)

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.

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`.

Enabling

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

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://docs.fabfile.org/en/2.4/" />
```

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:

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

Conda Support

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.

Activating Conda

Conda support is available using a *Configuration File*, see *conda*.

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)

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.

I Need Secrets (or Environment Variables) in my Build

It may happen that your documentation depends on an authenticated service to be built properly. In this case, you will require some secrets to access these services.

Read the Docs provides a way to define environment variables for your project to be used in the build process. All these variables will be exposed to all the commands executed when building your documentation.

To define an environment variable, you need to

1. Go to your project *Admin > Environment Variables*
2. Click on “Add Environment Variable” button
3. Input a Name and Value (your secret needed here)
4. Click “Save” button

Note: Values will never be exposed to users, even to owners of the project. Once you create an environment variable, you won't be able to see its value anymore for security purposes.

After adding an environment variable from your project's admin, you can access it from your build process using Python, for example:

```
# conf.py
import os
import requests

# Access to our custom environment variables
username = os.environ.get('USERNAME')
password = os.environ.get('PASSWORD')

# Request a username/password protected URL
response = requests.get(
    'https://httpbin.org/basic-auth/username/password',
    auth=(username, password),
)
```

Feature Flags

Read the Docs offers some additional flag settings which can only be configured by the site admin. These are optional settings and you might not need it for every project. By default, these flags are disabled for every project. A separate request can be made by opening an issue on our [github](#) to enable or disable one or more of these featured flags for a particular project.

Available Flags

`PIP_ALWAYS_UPGRADE`: Always run `pip install --upgrade`

`UPDATE_CONDA_STARTUP`: Upgrade conda before creating the environment

The version of `conda` used in the build process could not be the latest one. This is because we use Miniconda, which its release process is a little more slow than `conda` itself. In case you prefer to use the latest `conda` version available, this is the flag you need.

`CONDA_APPEND_CORE_REQUIREMENTS`: Append Read the Docs core requirements to `environment.yml` file

Makes Read the Docs to install all the requirements at once on `conda create` step. This helps users to pin dependencies on conda and to improve build time.

`DONT_OVERWRITE_SPHINX_CONTEXT`: Do not overwrite context vars in `conf.py` with Read the Docs context

`DONT_SHALLOW_CLONE`: Do not shallow clone when cloning git repos

The `DONT_SHALLOW_CLONE` flag is useful if your code accesses old commits during docs build, e.g. `python-reno` release notes manager is known to do that (error message line would probably include one of old Git commit id's).

`USE_TESTING_BUILD_IMAGE`: Use Docker image labelled as 'testing' to build the docs

`EXTERNAL_VERSION_BUILD`: Enable project to build on pull/merge requests

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.

Searching with Read the Docs

Read the Docs uses Elasticsearch to provide a better search experience. This guide is intended to show that how to add “search as you type” feature to your documentation, how to use advanced query syntax to get more accurate results and many other search features that Read the Docs supports with example searches.

You can find information on the search architecture and how we index document on our *Search* docs.

Table of contents

- *Improvements over Sphinx search*
- *Search features for project admins*
 - *Enable “search as you type” in your documentation*

- *Search analytics*
- *Search features for readers*
 - *Search across all projects*
 - *Search inside subprojects*
 - *Search query syntax*
 - * *Exact phrase search*
 - * *Exact phrase search with slop value*
 - * *Prefix query*
 - * *Fuzzy query*
 - *Using the search query syntax to build complex queries*

Improvements over Sphinx search

Sphinx is designed to create a self-contained webpage and all search indexing is done when the documentation is built. As a result, it would be impossible for Sphinx to add features like searching translations or subprojects or having analytics on common searches. Read the Docs supports a powerful documentation search unlike Sphinx which only have a basic search support.

Features of Read the Docs documentation search are:

- Search as you type feature supported.
- Search analytics.
- Search across multiple projects.
- Search inside subprojects.
- Advanced query syntax.
- Improved search result order.
- Public search API (documentation pending).
- Case insensitive search.
- Results from sections of the documentation.
- Code search.

Search features for project admins

Enable “search as you type” in your documentation

`readthedocs-sphinx-search` is a Sphinx extension which integrates your documentation more closely with Read the Docs’ search implementation. It adds a clean and minimal full page search UI which supports **search as you type** feature.

To get a glimpse of it, you can press / (forward slash) and start typing or just visit these URLs:

- https://docs.readthedocs.io/?rtd_search=contributing
- https://docs.readthedocs.io/?rtd_search=api/v3/projects/

Search analytics

Search queries are recorded and are stored in database to provide valuable analytics to the project admins. These analytics makes it easy to know what your users are looking for in your documentation. You can see these analytics in your project admin dashboard.

Note: Currently, this feature is in beta state and is available under a *feature flag*. We plan to make this available for everyone soon. If you want to test this feature out and help giving us feedback, please contact us via [GitHub issues](#).

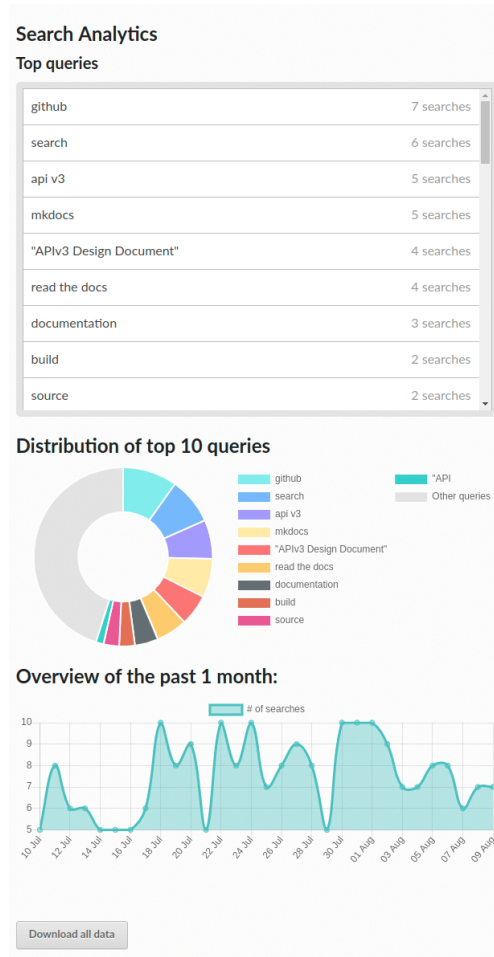


Fig. 3: Search analytics demo

Search features for readers

Search across all projects

Our [main site search](#) supports searching for projects and searching across all projects. You can also use it to select the specific project and version to narrow down the search results.

Example queries:

- <https://readthedocs.org/search/?q=celery&type=project>
- https://readthedocs.org/search/?q=celery._state&type=file
- https://readthedocs.org/search/?q=celery._state&type=file&project=celery
- https://readthedocs.org/search/?q=celery._state&type=file&project=celery&version=master

Search inside subprojects

We allow projects to be configured as subprojects of another project. You can read more about this in our [Subprojects](#) documentation.

If a search is made in a project which has one or more subprojects under it, the search results then also include the results from subprojects because they share a search index with their parent and sibling projects. For example: [Kombu](#) is one of the subprojects of [Celery](#), so if you search in Celery docs, then the results from Kombu will also be there. Example: https://docs.celeryproject.org/en/master/search.html?q=utilities&check_keywords=yes&area=default

Search query syntax

Read the Docs uses [Simple Query String](#) feature of [Elasticsearch](#), hence the search query can be made complex to get more accurate results.

Exact phrase search

If a query is wrapped in `"`, then only those results where the phrase is exactly matched will be returned.

Example queries:

- https://docs.readthedocs.io/?rtd_search=%22custom%20css%22
- https://docs.readthedocs.io/?rtd_search=%22adding%20a%20subproject%22
- https://docs.readthedocs.io/?rtd_search=%22when%20a%20404%20is%20returned%22

Exact phrase search with slop value

`~N` after a phrase signifies slop amount. It can be used to match words which are near one another.

Example queries:

- https://docs.readthedocs.io/?rtd_search=%22dashboard%20admin%22~2
- https://docs.readthedocs.io/?rtd_search=%22single%20documentation%22~1
- https://docs.readthedocs.io/?rtd_search=%22read%20the%20docs%20story%22~5

Prefix query

`*` at the end of any term signifies a prefix query. It returns the results containing the words with specific prefix.

Example queries:

- https://docs.readthedocs.io/?rtd_search=API%20v*
- https://docs.readthedocs.io/?rtd_search=single%20v*%20doc*

- https://docs.readthedocs.io/?rtd_search=build*%20and%20c*%20to%20doc*

Fuzzy query

~N after a word signifies edit distance (fuzziness). This type of query is helpful when spelling of the actual keyword is unsure. It returns results that contain terms similar to the search term, as measured by a [Levenshtein edit distance](#).

Example queries:

- https://docs.readthedocs.io/?rtd_search=reedthedcs~2
- https://docs.readthedocs.io/?rtd_search=authentication~3
- https://docs.readthedocs.io/?rtd_search=configurtion~1

Using the search query syntax to build complex queries

The search query syntaxes described in the previous section can be used with one another to build complex queries.

Example queries:

- https://docs.readthedocs.io/?rtd_search=auto*%20redirect*
- https://docs.readthedocs.io/?rtd_search=abandon*%20proj*
- https://docs.readthedocs.io/?rtd_search=localisation~3%20of%20doc*

Sitemaps: An SEO tool for documentation

Sitemaps allows us to inform search engines about URLs that are available for crawling and communicate them additional information about each URL of the project:

- when it was last updated,
- how often it changes,
- how important it is in relation to other URLs in the site, and
- what translations are available for a page.

Read the Docs automatically generates a sitemap for each project that hosts to improve results when performing a search on these search engines. This allow us to prioritize results based on the version number, for example to show `stable` as the top result followed by `latest` and then all the project's versions sorted following [semver](#).

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 a configuration file

The recommended approach for specifying a pip requirements file is to use a *Configuration File* file, see *Requirements 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 Docs' conda support: *Conda Support*.

Wiping a Build Environment

Sometimes it happens 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 be online after wiping the build environment.

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

3.8.3 Read the Docs for Business how-to guides

These guides are specific to *Read the Docs for Business*.

Using Private Git Submodules

Warning: This guide is for *Read the Docs for Business*.

Read the Docs uses SSH keys (with read only permissions) in order to clone private repositories. A SSH key is automatically generated and added to your main repository, but not to your submodules. In order to give Read the Docs access to clone your submodules you'll need to add the public SSH key to each repository of your submodules.

Note: You can manage which submodules Read the Docs should clone using a configuration file. See [submodules](#).

Table of contents

- [Copy your project's SSH Key](#)
- [Add the SSH key to your submodules](#)
 - [GitHub](#)
 - [GitLab](#)
 - [Bitbucket](#)
 - [Others](#)

Copy your project's SSH Key

You can find the public SSH key of your Read the Docs project by

1. Going to the *Admin* tab of your project
2. Click on *SSH Keys*
3. Click on the fingerprint of the SSH key (it looks like `6d:ca:6d:ca:6d:ca:6d:ca`)
4. Copy the text from the `Public` key section

Note: The private part of the SSH key is kept secret.

Add the SSH key to your submodules

GitHub

For GitHub, Read the Docs uses [deploy keys with read only access](#). Since GitHub doesn't allow you to reuse a deploy key across different repositories, you'll need to use [machine users](#) to give read access to several repositories using only one SSH key.

1. Remove the SSH deploy key that was added to the main repository on GitHub
 1. Go to your project on GitHub
 2. Click on *Settings*
 3. Click on *Deploy Keys*
 4. Delete the key added by Read the Docs Commercial (`readthedocs.com`)
2. Create a GitHub user and give it read only permissions to all the necessary repositories. You can do this by adding the account as:
 - A [collaborator](#)
 - An [outside collaborator](#)
 - A [team in an organization](#)
3. Attach the public SSH key from your project on Read the Docs to the GitHub user you just created
 1. Go to the user's settings
 2. Click on *SSH and GPG keys*
 3. Click on *New SSH key*
 4. Put a descriptive title and paste the
 5. *public SSH key from your Read the Docs project*
 6. Click on *Add SSH key*

GitLab

For GitLab, Read the Docs uses [deploy keys with read only access](#), which allows you to reuse a SSH key across different repositories. Since Read the Docs already added the public SSH key on your main repository, you only need to add it to each repository of your submodules.

1. Go to the project of your submodule on GitLab
2. Click on *Settings*
3. Click on *Repository*
4. Expand the *Deploy Keys* section
5. Put a descriptive title and paste the
6. *public SSH key from your Read the Docs project*
7. Click on *Add key*
8. Repeat the previous steps for each submodule

Bitbucket

For Bitbucket, Read the Docs uses [access keys with read only access](#), which allows you to reuse a SSH key across different repositories. Since Read the Docs already set the public SSH key on your main repository, you only need to add it to each repository of your submodules.

1. Go to the project of your submodule on Bitbucket
2. Click on *Settings*
3. Click on *Access keys*
4. Click on *Add key*
5. Put a descriptive label and paste the
6. [public SSH key from your Read the Docs project](#)
7. Click on *Add key*
8. Repeat the previous steps for each submodule

Others

If you are not using any of the above providers. Read the Docs will still generate a pair of SSH keys. You'll need to add the [public SSH key from your Read the Docs project](#) to the main repository and each of its submodules. Refer to your provider's documentation for the steps required to do this.

3.9 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.

3.9.1 API v2 (deprecated)

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 and better API v3 is ready to use. Some improvements coming in v3 are:

- Simpler URLs which use slugs
- Token based authentication
- Import a new project
- Activate a version
- Improved error reporting

See its full documentation at [API v3](#).

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](https://docs.readthedocs.io) 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",

```

(continues on next page)

(continued from previous page)

```

"repo": "https://github.com/pypa/pip",
"description": "Pip Installs Packages.",
"language": "en",
"documentation_type": "sphinx_htmldir",
"canonical_url": "http://pip.pypa.io/en/stable/",
"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,
  "type": "tag",
  "identifier": "3a6b3995c141c0888af6591a59240ba5db7d9914",
  "privacy_level": "public",
  "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",
      "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.

3.9.2 API v3

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

Table of contents

- *Authentication and authorization*
 - *Token*
 - *Session*
- *Resources*
 - *Projects*
 - * *Projects list*
 - * *Project details*
 - * *Project create*
 - * *Project update*
 - *Versions*
 - * *Versions listing*

- * *Version detail*
- * *Version update*
- *Builds*
 - * *Build details*
 - * *Builds listing*
 - * *Build triggering*
- *Subprojects*
 - * *Subproject details*
 - * *Subprojects listing*
 - * *Subproject create*
 - * *Subproject delete*
- *Translations*
 - * *Translations listing*
- *Redirects*
 - * *Redirect details*
 - * *Redirects listing*
 - * *Redirect create*
 - * *Redirect update*
 - * *Redirect delete*
- *Environment Variables*
 - * *Environment Variable details*
 - * *Environment Variables listing*
 - * *Environment Variable create*
 - * *Environment Variable delete*

Authentication and authorization

Requests to the Read the Docs public API are for public and private information. All endpoints require authentication.

Token

The `Authorization` HTTP header can be specified with `Token <your-access-token>` to authenticate as a user and have the same permissions that the user itself.

Note: You will find your access Token under [your profile settings](#).

Session

Warning: Authentication via session is not enabled yet.

Session authentication is allowed on very specific endpoints, to allow hitting the API when reading documentation. When a user is trying to authenticate via session, CSRF (Cross-site request forgery) check is performed.

Resources

This section shows all the resources that are currently available in APIv3. There are some URL attributes that applies to all of these resources:

?fields= Specify which fields are going to be returned in the response.

?omit= Specify which fields are going to be omitted from the response.

?expand= Some resources allow to expand/add extra fields on their responses (see *Project details* for example).

Tip: You can browse the full API by accessing its root URL: <https://readthedocs.org/api/v3/>

Projects

Projects list

GET `/api/v3/projects/`

Retrieve a list of all the projects for the current logged in user.

Example request:

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response:

```
{
  "count": 25,
  "next": "/api/v3/projects/?limit=10&offset=10",
  "previous": null,
  "results": ["PROJECT"]
}
```

Query Parameters

- **language** (*string*) – language code as en, es, ru, etc.
- **programming_language** (*string*) – programming language code as py, js, etc.

Project details

GET /api/v3/projects/ (**string:** *project_slug*) /
Retrieve details of a single project.

Example request:

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/  
↪pip/
```

Python

```
import requests  
URL = 'https://readthedocs.org/api/v3/projects/pip/'  
TOKEN = '<token>'  
HEADERS = {'Authorization': f'token {TOKEN}'}  
response = requests.get(URL, headers=HEADERS)  
print(response.json())
```

Example response:

```
{  
  "id": 12345,  
  "name": "Pip",  
  "slug": "pip",  
  "created": "2010-10-23T18:12:31+00:00",  
  "modified": "2018-12-11T07:21:11+00:00",  
  "language": {  
    "code": "en",  
    "name": "English"  
  },  
  "programming_language": {  
    "code": "py",  
    "name": "Python"  
  },  
  "repository": {  
    "url": "https://github.com/pypa/pip",  
    "type": "git"  
  },  
  "default_version": "stable",  
  "default_branch": "master",  
  "subproject_of": null,  
  "translation_of": null,  
  "urls": {  
    "documentation": "http://pip.pypa.io/en/stable/",  
    "home": "https://pip.pypa.io/"  
  },  
  "tags": [  
    "disutils",  
    "easy_install",  
  ]  
}
```

(continues on next page)

(continued from previous page)

```

    "egg",
    "setuptools",
    "virtualenv"
  ],
  "users": [
    {
      "username": "dstufft"
    }
  ],
  "active_versions": {
    "stable": "{VERSION}",
    "latest": "{VERSION}",
    "19.0.2": "{VERSION}"
  },
  "_links": {
    "_self": "/api/v3/projects/pip/",
    "versions": "/api/v3/projects/pip/versions/",
    "builds": "/api/v3/projects/pip/builds/",
    "subprojects": "/api/v3/projects/pip/subprojects/",
    "superproject": "/api/v3/projects/pip/superproject/",
    "redirects": "/api/v3/projects/pip/redirects/",
    "translations": "/api/v3/projects/pip/translations/"
  }
}

```

Query Parameters

- **expand** (*string*) – allows to add/expand some extra fields in the response. Allowed values are `active_versions`, `active_versions.last_build` and `active_versions.last_build.config`. Multiple fields can be passed separated by commas.

Project create

POST /api/v3/projects/

Import a project under authenticated user.

Example request:

Bash

```

$ curl \
-X POST \
-H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/ \
-H "Content-Type: application/json" \
-d @body.json

```

Python

```

import requests
import json
URL = 'https://readthedocs.org/api/v3/projects/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
data = json.load(open('body.json', 'rb'))

```

(continues on next page)

(continued from previous page)

```
response = requests.post(  
    URL,  
    json=data,  
    headers=HEADERS,  
)  
print(response.json())
```

The content of `body.json` is like,

```
{  
    "name": "Test Project",  
    "repository": {  
        "url": "https://github.com/readthedocs/template",  
        "type": "git"  
    },  
    "homepage": "http://template.readthedocs.io/",  
    "programming_language": "py",  
    "language": "es"  
}
```

Example response:

See [Project details](#)

Project update

PATCH `/api/v3/projects/(string: project_slug)/`

Update an existing project.

Example request:

Bash

```
$ curl \  
-X PATCH \  
-H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/pip/ \  
-H "Content-Type: application/json" \  
-d @body.json
```

Python

```
import requests  
import json  
URL = 'https://readthedocs.org/api/v3/projects/pip/'  
TOKEN = '<token>'  
HEADERS = {'Authorization': f'token {TOKEN}'}  
data = json.load(open('body.json', 'rb'))  
response = requests.patch(  
    URL,  
    json=data,  
    headers=HEADERS,  
)  
print(response.json())
```

The content of `body.json` is like,

```
{
  "name": "New name for the project",
  "repository": {
    "url": "https://github.com/readthedocs/readthedocs.org",
    "type": "git"
  },
  "language": "ja",
  "programming_language": "py",
  "homepage": "https://readthedocs.org/",
  "default_version": "v0.27.0",
  "default_branch": "develop",
  "analytics_code": "UA000000",
  "single_version": false,
}
```

Status Codes

- 204 No Content – Updated successfully

Versions

Versions are different versions of the same project documentation.

The versions for a given project can be viewed in a project's version page. For example, here is the [Pip project's version page](#). See [Versions](#) for more information.

Versions listing

GET `/api/v3/projects/(string: project_slug)/versions/`

Retrieve a list of all versions for a project.

Example request:

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/
↳pip/versions/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/versions/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response:

```
{
  "count": 25,
  "next": "/api/v3/projects/pip/versions/?limit=10&offset=10",
  "previous": null,
  "results": ["VERSION"]
}
```

Query Parameters

- **active** (*boolean*) – return only active versions
- **built** (*boolean*) – return only built versions

Version detail

GET `/api/v3/projects/(string: project_slug)/versions/`
string: *version_slug* / Retrieve details of a single version.

Example request:

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/  
↳pip/versions/stable/
```

Python

```
import requests  
URL = 'https://readthedocs.org/api/v3/projects/pip/versions/stable/'  
TOKEN = '<token>'  
HEADERS = {'Authorization': f'token {TOKEN}'}  
response = requests.get(URL, headers=HEADERS)  
print(response.json())
```

Example response:

```
{  
  "id": 71652437,  
  "slug": "stable",  
  "verbose_name": "stable",  
  "identifier": "3a6b3995c141c0888af6591a59240ba5db7d9914",  
  "ref": "19.0.2",  
  "built": true,  
  "active": true,  
  "type": "tag",  
  "last_build": "{BUILD}",  
  "downloads": {  
    "pdf": "https://pip.readthedocs.io/_/downloads/pdf/pip/stable/",  
    "htmlzip": "https://pip.readthedocs.io/_/downloads/htmlzip/pip/stable/",  
    "epub": "https://pip.readthedocs.io/_/downloads/epub/pip/stable/"  
  },  
  "urls": {  
    "documentation": "https://pip.pypa.io/en/stable/",  
    "vcs": "https://github.com/pypa/pip/tree/19.0.2"  
  },  
  "_links": {  
    "_self": "/api/v3/projects/pip/versions/stable/",  
    "builds": "/api/v3/projects/pip/versions/stable/builds/",  
    "project": "/api/v3/projects/pip/"  
  }  
}
```

Response JSON Object

- **ref** (*string*) – the version slug where the `stable` version points to. `null` when it's not the stable version.
- **built** (*boolean*) – the version has at least one successful build.

Query Parameters

- **expand** (*string*) – allows to add/expand some extra fields in the response. Allowed values are `last_build` and `last_build.config`. Multiple fields can be passed separated by commas.

Version update

PATCH `/api/v3/projects/(string: project_slug)/versions/`
string: *version_slug*/ Update a version.

Example request:

Bash

```
$ curl \
  -X PATCH \
  -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/pip/
↪versions/0.23/ \
  -H "Content-Type: application/json" \
  -d @body.json
```

Python

```
import requests
import json
URL = 'https://readthedocs.org/api/v3/projects/pip/versions/0.23/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
data = json.load(open('body.json', 'rb'))
response = requests.patch(
    URL,
    json=data,
    headers=HEADERS,
)
print(response.json())
```

The content of `body.json` is like,

```
{
  "active": true
}
```

Status Codes

- **204 No Content** – Updated successfully

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 page for a project. For example, here is [Pip's build page](#). See *Build Process* for more information.

Build details

GET `/api/v3/projects/(str: project_slug)/builds/`
`int: build_id`/ Retrieve details of a single build for a project.

Example request:

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/
↳pip/builds/8592686/?expand=config
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/builds/8592686/?expand=config'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response:

```
{
  "id": 8592686,
  "version": "latest",
  "project": "pip",
  "created": "2018-06-19T15:15:59+00:00",
  "finished": "2018-06-19T15:16:58+00:00",
  "duration": 59,
  "state": {
    "code": "finished",
    "name": "Finished"
  },
  "success": true,
  "error": null,
  "commit": "6f808d743fd6f6907ad3e2e969c88a549e76db30",
  "config": {
    "version": "1",
    "formats": [
      "htmlzip",
      "epub",
      "pdf"
    ],
    "python": {
      "version": 3,
      "install": [
        {
          "requirements": ".../stable/tools/docs-requirements.txt"
        }
      ],
      "use_system_site_packages": false
    },
    "conda": null,
  },
}
```

(continues on next page)

(continued from previous page)

```

    "build": {
      "image": "readthedocs/build:latest"
    },
    "doctype": "sphinx_htmldir",
    "sphinx": {
      "builder": "sphinx_htmldir",
      "configuration": "../stable/docs/html/conf.py",
      "fail_on_warning": false
    },
    "mkdocs": {
      "configuration": null,
      "fail_on_warning": false
    },
    "submodules": {
      "include": "all",
      "exclude": [],
      "recursive": true
    }
  },
  "_links": {
    "_self": "/api/v3/projects/pip/builds/8592686/",
    "project": "/api/v3/projects/pip/",
    "version": "/api/v3/projects/pip/versions/latest/"
  }
}

```

Response JSON Object

- **created** (*string*) – The ISO-8601 datetime when the build was created.
- **finished** (*string*) – The ISO-8601 datetime when the build has finished.
- **duration** (*integer*) – The length of the build in seconds.
- **state** (*string*) – The state of the build (one of `triggered`, `building`, `installing`, `cloning`, or `finished`)
- **error** (*string*) – An error message if the build was unsuccessful

Query Parameters

- **expand** (*string*) – allows to add/expand some extra fields in the response. Allowed value is `config`.

Builds listing

GET `/api/v3/projects/(str: project_slug)/builds/`
 Retrieve list of all the builds on this project.

Example request:

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/
↳ pip/builds/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/builds/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response:

```
{
  "count": 15,
  "next": "/api/v3/projects/pip/builds?limit=10&offset=10",
  "previous": null,
  "results": ["BUILD"]
}
```

Query Parameters

- **commit** (*string*) – commit hash to filter the builds returned by commit
- **running** (*boolean*) – filter the builds that are currently building/running

Build triggering

POST `/api/v3/projects/{string: project_slug}/versions/{string: version_slug}/builds/` Trigger a new build for the *version_slug* version of this project.

Example request:

Bash

```
$ curl \
  -X POST \
  -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/pip/
↪versions/latest/builds/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/versions/latest/builds/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.post(URL, headers=HEADERS)
print(response.json())
```

Example response:

```
{
  "build": "{BUILD}",
  "project": "{PROJECT}",
  "version": "{VERSION}"
}
```

Status Codes

- **202 Accepted** – the build was triggered

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. See [Subprojects](#) for more information.

Subproject details

GET `/api/v3/projects/(str: project_slug)/subprojects/`
str: *alias_slug*/ Retrieve details of a subproject relationship.

Example request:

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/
↳pip/subprojects/subproject-alias/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/subprojects/subproject-alias/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response:

```
{
  "alias": "subproject-alias",
  "child": ["PROJECT"],
  "_links": {
    "parent": "/api/v3/projects/pip/"
  }
}
```

Subprojects listing

GET `/api/v3/projects/(str: project_slug)/subprojects/`
 Retrieve a list of all sub-projects for a project.

Example request:

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/
↳pip/subprojects/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/subprojects/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
```

(continues on next page)

(continued from previous page)

```
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response:

```
{
  "count": 25,
  "next": "/api/v3/projects/pip/subprojects/?limit=10&offset=10",
  "previous": null,
  "results": ["SUBPROJECT RELATIONSHIP"]
}
```

Subproject create

POST /api/v3/projects/(str: *project_slug*)/subprojects/

Create a subproject relationship between two projects.

Example request:

Bash

```
$ curl \
-X POST \
-H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/pip/
↪subprojects/ \
-H "Content-Type: application/json" \
-d @body.json
```

Python

```
import requests
import json
URL = 'https://readthedocs.org/api/v3/projects/pip/subprojects/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
data = json.load(open('body.json', 'rb'))
response = requests.post(
    URL,
    json=data,
    headers=HEADERS,
)
print(response.json())
```

The content of `body.json` is like,

```
{
  "child": "subproject-child-slug",
  "alias": "subproject-alias"
}
```

Example response:

See Subproject details

Response JSON Object

- **child** (*string*) – slug of the child project in the relationship.

- **alias** (*string*) – optional slug alias to be used in the URL (e.g. /projects/<alias>/en/latest/). If not provided, child project's slug is used as alias.

Status Codes

- **201 Created** – Subproject created successfully

Subproject delete

DELETE /api/v3/projects/ (**str:** *project_slug*) /subprojects/
str: *alias_slug*/ Delete a subproject relationship.

Example request:

Bash

```
$ curl \
-X DELETE \
-H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/pip/
↳subprojects/subproject-alias/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/subprojects/subproject-alias/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.delete(URL, headers=HEADERS)
print(response.json())
```

Status Codes

- **204 No Content** – Subproject deleted successfully

Translations

Translations are the same version of a Project in a different language. See *Localization of Documentation* for more information.

Translations listing

GET /api/v3/projects/ (**str:** *project_slug*) /translations/
 Retrieve a list of all translations for a project.

Example request:

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/
↳pip/translations/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/translations/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response:

```
{
  "count": 25,
  "next": "/api/v3/projects/pip/translations/?limit=10&offset=10",
  "previous": null,
  "results": ["PROJECT"]
}
```

Redirects

Redirects allow the author to redirect an old URL of the documentation to a new one. This is useful when pages are moved around in the structure of the documentation set. See [User-defined Redirects](#) for more information.

Redirect details

GET `/api/v3/projects/{str: project_slug}/redirects/{int: redirect_id}` Retrieve details of a single redirect for a project.

Example request

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/
↳pip/redirects/1/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/redirects/1/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response

```
{
  "pk": 1,
  "created": "2019-04-29T10:00:00Z",
  "modified": "2019-04-29T12:00:00Z",
  "project": "pip",
  "from_url": "/docs/",
  "to_url": "/documentation/",
  "type": "page",
  "_links": {
    "_self": "/api/v3/projects/pip/redirects/1/",

```

(continues on next page)

(continued from previous page)

```

    "project": "/api/v3/projects/pip/"
  }
}

```

Redirects listing

GET /api/v3/projects/(str: *project_slug*)/redirects/
 Retrieve list of all the redirects for this project.

Example request

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/
↳pip/redirects/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/redirects/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response

```
{
  "count": 25,
  "next": "/api/v3/projects/pip/redirects/?limit=10&offset=10",
  "previous": null,
  "results": ["REDIRECT"]
}
```

Redirect create

POST /api/v3/projects/pip/redirects/
 Create a redirect for this project.

Example request:

Bash

```
$ curl \
-X POST \
-H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/pip/
↳redirects/ \
-H "Content-Type: application/json" \
-d @body.json
```

Python

```
import requests
import json
URL = 'https://readthedocs.org/api/v3/projects/pip/redirects/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
data = json.load(open('body.json', 'rb'))
response = requests.post(
    URL,
    json=data,
    headers=HEADERS,
)
print(response.json())
```

The content of `body.json` is like,

```
{
  "from_url": "/docs/",
  "to_url": "/documentation/",
  "type": "page"
}
```

Note: type can be one of `prefix`, `page`, `exact`, `sphinx_html` and `sphinx_htmldir`.

Depending on the type of the redirect, some fields may not be needed:

- `prefix` type does not require `to_url`.
 - `page` and `exact` types require `from_url` and `to_url`.
 - `sphinx_html` and `sphinx_htmldir` types do not require `from_url` and `to_url`.
-

Example response:

See Redirect details

Status Codes

- **201 Created** – redirect created successfully

Redirect update

PUT `/api/v3/projects/{str: project_slug}/redirects/{int: redirect_id}` Update a redirect for this project.

Example request:

Bash

```
$ curl \
  -X PUT \
  -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/pip/
↪redirects/1/ \
  -H "Content-Type: application/json" \
  -d @body.json
```

Python

```
import requests
import json
URL = 'https://readthedocs.org/api/v3/projects/pip/redirects/1/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
data = json.load(open('body.json', 'rb'))
response = requests.put(
    URL,
    json=data,
    headers=HEADERS,
)
print(response.json())
```

The content of `body.json` is like,

```
{
  "from_url": "/docs/",
  "to_url": "/documentation.html",
  "type": "page"
}
```

Example response:

See [Redirect details](#)

Redirect delete

DELETE `/api/v3/projects/{str: project_slug}/redirects/{int: redirect_id/}` Delete a redirect for this project.

Example request:

Bash

```
$ curl \
  -X DELETE \
  -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/pip/
↪ redirects/1/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/redirects/1/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.delete(URL, headers=HEADERS)
print(response.json())
```

Status Codes

- **204 No Content** – Redirect deleted successfully

Environment Variables

Environment Variables are variables that you can define for your project. These variables are used in the build process when building your documentation. They are useful to define secrets in a safe way that can be used by your

documentation to build properly. See *I Need Secrets (or Environment Variables) in my Build*

Environment Variable details

GET `/api/v3/projects/(str: project_slug)/environmentvariables/int: environmentvariable_id/` Retrieve details of a single environment variable for a project.

Example request

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/
↳pip/environmentvariables/1/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/environmentvariables/1/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response

```
{
  "_links": {
    "_self": "https://readthedocs.org/api/v3/projects/project/
↳environmentvariables/1/",
    "project": "https://readthedocs.org/api/v3/projects/project/"
  },
  "created": "2019-04-29T10:00:00Z",
  "modified": "2019-04-29T12:00:00Z",
  "pk": 1,
  "project": "project",
  "name": "ENVVAR"
}
```

Environment Variables listing

GET `/api/v3/projects/(str: project_slug)/environmentvariables/`
Retrieve list of all the environment variables for this project.

Example request

Bash

```
$ curl -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/
↳pip/environmentvariables/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/environmentvariables/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
```

(continues on next page)

(continued from previous page)

```
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response

```
{
  "count": 15,
  "next": "/api/v3/projects/pip/environmentvariables/?limit=10&offset=10",
  "previous": null,
  "results": ["ENVIRONMENTVARIABLE"]
}
```

Environment Variable create**POST /api/v3/projects/pip/environmentvariables/**

Create an environment variable for this project.

Example request:**Bash**

```
$ curl \
  -X POST \
  -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/pip/
  ↪environmentvariables/ \
  -H "Content-Type: application/json" \
  -d @body.json
```

Python

```
import requests
import json
URL = 'https://readthedocs.org/api/v3/projects/pip/environmentvariables/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
data = json.load(open('body.json', 'rb'))
response = requests.post(
    URL,
    json=data,
    headers=HEADERS,
)
print(response.json())
```

The content of body.json is like,

```
{
  "name": "MYVAR",
  "value": "My secret value"
}
```

Example response:

See Environment Variable details

Status Codes

- **201 Created** – Environment variable created successfully

Environment Variable delete

DELETE `/api/v3/projects/` (`str: project_slug`) `/environmentvariables/`
`int: environmentvariable_id`/ Delete an environment variable for this project.

Example request:

Bash

```
$ curl \
  -X DELETE \
  -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/pip/
  ↪environmentvariables/1/
```

Python

```
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/environmentvariables/1/'
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.delete(URL, headers=HEADERS)
print(response.json())
```

Request Headers

- `Authorization` – token to authenticate.

Status Codes

- `204 No Content` – Environment variable deleted successfully

The Read the Docs project and organization

Learn about Read the Docs, the project and the company, and find out how you can get involved and contribute to the development and success of Read the Docs and the larger software documentation ecosystem.

- **Getting involved with Read the Docs:** [Contributing](#) | [Roadmap](#) | [Google Summer of Code](#) | [Code of conduct](#)
- **Policies & Process:** [Security](#) | [Privacy policy](#) | [Terms of service](#) | [DMCA takedown policy](#) | [Policy for abandoned projects](#) | [Release notes & changelog](#)
- **The people and philosophy behind Read the Docs:** [Team](#) | [Open source philosophy](#) | [Our story](#)
- **Financial and material support:** [Advertising](#) | [Sponsors](#)
- **Read the Docs for Business:** [Support and additional features](#)
- **Running your own version of Read the Docs:** [Private installations](#)

4.1 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.

4.1.1 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](#).

4.1.2 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 it 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.

4.1.3 Contributing to documentation

Documentation for Read the Docs itself is hosted by Read the Docs at <https://docs.readthedocs.io> (likely the website you are currently reading).

There are guidelines around writing and formatting documentation for the project. For full details, including how to build it, see *Building and Contributing to Documentation*.

4.1.4 Developer documentation

These are guides and helpful documentation to running your own local version of Read the Docs for development or taking the open source Read the Docs codebase for your own *custom installation*.

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.

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.

Warning: Currently Read the Docs is using Django 1.11.x and this version of Django has a [bug](#) which breaks database migrations if you are using `sqlite 3.26.0` or Newer. So, we recommend using `sqlite < 3.26.0` to run Read the Docs properly on your machine.

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)
 - Follow [Search](#) documentation for more instruction.

Note: You can import Python 2.x or 3.x projects in RTD, make sure you install the appropriate Python version (2.x or 3.x) with virtualenv.

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.

Get and run Read the Docs

Clone the repository somewhere on your disk and enter to the repository:

```
git clone --recurse-submodules https://github.com/readthedocs/readthedocs.org.git
cd readthedocs.org
```

Create a virtual environment and activate it:

```
virtualenv --python=python3 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 that the port you're serving on (i.e. `python manage.py runserver 0.0.0.0:8080`) matches 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 a project successfully, it is probably because of some missing libraries for pdf and epub generation. You can uncheck this on the advanced settings of your project.

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.

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](#).

Development Standards

These are build standards that are adhered to by the core development team while developing Read the Docs and related services. If you are a new contributor to Read the Docs, it might be a good idea to follow these guidelines as well. The [standard installation instructions](#) do cover what it takes to get started with a local installation of Read the Docs and can be used for local development by non-core team.

Warning: Take into account that Core team does not offer support on this setup currently.

Core team standards

Core team members expect to have a development environment that closely approximates our production environment, in order to spot bugs and logical inconsistencies before they make their way to production.

Currently, core team members are migrating to a Docker Compose based solution that is not yet recommended for contributing to development.

This solution gives us many features that allows us to have an environment closer to production:

Celery runs as a separate process Avoids masking bugs that could be introduced by Celery tasks in a race conditions.

Celery runs multiple processes We run celery with multiple worker processes to discover race conditions between tasks.

Docker for builds Docker is used for a build backend instead of the local host build backend. There are a number of differences between the two execution methods in how processes are executed, what is installed, and what can potentially leak through and mask bugs – for example, local SSH agent allowing code check not normally possible.

Serve documentation under a subdomain There are a number of resolution bugs and cross-domain behavior that can only be caught by using `USE_SUBDOMAIN` setting.

PostgreSQL as a database It is recommended that Postgres be used as the default database whenever possible, as SQLite has issues with our Django version and we use Postgres in production. Differences between Postgres and SQLite should be masked for the most part however, as Django does abstract database procedures, and we don't do any Postgres-specific operations yet.

Celery is isolated from database Celery workers on our build servers do not have database access and need to be written to use API access instead.

Use NGINX as web server All the site is served via NGINX with the ability to change some configuration locally.

Azurite as Django storage backend All static and media files are served using Azurite –an emulator of Azure Blob Storage, which is the one used in production.

Serve documentation via El Proxito Documentation is proxied by NGINX to El Proxito and proxied back to NGINX to be served finally. El Proxito is a small application put in front of the documentation to serve files from the Django Storage Backend.

Search enabled by default Elasticsearch is properly configured and enabled by default. Besides, all the documentation indexes are updated after a build is finished.

Set up your environment

After cloning `readthedocs.org` repository, you need to

1. install [Docker](#) following [their installation guide](#).

2. install the requirements from `common` submodule:

```
$ pip install -r common/dockerfiles/requirements.txt
```

3. build the Docker image for the servers:

Warning: This command could take a while to finish since it will download several Docker images.

```
$ inv docker.build
```

Tip: If you pass `GITHUB_TOKEN` environment variable to this command, it will add support for `readthedocs-ext`.

4. pull down Docker images for the builders:

```
$ inv docker.pull --only-latest
```

5. start all the containers:

```
$ inv docker.up --init # --init is only needed the first time
```

6. go to <http://community.dev.readthedocs.io> to access your local instance of Read the Docs.

Working with Docker Compose

We wrote a wrapper with `invoke` around `docker-compose` to have some shortcuts and save some work while typing docker compose commands. This section explains these `invoke` commands:

inv docker.build Builds the generic Docker image used by our servers (web, celery, build and proxito).

inv docker.up Starts all the containers needed to run Read the Docs completely.

- `--no-search` can be passed to disable search
- `--init` is used the first time this command is ran to run initial migrations, create an admin user, setup Azurite containers, etc
- `--no-reload` makes all celery processes and django runserver to use no reload and do not watch for files changes

inv docker.shell Opens a shell in a container (web by default).

- `--running` the shell is open in a container that it's already running
- `--container` specifies in which container the shell is open

inv docker.manage {command} Executes a Django management command in a container.

Tip: Useful when modifying models to run makemigrations.

inv docker.down Stops and removes all containers running.

- `--volumes` will remove the volumes as well (database data will be lost)

inv docker.restart {containers} Restarts the containers specified (automatically restarts NGINX when needed).

inv docker.attach {container} Grab STDIN/STDOUT control of a running container.

Tip: Useful to debug with `pdb`. Once the program has stopped in your `pdb` line, you can run `inv docker.attach web` and jump into a `pdb` session (it also works with `ipdb` and `pdb++`)

inv docker.test Runs all the test suites inside the container.

- `--arguments` will pass arguments to Tox command (e.g. `--arguments "-e py36 -- -k test_api"`)

inv docker.pull Downloads and tags all the Docker images required for builders.

- `--only-latest` does not pull stable and testing images.

Adding a new Python dependency

The Docker image for the servers is built with the requirements defined in the `master` branch. In case you need to add a new Python dependency while developing, you can use the `common/dockerfiles/entrypoints/common.sh` script as shortcut.

This script is run at startup on all the servers (web, celery, builder, proxito) which allows you to test your dependency without re-building the whole image. To do this, add the `pip` command required for your dependency in `common.sh` file:

```
# common.sh
pip install my-dependency==1.2.3
```

Once the PR that adds this dependency was merged into `master`, you can rebuild the image so the dependency is added to the Docker image itself and it's not needed to be installed each time the container spins up.

Debugging Celery

In order to step into the worker process, you can't use `pdb` or `ipdb`, but you can use `celery.contrib.rdb`:

```
from celery.contrib import rdb; rdb.set_trace()
```

When the breakpoint is hit, the Celery worker will pause on the breakpoint and will alert you on `STDOUT` of a port to connect to. You can open a shell into the container with `inv docker.shell celery` (or `build`) and then use `telnet` or `netcat` to connect to the debug process port:

```
$ nc 127.0.0.1 6900
```

The `rdb` debugger is similar to `pdb`, there is no `ipdb` for remote debugging currently.

Search

Read The Docs uses [Elasticsearch](#) instead of the built in Sphinx search for providing better search results. Documents are indexed in the Elasticsearch index and the search is made through the API. All the Search Code is open source and lives in the [GitHub Repository](#). Currently we are using the [Elasticsearch 6.3](#) version.

Local Development Configuration

Installing and running Elasticsearch

You need to install and run [Elasticsearch](#) version 6.3 on your local development machine. You can get the installation instructions [here](#). Otherwise, you can also start an Elasticsearch Docker container by running the following command:

```
docker run -p 9200:9200 -p 9300:9300 \
  -e "discovery.type=single-node" \
  docker.elastic.co/elasticsearch/elasticsearch:6.3.2
```

You need to override the `ES_HOSTS` and `ELASTICSEARCH_DSL` settings to point to `127.0.0.1:9200`.

Indexing into Elasticsearch

For using search, you need to index data to the Elasticsearch Index. Run `reindex_elasticsearch` management command:

```
./manage.py reindex_elasticsearch
```

For performance optimization, we implemented our own version of management command rather than the built in management command provided by the [django-elasticsearch-dsl](#) package.

Auto Indexing

By default, Auto Indexing is turned off in development mode. To turn it on, change the `ELASTICSEARCH_DSL_AUTOSYNC` settings to `True` in the `readthedocs/settings/dev.py` file. After that, whenever a documentation successfully builds, or project gets added, the search index will update automatically.

Architecture

The search architecture is divided into 2 parts.

- One part is responsible for **indexing** the documents and projects (`documents.py`)
- The other part is responsible for **querying** the Index to show the proper results to users (`faceted_search.py`)

We use the `django-elasticsearch-dsl` package for our Document abstraction. `django-elasticsearch-dsl` is a wrapper around `elasticsearch-dsl` for easy configuration with Django.

Indexing

All the Sphinx documents are indexed into Elasticsearch after the build is successful. Currently, we do not index MkDocs documents to elasticsearch, but [any kind of help is welcome](#).

How we index documentations

After any build is successfully finished, `HTMLFile` objects are created for each of the HTML files and the old version's `HTMLFile` object is deleted. By default, `django-elasticsearch-dsl` package listens to the `post_create/post_delete` signals to index/delete documents, but it has performance drawbacks as it send HTTP request whenever any `HTMLFile` objects is created or deleted. To optimize the performance, `bulk_post_create` and `bulk_post_delete` signals are dispatched with list of `HTMLFile` objects so its possible to bulk index documents in elasticsearch (`bulk_post_create` signal is dispatched for created and `bulk_post_delete` is dispatched for deleted objects). Both of the signals are dispatched with the list of the instances of `HTMLFile` in `instance_list` parameter.

We listen to the `bulk_post_create` and `bulk_post_delete` signals in our `Search` application and index/delete the documentation content from the `HTMLFile` instances.

How we index projects

We also index project information in our search index so that the user can search for projects from the main site. We listen to the `post_create` and `post_delete` signals of `Project` model and index/delete into Elasticsearch accordingly.

Elasticsearch Document

`elasticsearch-dsl` provides a model-like wrapper for the `Elasticsearch` document. As per requirements of `django-elasticsearch-dsl`, it is stored in the `readthedocs/search/documents.py` file.

ProjectDocument: It is used for indexing projects. Signal listener of `django-elasticsearch-dsl` listens to the `post_save` signal of `Project` model and then index/delete into Elasticsearch.

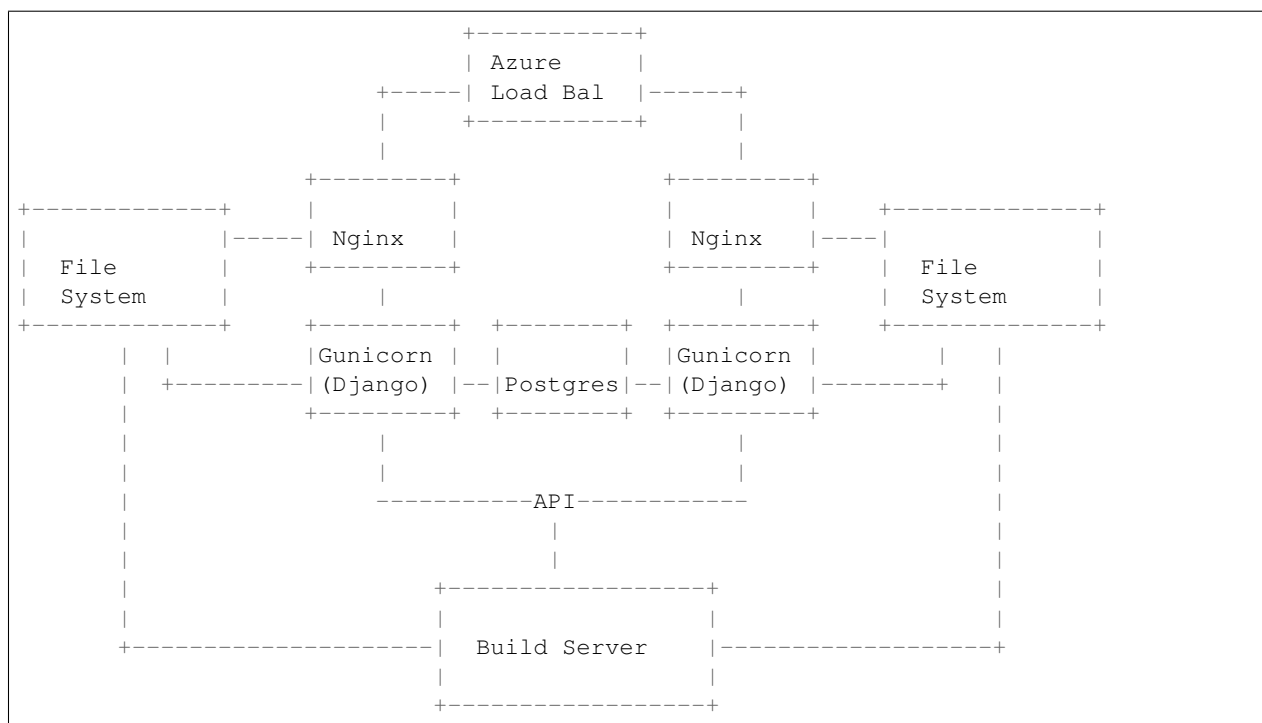
PageDocument: It is used for indexing documentation of projects. As mentioned above, our `Search` app listens to the `bulk_post_create` and `bulk_post_delete` signals and indexes/deleted documentation into Elasticsearch. The signal listeners are in the `readthedocs/search/signals.py` file. Both of the signals are dispatched after a successful documentation build.

The fields and ES Datatypes are specified in the `PageDocument`. The indexable data is taken from `processed_json` property of `HTMLFile`. This property provides python dictionary with document data like `title`, `sections`, `path` etc.

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.

Diagram



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
```

By default, tox won't run tests from search, in order to run all test including the search tests, you need to override tox's posargs. If you don't have any additional arguments to pass, you can also set the `TOX_POSARGS` environment variable to an empty string:

```
TOX_POSARGS='' tox
```

Note: If you need to override tox's posargs, but you still don't want to run the search tests, you need to include `-m 'not search'` to your command:

```
tox -- -m 'not search' -x
```

Warning: Running tests for search needs an Elasticsearch *instance running locally*.

To target a specific environment:

```
tox -e py36
```

The `tox` configuration has the following environments configured. You can target a single environment to limit the test suite:

py36 Run our test suite using Python 3.6

lint Run code linting using [Prospector](#). This currently runs `pylint`, `pyflakes`, `pep8` and other linting tools.

docs Test documentation compilation with Sphinx.

Pytest marks

The Read the Docs code base is deployed as three instances:

- Main: where you can see the dashboard.
- Build: where the builds happen.
- Serve/proxito: It is in charge of serving the documentation pages.

Each instance has its own settings. To make sure we test each part as close as possible to its real settings, we use `pytest marks`. This allow us to run each set of tests with different settings files, or skip some (like search tests):

```
DJANGO_SETTINGS_MODULE=custom.settings.file pytest -m mark
DJANGO_SETTINGS_MODULE=another.settings.file pytest -m "not mark"
```

Current marks are:

- search (tests that require Elastic Search)
- proxito (tests from the serve/proxito instance)

Tests without mark are from the main instance.

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/readthedocs/readthedocs.org>

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 first ensuring this project is set up locally according to the [Installation Guide](#). Follow the instructions just up to the point of activating the virtual environment and then continue here.

Next, install the documentation dependencies using `pip` (make sure you are inside of the virtual environment):

```
pip install -r requirements/local-docs-build.txt
```

This installs Sphinx, amongst other things.

After that is done, build the documentation by 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.

The brand

We are called **Read the Docs**. The *the* is not capitalized.

We do however use the acronym **RTD**.

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.

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` role and not a hyperlink.
- If you are cross-referencing to a section within our website, use the `ref` role with the label from the [autosectionlabel extension](#).

Front End Development

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.

Getting Started

You will need a working version of Node (tested with v10.17.0) 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.

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 working in. Currently, we still need a test runner that accumulates these files.

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.

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.

Design Documents

This is where we outline the design of major parts of our project. Generally this is only available for features that have been build in the recent past, but we hope to write more of them over time.

APIv3 Design Document

This document describes the design, some decisions already made and built (current Version 1 of APIv3) and an implementation plan for next Versions of APIv3.

APIv3 will be designed to be easy to use and useful to perform read and write operations as the main two goals.

It will be based on Resources as APIv2 but considering the `Project` resource as the main one, from where most of the endpoint will be based on it.

- *Goals*
- *Non-Goals*
- *Problems with APIv2*
- *Implementation stages*
- *Out of roadmap*
- *Nice to have*

Goals

- Easy to use for our users (access most of resources by `slug`)
- Useful to perform read and write operations
- Authentication/Authorization
 - Authentication based on scoped-tokens
 - Handle Authorization nicely using an abstraction layer
- Cover most useful cases:

- Integration on CI (check build status, trigger new build, etc)
- Usage from public Sphinx/MkDocs extensions
- Allow creation of flyout menu client-side
- Simplify migration from other services (import projects, create multiple redirects, etc)

Non-Goals

- Filter by arbitrary and non-useful fields
 - “Builds with `exit_code=1`”
 - “Builds containing `ERROR` on their output”
 - “Projects created after X datetime”
 - “Versions with tag `python`”
- Cover *all the actions* available from the WebUI

Problems with APIv2

There are several problem with our current APIv2 that we can list:

- No authentication
- It’s read-only
- Not designed for slugs
- Useful APIs not exposed (only for internal usage currently)
- Error reporting is a mess
- Relationships between API resources is not obvious
- Footer API endpoint returns HTML

Implementation stages

Version 1

The first implementation of APIv3 will cover the following aspects:

- Authentication
 - all endpoints require authentication via `Authorization:` request header
 - detail endpoints are available for all authenticated users
 - only Project’s maintainers can access listing endpoints
 - personalized listing
- Read and Write
 - edit attributes from Version (only `active` and `privacy_level`)
 - trigger Build for a specific Version
- Accessible by slug

- Projects are accessed by `slug`
- Versions are accessed by `slug`
- `/projects/` endpoint is the main one and all of the other are nested under it
- Builds are accessed by `id`, as exception to this rule
- access all (active/non-active) Versions of a Project by `slug`
- get latest Build for a Project (and Version) by `slug`
- filter by relevant fields
- Proper status codes to report errors
- Browse-able endpoints
 - browse is allowed hitting `/api/v3/projects/` as starting point
 - ability to navigate clicking on other resources under `_links` attribute
- Rate limited

Version 2

Note: This is currently implemented and live.

Second iteration will polish issues found from the first step, and add new endpoints to allow *import a project and configure it* without the needed of using the WebUI as a main goal.

After Version 2 is deployed, we will invite users that reach us as beta testers to receive more feedback and continue improving it by supporting more use cases.

This iteration will include:

- Minor changes to fields returned in the objects
- Import Project endpoint
- Edit Project attributes (“Settings” and “Advanced settings-Global settings” in the WebUI)
- Trigger Build for default version
- Allow CRUD for Redirect, Environment Variables and Notifications (`WebHook` and `EmailHook`)
- Create/Delete a Project as subproject of another Project
- Documentation

Version 3

Third iteration will implement granular permissions. Keeping in mind how Sphinx extension will use it:

- `sphinx-version-warning` needs to get *all active Versions of a Project*
- An extension that creates a landing page, will need *all the subprojects of a Project*

To fulfill these requirements, this iteration will include:

- Scope-based authorization token

Version 4

- Specific endpoint for our flyout menu (returning JSON instead of HTML)

Out of roadmap

These are some features that we may want to build at some point. Although, they are currently out of our near roadmap because they don't affect too many users, or are for internal usage only.

- CRUD for Domain
- Add User as maintainer
- Give access to a documentation page (`objects.inv`, `/design/core.html`)
- Internal Build process

Nice to have

- Request-ID header
- JSON minified by default (maybe with `?pretty=true`)
- JSON schema and validation with docs

In Doc Search UI

Giving readers the ability to easily search the information that they are looking for is important for us. We have already upgraded to the latest version of [Elasticsearch](#) and we plan to implement `search as you type` feature for all the documentations hosted by us. It will be designed to provide instant results as soon as the user starts typing in the search bar with a clean and minimal frontend. This design document aims to provide the details of it. This is a GSoC'19 project.

Warning: This design document details future features that are **not yet implemented**. To discuss this document, please get in touch in the [issue tracker](#).

The final result may look something like this:

Fig. 1: Short demo

Goals And Non-Goals

Project Goals

- Support a search-as-you-type/autocomplete interface.
- Support across all (or virtually all) Sphinx themes.
- Support for the JavaScript user experience down to IE11 or graceful degradation where we can't support it.
- Project maintainers should have a way to opt-in/opt-out of this feature.

- (Optional) Project maintainers should have the flexibility to change some of the styles using custom CSS and JS files.

Non-Goals

- For the initial release, we are targeting only Sphinx documentations as we don't index MkDocs documentations to our Elasticsearch index.

Existing Search Implementation

We have a detailed documentation explaining the underlying architecture of our search backend and how we index documents to our Elasticsearch index. You can read about it [here](#).

Proposed Architecture for In-Doc Search UI

Frontend

Technologies

Frontend is to designed in a theme agnostics way. For that, we explored various libraries which may be of use but none of them fits our needs. So, we might be using vanilla JavaScript for this purpose. This will provide us some advantages over using any third party library:

- Better control over the DOM.
- Performance benefits.

Proposed Architecture

We plan to select the search bar, which is present in every theme, and use the `querySelector()` method of JavaScript. Then add an event listener to it to listen for the changes and fire a search query to our backend as soon as there is any change. Our backend will then return the suggestions, which will be shown to the user in a clean and minimal UI. We will be using `document.createElement()` and `node.removeChild()` method provided by JavaScript as we don't want empty `<div>` hanging out in the DOM.

We have a few ways to include the required JavaScript and CSS files in all the projects:

- Add CSS into `readthedocs-doc-embed.css` and JS into `readthedocs-doc-embed.js` and it will get included.
- Package the in-doc search into it's own self-contained CSS and JS files and include them in a similar manner to `readthedocs-doc-embed.*`.
- It might be possible to package up the in-doc CSS/JS as a sphinx extension. This might be nice because then it's easy to enable it on a per-project basis. When we are ready to roll it out to a wider audience, we can make a decision to just turn it on for everybody (put it in [here](#)) or we could enable it as an opt-in feature like the [404 extension](#).

UI/UX

We have two ways which can be used to show suggestions to the user.

- Show suggestions below the search bar.
- Open a full page search interface when the user click on search field.

Backend

We have a few options to support `search as you type` feature, but we need to decide that which option would be best for our use-case.

Edge NGram Tokenizer

- Pros
 - More effective than Completion Suggester when it comes to autocompleting words that can appear in any order.
 - It is considerable fast because most of the work is being done at index time, hence the time taken for autocompletion is reduced.
 - Supports highlighting of the matching terms.
- Cons
 - Requires greater disk space.

Completion Suggester

- Pros
 - Really fast as it is optimized for speed.
 - Does not require large disk space.
- Cons
 - Matching always starts at the beginning of the text. So, for example, “Hel” will match “Hello, World” but not “World Hello”.
 - Highlighting of the matching words is not supported.
 - According to the official docs for Completion Suggester, fast lookups are costly to build and are stored in-memory.

Milestones

Milestone	Due Date
A local implementation of the project.	12th June, 2019
In-doc search on a test project hosted on Read the Docs using the RTD Search API.	20th June, 2019
In-doc search on docs.readthedocs.io.	20th June, 2019
Friendly user trial where users can add this on their own docs.	5th July, 2019
Additional UX testing on the top-10 Sphinx themes.	15th July, 2019
Finalize the UI.	25th July, 2019
Improve the search backend for efficient and fast search results.	10th August, 2019

Open Questions

- Should we rely on jQuery, any third party library or pure vanilla JavaScript?
- Are the subprojects to be searched?
- Is our existing Search API is sufficient?
- Should we go for edge ngrams or completion suggerter?

Organizations

Currently we don't support organizations in the community site (a way to group different projects), we only support individual accounts.

Several integrations that we support like GitHub and Bitbucket have organizations, where users group their repositories and manage them in groups rather than individually.

Why move organizations in the community site?

We support organizations in the commercial site, having no organizations in the community site makes the code maintenance difficult for Read the Docs developers. Having organizations in the community site will make the differences between both more easy to manage.

Users from the community site can have organizations in external sites from where we import their projects (like GitHub, Gitlab). Currently users have all projects from different organizations in their account. Having not a clear way to group/separate those.

We are going to first move the code, and after that enable the feature on the community site.

How are we going to support organizations?

Currently only users can own projects in the community site. With organizations this is going to change to: Users and organizations can own projects.

With this, the migration process would be straightforward for the community site.

For the commercial site we are only to allow organizations to own projects for now (since the we have only subscriptions per organizations).

What features of organizations are we going to support?

We have the following features in the commercial site that we don't have on the community site:

- Owners
- Teams
- Permissions
- Subscriptions

Owners should be included to represent owners of the current organization.

Teams, this is also handy to manage access to different projects under the same organization.

Permissions, currently we have two type of permissions for teams: admin and read only. Read only permissions doesn't make sense in the community site since we only support public projects/versions (we do support private versions now, but we are planning to remove those). So, we should only support admin permissions for teams.

Subscriptions, this is only valid for the corporate site, since we don't charge for use in the community site.

How to migrate current projects

Since we are not replacing the current implementation, we don't need to migrate current projects from the community site nor from the corporate site.

How to migrate the organizations app

The migration can be split in:

1. Remove/simplify code from the organizations app on the corporate site.
2. Isolate/separate models and code that isn't going to be moved.
3. Start by moving the models, managers, and figure out how to handle migrations.
4. Move the rest of the code as needed.
5. Activate organizations app on the community site.
6. Integrate the code from the community site to the new code.
7. UI changes

We should start by removing unused features and dead code from the organizations in the corporate site, and simplify existing code if possible (some of this was already done).

Isolate/separate the models to be moved from the ones that aren't going to be moved. We should move the models that aren't going to be moved to another app.

- Plan
- PlanFeature
- Subscription

This app can be named *subscriptions*. We can get around the table names and migrations by setting the explicitly the table name to `organizations_<model>`, and doing a fake migration. Following suggestions in <https://stackoverflow.com/questions/48860227/moving-multiple-models-from-one-django-app-to-another>, that way we avoid having any downtime during the migration and any inconvenient caused from renaming the tables manually.

Code related to subscriptions should be moved out from the organizations app.

After that, it should be easier to move the organizations *app* (or part of it) to the community site (and no changes to table names would be required).

We start by moving the models.

- Organization
- OrganizationOwner
- Team
- TeamInvite
- TeamMember

Migrations aren't moved, since all current migrations depend on other models that aren't going to be moved. In the community site we run an initial migration, for the corporate site we run a fake migration. The migrations left from the commercial site can be removed after that.

For managers and queriesets that depend on subscriptions, we can use our pattern to make overridable classes (inheriting from `SettingsOverrideObject`).

Templates, urls, views, forms, notifications, signals, tasks can be moved later (we just need to make use of the models from the `readthedocs.organizations` module).

If we decide to integrate organizations in the community site, we can add/move the UI elements and enable the app.

After the app is moved, we can move more code that depends on organizations to the community site.

Namespace

Currently we use the project's slug as namespace, in the commercial site we use the combination of `organization.slug + project.slug` as namespace, since in the corporate site we don't care so much about a unique namespace between all users, but a unique namespace per organization.

For the community site probably this approach isn't the best, since we always serve docs publicly from `slug.readthedocs.io`. And most of the users don't have a custom domain.

The corporate site will use `organization.slug + project.slug` as slug, And the community site will always use `project.slug` as slug, even if the project belongs to an organization.

We need to refactor the way we get the namespace to be more easy to manage in both sites.

Future Changes

Changes that aren't needed immediately after the migration, but that should be done:

- UI for organizations in the community site.
- Add new endpoints to the API (v3 only).
- Make the relationship between the models `Organization` and `Project` one to many (currently many to many).

Design of Pull Request Builder

Background

This will focus on automatically building documentation for Pull Requests on Read the Docs projects. This is one of the most requested feature of Read the Docs. This document will serve as a design document for discussing how to implement this features.

Scope

- Making Pull Requests work like temporary `Version`
- Excluding PR Versions from Elasticsearch Indexing
- Adding a PR Builds Tab in the Project Dashboard
- Updating the Footer API

- Adding Warning Banner to Docs
- Serving PR Docs
- Excluding PR Versions from Search Engines
- Receiving `pull_request` webhook event from Github
- Fetching data from pull requests
- Storing PR Version build Data
- Creating PR Versions when a pull request is opened and Triggering a build
- Triggering Builds on new commits on a PR
- Status reporting to Github

Fetching Data from Pull Requests

We already get Pull request events from Github webhooks. We can utilize that to fetch data from pull requests. when a `pull_request` event is triggered we can fetch the data of that pull request. We can fetch the pull request by doing something similar to travis-ci. ie: `git fetch origin +refs/pull/<pr_number>/merge:`

Modeling Pull Requests as a Type of Version

Pull requests can be Treated as a Type of Temporary Version. We might consider adding a `VERSION_TYPES` to the `Version` model.

- If we go with `VERSION_TYPES` we can add something like `pull_request` alongside Tag and Branch.

We should add `Version` and `Build` Model Managers for PR and Regular Versions and Builds. The proposed names for PR and Regular Version and Build Mangers are `external` and `internal`.

We can then use `Version.internal.all()` to get all regular versions, `Version.external.all()` to get all PR versions.

We can then use `Build.internal.all()` to get all regular version builds, `Build.external.all()` to get all PR version builds.

Excluding PR Versions from Elasticsearch Indexing

We should exclude to PR Versions from being Indexed to Elasticsearch. We need to update the `queryset` to exclude PR Versions.

Adding a PR Builds Tab in the Project Dashboard

We can add a Tab in the project dashboard that will listout the PR Builds of that project. We can name it `PR Builds`.

Creating Versions for Pull Requests

If the Github webhook event is `pull_request` and action is `opened`, this means a pull request was opened in the projects repository. We can create a `Version` from the Payload data and trigger a initial build for the version. A version will be created whenever RTD receives an event like this.

Triggering Build for New Commits in a Pull Request

We might want to trigger a new build for the PR version if there is a new commit on the PR. If the Github webhook event is `pull_request` and action is `synchronize`, this means a new commit was added to the pull request.

Status Reporting to Github

We could send build status reports to Github. We could send if the build was Successful or Failed. We can also send the build URL. By this we could show if the build passed or failed on Github something like `travis-ci` does.

As we already have the `repo:status` scope on our OAuth App, we can send the status report to Github using the Github Status API.

Sending the status report would be something like this:

POST /repos/:owner/:repo/statuses/:sha

```
{
  "state": "success",
  "target_url": "<pr_build_url>",
  "description": "The build succeeded!",
  "context": "continuous-documentation/read-the-docs"
}
```

Storing Pull Request Docs

We need to think about how and where to store data after a PR Version build is finished. We can store the data in a blob storage.

Excluding PR Versions from Search Engines

We should Exclude the PR Versions from Search Engines, because it might cause problems for RTD users. As users might land to a pull request doc but not the original Project Docs. This will cause confusion for the users.

Serving PR Docs

We need to think about how we want to serve the PR Docs.

- We could serve the PR Docs from another Domain.
- We could serve the PR Docs using `<pr_number>` namespace on the same Domain.
 - Using `pr-<pr_number>` as the version slug `https://<project_slug>.readthedocs.io/<language_code>/pr-<pr_number>/`
 - Using `pr` subdomain `https://pr.<project_slug>.readthedocs.io/<pr_number>/`

Updating the Footer API

We need to update the Footer API to reflect the changes. We might want to have a way to show that if this is a PR Build on the Footer.

- For regular project docs we should remove the PR Versions from the version list of the Footer.

- We might want to send `is_pr` data with the Footer API response.

Adding Warning Banner to Docs

We need to add a warning banner to the PR Version Docs to let the users know that this is a Draft/PR version. We can use a sphinx extension that we will force to install on the PR Versions to add the warning banner.

Related Issues

- [Autobuild Docs for Pull Requests](#)
- [Add travis-ci style pull request builder](#)

Read the Docs data passed to Sphinx build context

Before calling `sphinx-build` to render your docs, Read the Docs injects some extra context in the templates by using the `html_context` Sphinx setting in the `conf.py` file. This extra context can be used to build some awesome features in your own theme.

Warning: This design document details future features that are **not yet implemented**. To discuss this document, please get in touch in the [issue tracker](#).

Note: The [Read the Docs Sphinx Theme](#) uses this context to add additional features to the built documentation.

Context injected

Here is the full list of values injected by Read the Docs as a Python dictionary. Note that this dictionary is injected under the main key `readthedocs`:

```
{
  'readthedocs': {
    'v1': {
      'version': {
        'id': int,
        'slug': str,
        'verbose_name': str,
        'identifier': str,
        'type': str,
        'build_date': str,
        'downloads': {
          'pdf': str,
          'htmlzip': str,
          'epub': str
        },
      },
      'links': [{
        'href': 'https://readthedocs.org/api/v2/version/{id}/',
        'rel': 'self'
      }],
    },
  },
}
```

(continues on next page)

(continued from previous page)

```

    },
    'project': {
        'id': int,
        'name': str,
        'slug': str,
        'description': str,
        'language': str,
        'canonical_url': str,
        'subprojects': [{
            'id': int,
            'name': str,
            'slug': str,
            'description': str,
            'language': str,
            'canonical_url': str,
            'links': [{
                'href': 'https://readthedocs.org/api/v2/project/{id}/',
                'rel': 'self'
            }]
        }]
    },
    'links': [{
        'href': 'https://readthedocs.org/api/v2/project/{id}/',
        'rel': 'self'
    }]
},
'sphinx': {
    'html_theme': str,
    'source_suffix': str
},
'analytics': {
    'user_analytics_code': str,
    'global_analytics_code': str
},
'vcs': {
    'type': str, # 'bitbucket', 'github', 'gitlab' or 'svn'
    'user': str,
    'repo': str,
    'commit': str,
    'version': str,
    'display': bool,
    'conf_py_path': str
},
'meta': {
    'API_HOST': str,
    'MEDIA_URL': str,
    'PRODUCTION_DOMAIN': str,
    'READTHEDOCS': True
}
}
}

```

Warning: Read the Docs passes information to `sphinx-build` that may change in the future (e.g. at the moment of building the version 0.6 this was the latest but then 0.7 and 0.8 were added to the project and also built under Read the Docs) so it's your responsibility to use this context in a proper way.

In case you want *fresh data* at the moment of reading your documentation, you should consider using the [Read the Docs Public API](#) via Javascript.

Using Read the Docs context in your theme

In case you want to access to this data from your theme, you can use it like this:

```
{% if readthedocs.v1.vcs.type == 'github' %}
  <a href="https://github.com/{{ readthedocs.v1.vcs.user }}/{{ readthedocs.v1.vcs.
↪repo }}"
  /blob/{{ readthedocs.v1.vcs.version }}/{{ readthedocs.v1.vcs.conf_py_path }}/{{ _
↪pagename }}.rst">
    Show on GitHub</a>
{% endif %}
```

Note: In this example, we are using `pagename` which is a Sphinx variable representing the name of the page you are on. More information about Sphinx variables can be found in the [Sphinx documentation](#).

Customizing the context

In case you want to add some extra context you will have to declare your own `html_context` in your `conf.py` like this:

```
html_context = {
    'author': 'My Name',
    'date': datetime.date.today().strftime('%d/%m/%y'),
}
```

and use it inside your theme as:

```
<p>This documentation was written by {{ author }} on {{ date }}.</p>
```

Warning: Take into account that the Read the Docs context is injected after your definition of `html_context` so, it's not possible to override Read the Docs context values.

YAML Configuration File

Background

The current YAML configuration file is in beta state. There are many options and features that it doesn't support yet. This document will serve as a design document for discuss how to implement the missing features.

Scope

- Finish the spec to include all the missing options

- Have consistency around the spec
- Proper documentation for the end user
- Allow to specify the spec's version used on the YAML file
- Collect/show metadata about the YAML file and build configuration
- Promote the adoption of the configuration file

RTD settings

No all the RTD settings are applicable to the YAML file, others are applicable for each build (or version), and others for the global project.

Not applicable settings

Those settings can't be on the YAML file because: may depend for the initial project setup, are planned to be removed, security and privacy reasons.

- Project Name
- Repo URL
- Repo type
- Privacy level (this feature is planned to be removed¹)
- Project description (this feature is planned to be removed²)
- Single version
- Default branch
- Default version
- Domains
- Active versions
- Translations
- Subprojects
- Integrations
- Notifications
- Language
- Programming Language
- Project homepage
- Tags
- Analytics code
- Global redirects

¹ <https://github.com/readthedocs/readthedocs.org/issues/2663>

² <https://github.com/readthedocs/readthedocs.org/issues/3689>

Global settings

To keep consistency with the per-version settings and avoid confusion, this settings will not be stored in the YAML file and will be stored in the database only.

Local settings

Those configurations will be read from the YAML file in the current version that is being built.

Several settings are already implemented and documented on <https://docs.readthedocs.io/en/latest/yaml-config.html>. So, they aren't covered with much detail here.

- Documentation type
- Project installation (virtual env, requirements file, sphinx configuration file, etc)
- Additional builds (pdf, epub)
- Python interpreter
- Per-version redirects

Configuration file

Format

The file format is based on the YAML spec 1.2³ (latest version on the time of this writing).

The file must be on the root directory of the repository, and must be named as:

- `readthedocs.yml`
- `readthedocs.yaml`
- `.readthedocs.yml`
- `.readthedocs.yaml`

Conventions

The spec of the configuration file must use this conventions.

- Use `[]` to indicate an empty list
- Use `null` to indicate a null value
- Use `all` (internal string keyword) to indicate that all options are included on a list with predetermined choices.
- Use `true` and `false` as only options on boolean fields

Spec

The current spec is documented on <https://docs.readthedocs.io/en/latest/yaml-config.html>. It will be used as base for the future spec. The spec will be written using a validation schema such as <https://json-schema-everywhere.github.io/yaml>.

³ <https://yaml.org/spec/1.2/spec.html>

Versioning the spec

The version of the spec that the user wants to use will be specified on the YAML file. The spec only will have mayor versions (1.0, not 1.2)⁴. For keeping compatibility with older projects using a configuration file without a version, the latest compatible version will be used (1.0).

Adoption of the configuration file

When a user creates a new project or it's on the settings page, we could suggest her/him an example of a functional configuration file with a minimal setup. And making clear where to put global configurations.

For users that already have a project, we can suggest him/her a configuration file on each build based on the current settings.

Configuration file and database

The settings used in the build from the configuration file (and other metadata) needs to be stored in the database, this is for later usage only, not to populate existing fields.

The build process

- The repository is updated
- Checkout to the current version
- Retrieve the settings from the database
- Try to parse the YAML file (the build fails if there is an error)
- Merge the both settings (YAML file and database)
- The version is built according to the settings
- The settings used to build the documentation can be seen by the user

Dependencies

Current repository which contains the code related to the configuration file: <https://github.com/readthedocs/readthedocs-build>

Footnotes

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.

⁴ <https://github.com/readthedocs/readthedocs.org/issues/3806>

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](#).

Note: The size of the Docker images is around 5 to 9 GB.

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:latest`. 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](#).

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: `{'memory': '200m', 'time': 600}`

DOCKER_IMAGE Tag of a Docker image to use as a base image.

Default: `readthedocs/build:latest`

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: `unix:///var/run/docker.sock`

DOCKER_VERSION Version of the API to use for the Docker API client.

Default: `auto`

Local development

On Linux development environments, it's likely that your UID and GID do not match the `docs` user that is set up as the default user for builds. In this case, it's necessary to make a new image that overrides the UID and GID for the normal container user:

```
contrib/docker_build.sh latest
```

This will create a new image, `readthedocs/build-dev:latest`. To build a different image, you can instead specify a version to build:

```
contrib/docker_build.sh 5.0
```

This will create a new image, `readthedocs/build-dev:5.0`.

You can set a `local_settings.py` option to automatically patch the image names to the development image names that are built here:

DOCKER_USE_DEV_IMAGES If set to `True`, replace the normal Docker image name used in building `readthedocs/build` with the image name output for these commands, `readthedocs/build-dev`.

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.

Nginx

We handle a couple of different types of requests in nginx:

- Requests to a `readthedocs.io` subdomain
- Requests to a custom domain

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;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
```

(continues on next page)

(continued from previous page)

```
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.

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 `fabfile` 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 `fabfile`. When we build the `fabfile` docs, we create a symlink for all domains that have pointed at `fabfile` 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.

Interesting Settings

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.

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.

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`.

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.

MULTIPLE_APP_SERVERS

Default: `['celery']`

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.

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.

INDEX_ONLY_LATEST

Default: `None`

In search, only index the `latest` version of a Project.

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.

PUBLIC_DOMAIN

Default: `None`

A special domain for serving public documentation. If set, public docs will be linked here instead of the `PRODUCTION_DOMAIN`.

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.

ALLOW_ADMIN

Default: True

Whether to include `django.contrib.admin` in the URL's.

RTD_BUILD_MEDIA_STORAGE

Default: `readthedocs.builds.storage.BuildMediaFileSystemStorage`

Use this storage class to upload build artifacts to cloud storage (S3, Azure storage). This should be a dotted path to the relevant class (eg. `'path.to.MyBuildMediaStorage'`). Your class should mixin `readthedocs.builds.storage.BuildMediaStorageMixin`.

ELASTICSEARCH_DSL

Default:

```
{
  'default': {
    'hosts': '127.0.0.1:9200'
  },
}
```

Settings for elasticsearch connection. This settings then pass to `elasticsearch-dsl-py.connections.configure`

ES_INDEXES

Default:

```
{
  'project': {
    'name': 'project_index',
    'settings': {'number_of_shards': 5,
                 'number_of_replicas': 0
    },
  },
  'page': {
    'name': 'page_index',
    'settings': {
      'number_of_shards': 5,
      'number_of_replicas': 0,
    },
  },
}
```

Define the elasticsearch name and settings of all the index separately. The key is the type of index, like `project` or `page` and the value is another dictionary containing `name` and `settings`. Here the `name` is the index name and the `settings` is used for configuring the particular index.

ES_TASK_CHUNK_SIZE

Default: 100

The maximum number of data send to each elasticsearch indexing celery task. This has been used while running `elasticsearch_reindex` management command.

ES_PAGE_IGNORE_SIGNALS

Default: `False`

This settings is used to determine whether to index each page separately into elasticsearch. If the setting is `True`, each HTML page will not be indexed separately but will be indexed by bulk indexing.

ELASTICSEARCH_DSL_AUTOSYNC

Default: `True`

This setting is used for automatically indexing objects to elasticsearch. `False` by default in development so it is possible to create project and build documentations without having elasticsearch.

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

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')),  
    # 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.

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.  
{% 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.

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 `gettext()` 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 gettext 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 `gettext()` 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, `gettext_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 gettext_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()`.

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 [Invoke](#) with a `tasks.py` file to follow this process:

1. Update files and push sources (English) to Transifex:

```
$ invoke l10n.push
```

2. Pull the updated translations from Transifex:

```
$ invoke l10n.pull
```

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.


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 Docs'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.

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.


Read the Docs
Go
Dashboard
Log Out

Header 1.

Header 2.

Header 3.

Header 4.

Header 5.

Paragraph. *Aside*.

Paragraph with [link](#).

Paragraph with highlighted text.

Long form text. Read the Docs hosts documentation, making it fully *searchable* and easy to find. You can import your docs using any major version control system, including Mercurial, Git, Subversion, and Bazaar. We support [links](#) so your docs get built when you commit code. There's also support for versioning so you can build docs from tags and branches of your code in your repository. A [website](#) is available.

It's free and simple. Read the [Getting Started](#) guide to get going!

Table header	Table header 2
Table element.	Table element 2.
Table element.	Table element 2.

Form Paragraph.

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.

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.

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).

Brand Guidelines

Find our branding guidelines in our guidelines documentation: <https://read-the-docs-guidelines.readthedocs-hosted.com>.

4.1.5 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 dataclasses*
- *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.

4.1.6 Helping on translations

If you wish to contribute translations, please do so on [Transifex](#).

4.2 Roadmap

4.2.1 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 issue 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 version). 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 priority, 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.

4.2.2 Current roadmap

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

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

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

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

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

Cleanup <https://github.com/readthedocs/readthedocs.org/milestone/10>

Refactoring <https://github.com/readthedocs/readthedocs.org/milestone/34>

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

4.3 Google Summer of Code

Warning: Read the Docs will not be participating in the Google Summer of Code in 2020. We hope to return to the program in the future, and appreciate the interest everyone has shown.

Thanks for your interest in Read the Docs! Please follow the instructions in [Getting Started](#), as a good place to start. **Contacting us will not increase your chance of being accepted, but opening Pull Requests with docs and tests will.**

You can see our [Projects from previous years](#) for the work that students have done in the past.

4.3.1 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!

4.3.2 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.

4.3.3 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.

4.3.4 Mentors

Currently we have a few folks signed up:

- Eric Holscher
- Manuel Kaufmann
- Anthony Johnson

- Safwan Rahman

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

4.3.5 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.

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.

Integration with OpenAPI/Swagger

Integrate the existing tooling around OpenAPI & Swagger into Sphinx and Read the Docs. This will include building some extensions that generate `reStructuredText`, and backend Django code that powers the frontend Javascript.

This could include:

- Building a live preview for testing an API in the documentation
- Taking a swagger YAML file and generating HTML properly with Sphinx
- Integration with our existing API to generate Swagger output

Build a new Sphinx theme

Sphinx v2 will introduce a new format for themes, supporting HTML5 and new markup. We are hoping to build a new Sphinx theme that supports this new structure.

This project would include:

- A large amount of design, including working with CSS & SASS
- Iterating with the community to build something that works well for a number of use cases

This is not as well defined as the other tasks, so would require a higher level of skill from an incoming student.

Better MkDocs integration

Currently we don't have a good integration with MkDocs as we do with Sphinx. And it's hard to maintain compatibility with new versions.

This project would include:

- Support the latest version of MkDocs
- Support downloads (#1939)
- Write a plugin to allow us to have more control over the build process (#4924)
- Support search (#1088)

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.

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 and 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.

Additional Ideas

We have some medium sized projects sketched out in our issue tracker with the tag *Feature*. 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.

4.3.6 Projects from previous years

- [Improved Search And Search As You Type](#) (2019)
- [Building Docs For Pull Requests](#) (2019)
- [Search Improvement](#) (2018)

4.3.7 Thanks

This page was heavily inspired by Mailman's similar [GSOC page](#). Thanks for the inspiration.

4.4 Code of Conduct

Like the technical community as a whole, the Read the Docs team and community is made up of a mixture of professionals 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](#).

4.5 Security

Security is very important to us at Read the Docs. We follow generally accepted industry standards to protect the personal information submitted to us, both during transmission and once we receive it. In the spirit of transparency, we are committed to responsible reporting and disclosure of security issues.

Contents

- [Account security](#)
- [Reporting a security issue](#)
- [PGP key](#)
- [Security issue archive](#)

4.5.1 Account security

- All traffic is encrypted in transit so your login is protected.
- Read the Docs stores only one-way hashes of all passwords. Nobody at Read the Docs has access to your passwords.
- Account login is protected from brute force attacks with rate limiting.

- While most projects and docs on Read the Docs are public, we treat your private repositories and private documentation as confidential and Read the Docs employees may only view them with your explicit permission in response to your support requests, or when required for security purposes.
- You can read more about account privacy in our [Privacy Policy](#).

4.5.2 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](#).

4.5.3 PGP key

You may use this PGP key to securely communicate with us and to verify signed messages you receive from us.

4.5.4 Security issue archive

Version 3.5.1

[Version 3.5.1](#) fixed an issue that affected projects with “prefix” or “sphinx” user-defined redirects. The issue allowed the creation of hyperlinks that looked like they would go to a documentation domain on Read the Docs (either `*.readthedocs.io` or a custom docs domain) but instead went to a different domain.

This issue was reported by Peter Thomassen and the [dsec.io](#) DNS security project and was funded by [SSE](#).

Version 3.2.0

[Version 3.2.0](#) resolved an issue where a specially crafted request could result in a DNS query to an arbitrary domain.

This issue was found by [Cyber Smart Defence](#) who reported it as part of a security audit to a firm running a local installation of Read the Docs.

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.

4.6 Privacy Policy

Effective date: **September 30, 2019**

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

4.6.1 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!

4.6.2 Our services

Read the Docs is made up of:

readthedocs.org (“**Read the Docs Community**”) This is a website aimed at documentation authors and project maintainers writing and distributing technical documentation. This Privacy Policy applies to this site in full without reservation.

readthedocs.com (“**Read the Docs for Business**”) This website is a commercial hosted offering for hosting private documentation for corporate clients. This Privacy Policy applies to this site in full without reservation.

readthedocs.io, readthedocs-hosted.com, and other domains (“**Documentation Sites**”) These websites are where Read the Docs hosts documentation (“*User-Generated Content*”) 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.

All use of Read the Docs is subject to this Privacy Policy, together with our *Terms of service*.

4.6.3 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 information?

- 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](#).

4.6.4 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.

Read the Docs for Business may host [private projects](#) which we treat as confidential and we only access them for support reasons, with your consent, or if required to for security reasons

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.

4.6.5 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 Read the Docs for Business. 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.

4.6.6 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 Read the Docs Community and Read the Docs for Business. 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*.

4.6.7 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.

4.6.8 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.

4.6.9 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.

4.6.10 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.

4.6.11 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 Read the Docs Community, Read the Docs for Business, and Documentation Sites are deleted after 10 days barring legal obligations.

4.6.12 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.

4.6.13 Contacting Read the Docs

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

4.7 Read the Docs Terms of Service

Effective date: **September 30, 2019**

Thank you for using Read the Docs! We're happy you're here. Please read this Terms of Service agreement carefully before accessing or using Read the Docs. Because it is such an important contract between us and our users, we have tried to make it as clear as possible. For your convenience, we have presented these terms in a short non-binding summary followed by the full legal terms.

Table of contents

- *Definitions*
- *Account terms*
- *Acceptable use*
- *User-Generated Content*
- *Private projects*
- *Copyright infringement and DMCA policy*

- *Intellectual property notice*
- *API terms*
- *Additional terms for Documentation Sites*
- *Third party applications*
- *Advertising on Documentation Sites*
- *Payment*
- *Cancellation and termination*
- *Communications with Read the Docs*
- *Disclaimer of warranties*
- *Limitation of liability*
- *Release and indemnification*
- *Changes to these terms*
- *Miscellaneous*

4.7.1 Definitions

Short version: *We use these basic terms throughout the agreement, and they have specific meanings. You should know what we mean when we use each of the terms. There's not going to be a test on it, but it's still useful information.*

1. The “Agreement” refers, collectively, to all the terms, conditions, notices contained or referenced in this document (the “Terms of Service” or the “Terms”) and all other operating rules, policies (including our [Privacy Policy](#)) and procedures that we may publish from time to time on our sites.
2. Our “Service” or “Services” refers to the applications, software, products, and services provided by Read the Docs (see [Our services](#)).
3. The “Website” or “Websites” refers to Read the Docs’ websites located at [readthedocs.org](#), [readthedocs.com](#), Documentation Sites, and all content, services, and products provided by Read the Docs at or through those Websites.
4. “The User,” “You,” and “Your” refer to the individual person, company, or organization that has visited or is using ours Websites or Services; that accesses or uses any part of the Account; or that directs the use of the Account in the performance of its functions. A User must be at least 13 years of age.
5. “Read the Docs,” “We,” and “Us” refer to Read the Docs, Inc., as well as our affiliates, directors, subsidiaries, contractors, licensors, officers, agents, and employees.
6. “Content” refers to content featured or displayed through the Websites, including without limitation text, data, articles, images, photographs, graphics, software, applications, designs, features, and other materials that are available on our Websites or otherwise available through our Services. “Content” also includes Services. “User-Generated Content” is Content, written or otherwise, created or uploaded by our Users. “Your Content” is Content that you create or own.
7. An “Account” represents your legal relationship with Read the Docs. A “User Account” represents an individual User’s authorization to log in to and use the Service and serves as a User’s identity on Read the Docs. “Organizations” are shared workspaces that may be associated with a single entity or with one or more Users where multiple Users can collaborate across many projects at once. A User Account can be a member of any number of Organizations.

8. “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. Our [Privacy Policy](#) goes into more details on User Personal Information, what data Read the Docs collects, and why we collect it.

Our services

Read the Docs is made up of the following Websites:

readthedocs.org (“Read the Docs Community”) This Website is used by documentation authors and project maintainers for writing and distributing technical documentation.

readthedocs.com (“Read the Docs for Business”) This Website is a commercial hosted offering for hosting private documentation for corporate clients.

readthedocs.io, readthedocs-hosted.com, and other domains (“Documentation Sites”) These Websites are where Read the Docs hosts *User-Generated Content* on behalf of documentation authors.

4.7.2 Account terms

Short version: *User Accounts and Organizations have different administrative controls; a human must create your Account; you must be 13 or over; and you must provide a valid email address. You alone are responsible for your Account and anything that happens while you are signed in to or using your Account. You are responsible for keeping your Account secure.*

Account controls

Users Subject to these Terms, you retain ultimate administrative control over your User Account and the Content within it.

Organizations The “owner” of an Organization that was created under these Terms has ultimate administrative control over that Organization and the Content within it. Within our Services, an owner can manage User access to the Organization’s data and projects. An Organization may have multiple owners, but there must be at least one User Account designated as an owner of an Organization. If you are the owner of an Organization under these Terms, we consider you responsible for the actions that are performed on or through that Organization.

Required information

You must provide a valid email address in order to complete the signup process. Any other information requested, such as your real name, is optional, unless you are accepting these terms on behalf of a legal entity (in which case we need more information about the legal entity) or if you opt for a *paid Account*, in which case additional information will be necessary for billing purposes.

Account requirements

We have a few simple rules for User Accounts on Read the Docs’ Services.

- You must be a human to create an Account. Accounts registered by “bots” or other automated methods are not permitted. We do permit machine accounts:
- A machine account is an Account set up by an individual human who accepts the Terms on behalf of the Account, provides a valid email address, and is responsible for its actions. A machine account is used exclusively for

performing automated tasks. Multiple users may direct the actions of a machine account, but the owner of the Account is ultimately responsible for the machine's actions.

- You must be age 13 or older. While we are thrilled to see brilliant young developers and authors get excited by learning to program, we must comply with United States law. Read the Docs does not target our Services to children under 13, and we do not permit any Users under 13 on our Service. If we learn of any User under the age of 13, we will have to close your account. If you are a resident of a country outside the United States, your country's minimum age may be older; in such a case, you are responsible for complying with your country's laws.
- You may not use Read the Docs in violation of export control or sanctions laws of the United States or any other applicable jurisdiction. You may not use Read the Docs if you are or are working on behalf of a [Specially Designated National \(SDN\)](#) or a person subject to similar blocking or denied party prohibitions administered by a U.S. government agency. Read the Docs may allow persons in certain sanctioned countries or territories to access certain Read the Docs services pursuant to U.S. government authorizations.

User Account security

You are responsible for keeping your Account secure while you use our Service.

- You are responsible for all content posted and activity that occurs under your Account.
- You are responsible for maintaining the security of your Account and password. Read the Docs cannot and will not be liable for any loss or damage from your failure to comply with this security obligation.
- You will promptly *notify Read the Docs* if you become aware of any unauthorized use of, or access to, our Services through your Account, including any unauthorized use of your password or Account.

Additional terms

In some situations, third parties' terms may apply to your use of Read the Docs. For example, you may be a member of an organization on Read the Docs with its own terms or license agreements; or you may download an application that integrates with Read the Docs. Please be aware that while these Terms are our full agreement with you, other parties' terms govern their relationships with you.

4.7.3 Acceptable use

Short version: *Read the Docs hosts a wide variety of collaborative projects from all over the world, and that collaboration only works when our users are able to work together in good faith. While using the service, you must follow the terms of this section, which include some restrictions on content you can post, conduct on the service, and other limitations. In short, be excellent to each other.*

Your use of our Websites and Services must not violate any applicable laws, including copyright or trademark laws, export control or sanctions laws, or other laws in your jurisdiction. You are responsible for making sure that your use of the Service is in compliance with laws and any applicable regulations.

4.7.4 User-Generated Content

Short version: *You own content you create, but you allow us certain rights to it, so that we can display and share the content and documentation you post. You still have control over your content, and responsibility for it, and the rights you grant us are limited to those we need to provide the service. We have the right to remove content or close Accounts if we need to.*

Responsibility for User-Generated Content

You may create or upload User-Generated Content while using the Service. You are solely responsible for the content of, and for any harm resulting from, any User-Generated Content that you post, upload, link to or otherwise make available via the Service, regardless of the form of that Content. We are not responsible for any public display or misuse of your User-Generated Content.

Read the Docs may remove Content

We do not pre-screen User-Generated Content, but we have the right (though not the obligation) to refuse or remove any User-Generated Content that, in our sole discretion, violates any Read the Docs terms or policies.

Ownership of Content, right to post, and license grants

You retain ownership of and responsibility for Your Content. If you're posting anything you did not create yourself or do not own the rights to, you agree that you are responsible for any Content you post; that you will only submit Content that you have the right to post; and that you will fully comply with any third party licenses relating to Content you post.

Because you retain ownership of and responsibility for Your Content, we need you to grant us — and other Read the Docs Users — certain legal permissions, listed below (in [License grant to us](#), [License grant to other users](#) and [Moral rights](#)). These license grants apply to Your Content. If you upload Content that already comes with a license granting Read the Docs the permissions we need to run our Service, no additional license is required. You understand that you will not receive any payment for any of the rights granted. The licenses you grant to us will end when you remove Your Content from our servers.

License grant to us

We need the legal right to do things like host Your Content, publish it, and share it. You grant us and our legal successors the right to store, parse, and display Your Content, and make incidental copies as necessary to render the Website and provide the Service. This includes the right to do things like copy it to our database and make backups; show it to you and other users; parse it into a search index or otherwise analyze it on our servers; share it with other users; and perform it, in case Your Content is something like music or video.

This license does not grant Read the Docs the right to sell Your Content or otherwise distribute or use it outside of our provision of the Service.

License grant to other users

Any User-Generated Content you post publicly may be viewed by others. By setting your projects to be viewed publicly, you agree to allow others to view your Content.

On Read the Docs Community, all Content is public.

Moral rights

You retain all moral rights to Your Content that you upload, publish, or submit to any part of our Services, including the rights of integrity and attribution. However, you waive these rights and agree not to assert them against us, to enable us to reasonably exercise the rights granted in [License grant to us](#), but not otherwise.

To the extent this agreement is not enforceable by applicable law, you grant Read the Docs the rights we need to use Your Content without attribution and to make reasonable adaptations of Your Content as necessary to render our Websites and provide our Services.

4.7.5 Private projects

Short version: *You may connect Read the Docs for Business to your private repositories or host documentation privately. We treat the content of these private projects as confidential, and we only access it for support reasons, with your consent, or if required to for security reasons.*

Confidentiality of private projects

Read the Docs considers the contents of private projects to be confidential to you. Read the Docs will protect the contents of private projects from unauthorized use, access, or disclosure in the same manner that we would use to protect our own confidential information of a similar nature and in no event with less than a reasonable degree of care.

Access

Read the Docs employees may only access the content of your private projects in the following situations:

- With your consent and knowledge, for support reasons. If Read the Docs accesses a private project for support reasons, we will only do so with the owner's consent and knowledge.
- When access is required for security reasons, including when access is required to maintain ongoing confidentiality, integrity, availability and resilience of Read the Docs' systems and Services.

Exclusions

If we have reason to believe the contents of a private project are in violation of the law or of these Terms, we have the right to access, review, and remove them. Additionally, we may be *compelled by law* to disclose the contents of your private projects.

4.7.6 Copyright infringement and DMCA policy

If you believe that content on our website violates your copyright or other rights, please contact us in accordance with our *Digital Millennium Copyright Act Policy*. There may be legal consequences for sending a false or frivolous takedown notice. Before sending a takedown request, you must consider legal uses such as fair use and licensed uses.

We will terminate the Accounts of repeat infringers of this policy.

4.7.7 Intellectual property notice

Short version: *We own the Service and all of our Content. In order for you to use our Content, we give you certain rights to it, but you may only use our Content in the way we have allowed.*

Read the Docs' rights to content

Read the Docs and our licensors, vendors, agents, and/or our content providers retain ownership of all intellectual property rights of any kind related to our Websites and Services. We reserve all rights that are not expressly granted to you under this Agreement or by law.

Read the Docs trademarks and logos

If you'd like to use Read the Docs's trademarks, you must follow all of our [trademark guidelines](#).

4.7.8 API terms

Short version: *You agree to these Terms of Service, plus this Section, when using any of Read the Docs' APIs (Application Provider Interface), including use of the API through a third party product that accesses Read the Docs.*

No abuse or overuse of the API

Abuse or excessively frequent requests to Read the Docs via the API may result in the temporary or permanent suspension of your Account's access to the API. Read the Docs, in our sole discretion, will determine abuse or excessive usage of the API. We will make a reasonable attempt to warn you via email prior to suspension.

You may not share API tokens to exceed Read the Docs' rate limitations.

You may not use the API to download data or Content from Read the Docs for spamming purposes, including for the purposes of selling Read the Docs users' personal information, such as to recruiters, headhunters, and job boards.

All use of the Read the Docs API is subject to these Terms of Service and our [Privacy Policy](#).

Read the Docs may offer subscription-based access to our API for those Users who require high-throughput access or access that would result in resale of Read the Docs' Service.

4.7.9 Additional terms for Documentation Sites

Short version: *Documentation Sites on Read the Docs are subject to certain rules, in addition to the rest of the Terms.*

Documentation Sites

Each Read the Docs Account comes with the ability to host Documentation Sites. This hosting service is intended to host static web pages for All Users. Documentation Sites are subject to some specific bandwidth and usage limits, and may not be appropriate for some high-bandwidth uses or other prohibited uses.

4.7.10 Third party applications

Short version: *You need to follow certain rules if you create an application for other Users.*

Creating applications

If you create a third-party application or other developer product that collects User Personal Information or User-Generated Content and integrates with the Service through Read the Docs' API, OAuth mechanism, or otherwise ("Developer Product"), and make it available for other Users, then you must comply with the following requirements:

- You must comply with this Agreement and our [Privacy Policy](#).
- Except as otherwise permitted, such as by law or by a license, you must limit your usage of the User Personal Information or User-Generated Content you collect to that purpose for which the User has authorized its collection.
- You must take all reasonable security measures appropriate to the risks, such as against accidental or unlawful destruction, or accidental loss, alteration, unauthorized disclosure or access, presented by processing the User Personal Information or User-Generated Content.
- You must not hold yourself out as collecting any User Personal Information or User-Generated Content on Read the Docs' behalf, and provide sufficient notice of your privacy practices to the User, such as by posting a privacy policy.

- You must provide Users with a method of deleting any User Personal Information or User-Generated Content you have collected through Read the Docs after it is no longer needed for the limited and specified purposes for which the User authorized its collection, except where retention is required by law or otherwise permitted, such as through a license.

4.7.11 Advertising on Documentation Sites

Short version: *We do not generally prohibit use of Documentation Sites for advertising. However, we expect our users to follow certain limitations, so Read the Docs does not become a spam haven. No one wants that.*

Our advertising

We host advertising on Documentation Sites on Read the Docs Community. 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](#).

Acceptable advertising on Documentation Sites

We offer Documentation Sites primarily as a showcase for personal and organizational projects. Some project monetization efforts are permitted on Documentation Sites, such as donation buttons and crowdfunding links.

Spamming and inappropriate use of Read the Docs

Advertising Content, like all Content, must not violate the law or these Terms of Use, for example through excessive bulk activity such as spamming. We reserve the right to remove any projects that, in our sole discretion, violate any Read the Docs terms or policies.

4.7.12 Payment

Short version: *You are responsible for any fees associated with your use of Read the Docs. We are responsible for communicating those fees to you clearly and accurately, and letting you know well in advance if those prices change.*

Pricing

Our pricing and payment terms are available at <https://readthedocs.com/pricing/>. If you agree to a subscription price, that will remain your price for the duration of the payment term; however, prices are subject to change at the end of a payment term.

Upgrades, downgrades, and changes

- We will immediately bill you when you upgrade from the free plan to any paying plan (either Read the Docs for Business or a Gold membership).
- If you change from a monthly billing plan to a yearly billing plan, Read the Docs will bill you for a full year at the next monthly billing date.
- If you upgrade to a higher level of service, we will bill you for the upgraded plan immediately.
- You may change your level of service at any time by going into your billing settings. If you choose to downgrade your Account, you may lose access to Content, features, or capacity of your Account.

Billing schedule; no refunds

- For monthly or yearly payment plans, the Service is billed in advance on a monthly or yearly basis respectively and is non-refundable. There will be no refunds or credits for partial months of service, downgrade refunds, or refunds for months unused with an open Account; however, the service will remain active for the length of the paid billing period.
- Exceptions to these rules are at Read the Docs' sole discretion.

Authorization

By agreeing to these Terms, you are giving us permission to charge your on-file credit card, PayPal account, or other approved methods of payment for fees that you authorize for Read the Docs.

Responsibility for payment

You are responsible for all fees, including taxes, associated with your use of the Service. By using the Service, you agree to pay Read the Docs any charge incurred in connection with your use of the Service. If you dispute the matter, [contact us](#). You are responsible for providing us with a valid means of payment for paid Accounts. Free Accounts are not required to provide payment information.

4.7.13 Cancellation and termination

Short version: *You may close your Account at any time. If you do, we'll treat your information responsibly.*

Account cancellation

It is your responsibility to properly cancel your Account with Read the Docs. You can cancel your Account at any time by going into your Settings in the global navigation bar at the top of the screen. We are not able to cancel Accounts in response to an email or phone request.

Upon cancellation

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 and the Content of your repositories within 90 days of cancellation or termination. This information can not be recovered once your Account is cancelled.

Read the Docs may terminate

Read the Docs has the right to suspend or terminate your access to all or any part of the Website at any time, with or without cause, with or without notice, effective immediately. Read the Docs reserves the right to refuse service to anyone for any reason at any time.

Survival

All provisions of this Agreement which, by their nature, should survive termination *will* survive termination – including, without limitation: ownership provisions, warranty disclaimers, indemnity, and limitations of liability.

4.7.14 Communications with Read the Docs

Short version: *We use email and other electronic means to stay in touch with our users.*

Electronic communication required

For contractual purposes, you:

1. Consent to receive communications from us in an electronic form via the email address you have submitted or via the Service
2. Agree that all Terms of Service, agreements, notices, disclosures, and other communications that we provide to you electronically satisfy any legal requirement that those communications would satisfy if they were on paper. This section does not affect your non-waivable rights.

Legal notice to Read the Docs must be in writing

Communications made through email or Read the Docs' support system will not constitute legal notice to Read the Docs or any of its officers, employees, agents or representatives in any situation where notice to Read the Docs is required by contract or any law or regulation. Legal notice to Read the Docs must be in writing.

No phone support

Read the Docs only offers support via email, in-Service communications, and electronic messages. We do not offer telephone support.

4.7.15 Disclaimer of warranties

Short version: *We provide our service as is, and we make no promises or guarantees about this service. Please read this section carefully; you should understand what to expect.*

Read the Docs provides the Website and the Service “as is” and “as available,” without warranty of any kind. Without limiting this, we expressly disclaim all warranties, whether express, implied or statutory, regarding the Website and the Service including without limitation any warranty of merchantability, fitness for a particular purpose, title, security, accuracy and non-infringement.

Read the Docs does not warrant that the Service will meet your requirements; that the Service will be uninterrupted, timely, secure, or error-free; that the information provided through the Service is accurate, reliable or correct; that any defects or errors will be corrected; that the Service will be available at any particular time or location; or that the Service is free of viruses or other harmful components. You assume full responsibility and risk of loss resulting from your downloading and/or use of files, information, content or other material obtained from the Service.

4.7.16 Limitation of liability

Short version: *We will not be liable for damages or losses arising from your use or inability to use the Service or otherwise arising under this agreement. Please read this section carefully; it limits our obligations to you.*

You understand and agree that we will not be liable to you or any third party for any loss of profits, use, goodwill, or data, or for any incidental, indirect, special, consequential or exemplary damages, however arising, that result from:

- the use, disclosure, or display of your User-Generated Content;
- your use or inability to use the Service;

- any modification, price change, suspension or discontinuance of the Service;
- the Service generally or the software or systems that make the Service available;
- unauthorized access to or alterations of your transmissions or data;
- statements or conduct of any third party on the Service;
- any other user interactions that you input or receive through your use of the Service; or
- any other matter relating to the Service.

Our liability is limited whether or not we have been informed of the possibility of such damages, and even if a remedy set forth in this Agreement is found to have failed of its essential purpose. We will have no liability for any failure or delay due to matters beyond our reasonable control.

4.7.17 Release and indemnification

Short version: *You are responsible for your use of the service. If you harm someone else or get into a dispute with someone else, we will not be involved.*

If you have a dispute with one or more Users, you agree to release Read the Docs from any and all claims, demands and damages (actual and consequential) of every kind and nature, known and unknown, arising out of or in any way connected with such disputes.

You agree to indemnify us, defend us, and hold us harmless from and against any and all claims, liabilities, and expenses, including attorneys' fees, arising out of your use of the Website and the Service, including but not limited to your violation of this Agreement, provided that Read the Docs:

1. Promptly gives you written notice of the claim, demand, suit or proceeding
2. Gives you sole control of the defense and settlement of the claim, demand, suit or proceeding (provided that you may not settle any claim, demand, suit or proceeding unless the settlement unconditionally releases Read the Docs of all liability)
3. Provides to you all reasonable assistance, at your expense.

4.7.18 Changes to these terms

Short version: *We want our users to be informed of important changes to our terms, but some changes aren't that important — we don't want to bother you every time we fix a typo. So while we may modify this agreement at any time, we will notify users of any changes that affect your rights and give you time to adjust to them.*

We reserve the right, at our sole discretion, to amend these Terms of Service at any time and will update these Terms of Service in the event of any such amendments. We will notify our Users of material changes to this Agreement, such as price changes, at least 30 days prior to the change taking effect by posting a notice on our Website. For non-material modifications, your continued use of the Website constitutes agreement to our revisions of these Terms of Service.

We reserve the right at any time and from time to time to modify or discontinue, temporarily or permanently, the Website (or any part of it) with or without notice.

4.7.19 Miscellaneous

Governing law

Except to the extent applicable law provides otherwise, this Agreement between you and Read the Docs and any access to or use of our Websites or our Services are governed by the federal laws of the United States of America and the laws of the State of Oregon, without regard to conflict of law provisions.

Non-assignability

Read the Docs may assign or delegate these Terms of Service and/or our [Privacy Policy](#), in whole or in part, to any person or entity at any time with or without your consent, including the license grant in [License grant to us](#). You may not assign or delegate any rights or obligations under the Terms of Service or Privacy Policy without our prior written consent, and any unauthorized assignment and delegation by you is void.

Section headings and summaries

Throughout this Agreement, each section includes titles and brief summaries of the following terms and conditions. These section titles and brief summaries are not legally binding.

Severability, no waiver, and survival

If any part of this Agreement is held invalid or unenforceable, that portion of the Agreement will be construed to reflect the parties' original intent. The remaining portions will remain in full force and effect. Any failure on the part of Read the Docs to enforce any provision of this Agreement will not be considered a waiver of our right to enforce such provision. Our rights under this Agreement will survive any termination of this Agreement.

Amendments; complete agreement

This Agreement may only be modified by a written amendment signed by an authorized representative of Read the Docs, or by the posting by Read the Docs of a revised version in accordance with [Changes to these terms](#). These Terms of Service, together with our [Privacy Policy](#), represent the complete and exclusive statement of the agreement between you and us. This Agreement supersedes any proposal or prior agreement oral or written, and any other communications between you and Read the Docs relating to the subject matter of these terms including any confidentiality or nondisclosure agreements.

Questions

Questions about the Terms of Service? [Get in touch](#).

4.8 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.

4.8.1 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

4.8.2 Request Archive

Currently, Read the Docs has not received any takedown requests.

4.9 Policy for Abandoned Projects

This policy describes the process by which a Read the Docs project name may be changed.

4.9.1 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.

4.9.2 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

4.9.3 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.

4.9.4 Prior art

The Python Package Index (PyPI) policy for claiming abandoned packages ([PEP-0541](#)) heavily influenced this policy.

4.10 Changelog

4.10.1 Version 4.0.3

Date March 10, 2020

- [@stsewd](#): Document usage or pytest marks (#6764)
- [@stsewd](#): Update some dependencies (#6762)

- @stsewd: Refactor search view to make use of permission_classes (#6761)
- @ericholscher: Revert “Merge pull request #6739 from readthedocs/agj/docs-tos-pdf” (#6760)
- @ericholscher: Expand the logic in our proxito mixin. (#6759)
- @comradekingu: Spelling: “Set up your environment” (#6752)
- @humitos: Use storage.exists on HEAD method (#6751)
- @humitos: Pull only latest image for development (#6750)
- @humitos: Update common submodule (#6749)
- @ericholscher: Release 4.0.2 (#6741)
- @agjohnson: Add TOS PDF output (#6739)
- @ericholscher: Don’t call virtualenv with --no-site-packages (#6738)
- @GallowayJ: Drop mock dependency (#6723)
- @stsewd: Run proxito tests with proxito (#6714)
- @humitos: New block on footer template to override from corporate (#6702)
- @humitos: Point users to support email instead asking to open an issue (#6650)
- @stsewd: Proxy footer api on docs’ domains (#6630)

4.10.2 Version 4.0.2

Date March 04, 2020

- @ericholscher: Don’t call virtualenv with --no-site-packages (#6738)
- @stsewd: Catch ConnectionError from request on api timing out (#6735)
- @ericholscher: Release 4.0.1 (#6733)
- @humitos: Improve Proxito 404 handler to render user-facing Maze when needed (#6726)

4.10.3 Version 4.0.1

Date March 03, 2020

- @ericholscher: Add feature flag for branch & tag syncing to API. (#6729)
- @stsewd: Don’t fail a build on api timing out (#6719)
- @stsewd: Be explicit on privacy level for search tests (#6713)
- @stsewd: Make easy to run search tests in docker compose (#6711)
- @davidfischer: Docker settings improvements (#6709)
- @davidfischer: Workaround SameSite cookies (#6708)
- @davidfischer: Figure out the host IP when using Docker (#6707)
- @davidfischer: Pin the version of Azurite for docker-compose development (#6706)
- @ericholscher: Release 4.0.0 (#6704)
- @humitos: Rename docker settings to fix local environment (#6703)
- @sduthil: API v3 doc: fix typos in URL for PATCH /versions/slug/ (#6698)

- @humitos: Sort versions in-place to help performance (#6696)
- @humitos: Use `.iterator` when sorting versions (#6694)
- @agjohnson: Add feature flag to just completely skip sync and symlink operations (#6689)
- @humitos: Disable more loggings in development environment (#6683)
- @davidfischer: Use `x-forwarded-host` in local docker environment (#6679)
- @humitos: Allow user to set `build.image: testing` in the config file (#6676)
- @agjohnson: Add `azurite -loose` option (#6669)
- @stsewd: Have more control over search tests (#6644)
- @davidfischer: Enable content security policy in report-only mode (#6642)
- @stsewd: Add test settings file for proxito (#6623)
- @stsewd: Guide: using private submodules in `rtd.com` (#6527)

4.10.4 Version 4.0.0

Date February 25, 2020

This release upgrades our codebase to run on Django 2.2. This is a breaking change, so we have released it as our 4th major version.

- @stsewd: Data migration for old integration models (#6675)
- @ericholscher: Release 3.12.0 (#6674)
- @humitos: Upgrade to Django 2.2.9 (#6494)
- @davidfischer: Show message if version list truncated (#6276)

4.10.5 Version 3.12.0

Date February 18, 2020

This version has two major changes:

- It updates our default docker images to `stable=5.0` and `latest=6.0`.
- It changes our PR builder domain to `readthedocs.build`
- @humitos: Use `PUBLIC_DOMAIN_USES_HTTPS` for resolver tests (#6673)
- @stsewd: Always run `CoreTagsTests` with `http` (#6671)
- @ericholscher: Remove old docker settings (#6670)
- @stsewd: Update `gitpython` and `django` (#6667)
- @humitos: New docker release (6.0 and testing) (#6654)
- @humitos: Default python version per Docker image (#6653)
- @stsewd: Add `pytest-custom_exit_code` (#6648)
- @ericholscher: Initial attempt to serve PR builds at `readthedocs.build` (#6629)
- @ericholscher: Remove re-authing of users on downloads. (#6619)
- @stsewd: Don't trigger a sync twice on creation/deletion for GitHub (#6614)

- @s-weigand: Add linkcheck test for the docs (#6543)

4.10.6 Version 3.11.6

Date February 04, 2020

- @ericholscher: Note we aren't doing GSOC in 2020 (#6618)
- @ericholscher: only serve x-rtd-slug project if it exists (#6617)
- @ericholscher: Add check for a single_version project having a version_slug for PR builds (#6615)
- @stsewd: Fix linter (#6613)
- @stsewd: Create unique container per sync (#6612)
- @stsewd: Check for None before assignment (#6611)
- @ericholscher: Raise exception when we get an InfiniteRedirect (#6609)
- @ericholscher: Release 3.11.5 (#6608)
- @humitos: Avoid infinite redirect on El Proxito on 404 (#6606)
- @stsewd: Don't error when killing/removing non-existent container (#6605)
- @humitos: Use proper path to download/install readthedocs-ext (#6603)
- @humitos: Use timeout on internal API calls (#6602)
- @stsewd: Don't assume build isn't None in a docker build env (#6599)
- @ericholscher: Fix issue with pip 20.0 breaking on install (#6598)
- @stsewd: More protection against None (#6597)
- @agjohnson: Revert "Update celery requirements to its latest version" (#6596)
- @Blackcipher101: Changed documentation of Api v3 (#6574)
- @ericholscher: Use our standard auth mixin for proxito downloads (#6572)
- @humitos: Move common docker compose configs to common repository (#6539)

4.10.7 Version 3.11.5

Date January 29, 2020

- @humitos: Avoid infinite redirect on El Proxito on 404 (#6606)
- @humitos: Use proper path to download/install readthedocs-ext (#6603)
- @stsewd: Don't assume build isn't None in a docker build env (#6599)
- @ericholscher: Fix issue with pip 20.0 breaking on install (#6598)
- @agjohnson: Revert "Update celery requirements to its latest version" (#6596)
- @stsewd: Remove .cache from parent dir (#6595)
- @agjohnson: Release 3.11.4 again (#6594)
- @agjohnson: Release 3.11.4 (#6593)
- @ericholscher: Use our standard auth mixin for proxito downloads (#6572)
- @stsewd: Migrate doctype from project to version (#6523)

4.10.8 Version 3.11.4

Date January 28, 2020

- @humitos: Disable django debug toolbar in El Proximo (#6591)
- @stsewd: Respect docker setting on repo sync (#6589)
- @humitos: Merge pull request #6588 from readthedocs/humitos/support-ext (#6588)
- @humitos: Fix argument of update_repos (#6583)
- @humitos: Mount proper shared docker volume (#6581)
- @ericholscher: Use our standard auth mixin for proxito downloads (#6572)
- @stsewd: Delete .cache dir on wipe (#6571)
- @humitos: Run old redirect tests via El Proximo (#6570)
- @humitos: Remove 'build environment' from guides (#6568)
- @ericholscher: Fix /en/latest redirects (#6564)
- @stsewd: Merge pull request #6561 from stsewd/move-method (#6561)
- @stsewd: Use settings override in footer (#6560)
- @ericholscher: Fix proxito redirects breaking without a / (#6558)
- @stsewd: Remove unused file (#6557)
- @mgeier: DOC: Change a lot of http links to https (#6553)
- @stsewd: Don't use an instance of VCS when isn't needed (#6548)
- @saadm11: Add GitHub OAuth App Permission issue to PR Builder Troubleshooting docs (#6547)
- @humitos: Move common docker compose configs to common repository (#6539)
- @preetmishra: Update Transifex Integration details in Internationalization page. (#6531)
- @stsewd: Migrate doctype from project to version (#6523)
- @stsewd: Simplify docker image (#6519)
- @Parth1811: Fixes #5388 – Added Documentation for constraint while using Conda (#6509)
- @stsewd: Improve test for sync_repo (#6504)
- @humitos: Show debug toolbar when running docker compose (#6488)
- @dibyaaaaax: Add python examples for API v3 Documentation (#6487)

4.10.9 Version 3.11.3

Date January 21, 2020

- @ericholscher: Pass proper path to redirect code (#6555)
- @Daniel-Mietchen: Fixing a broken link (#6550)
- @stsewd: Guide: Intersphinx in Read the Docs (#6520)
- @humitos: Add netcat and telnet for celery debugging with rdb (#6518)
- @humitos: Core team development standards guide (#6517)

- @dibyaaaaax: Add www to the broken link (#6513)
- @davidfischer: Don't allow empty tags (#6512)
- @Parth1811: Fixes #6510 – Removed the `show_analytics` checks from the template (#6511)
- @stsewd: Only install node on eslint step on travis (#6505)
- @stsewd: Don't pass build to environment when doing a sync (#6503)
- @ericholscher: Release 3.11.2 (#6502)
- @Blackcipher101: Added “dirhtml” target (#6500)
- @humitos: Use `CELERY_APP_NAME` to call the proper celery app (#6499)
- @stsewd: Copy path from host only when using a LocalBuildEnvironment (#6482)
- @stsewd: Set env variables in the same way for DockerBuildEnvironment and Loc... (#6481)
- @stsewd: Use environment variable per run, not per container (#6480)
- @humitos: Update celery requirements to its latest version (#6448)
- @stsewd: Execute checkout step respecting docker setting (#6436)
- @humitos: Serve non-html at documentation domain though El Proxito (#6419)

4.10.10 Version 3.11.2

Date January 08, 2020

- @ericholscher: Fix link to my blog post breaking https (#6495)
- @humitos: Use a fixed IP for NGINX under docker-compose (#6491)
- @humitos: Add ‘index.html’ to the path before using `storage.url(path)` (#6476)
- @agjohnson: Release 3.11.1 (#6473)
- @humitos: Use tasks from common (including docker ones) (#6471)
- @humitos: Upgrade Django due a security issue (#6470)
- @humitos: Fix celery auto-reload command (#6469)
- @humitos: Use django storage to build URL returned by El Proxito (#6466)
- @ericholscher: Handle GitHub Push events with `deleted: true` in the JSON (#6465)
- @humitos: Serve external version through El Proxito (#6434)
- @segevfiner: Remove a stray backtick from `import-guide.rst` (#6362)

4.10.11 Version 3.11.1

Date December 18, 2019

- @humitos: Upgrade Django due a security issue (#6470)
- @humitos: Use django storage to build URL returned by El Proxito (#6466)
- @ericholscher: Handle GitHub Push events with `deleted: true` in the JSON (#6465)
- @ericholscher: Update troubleshooting steps for PR builder (#6463)
- @ericholscher: Add `DOCKER_NOELOAD` to compose settings (#6461)

- @stsewd: Be explicit when using setup_env (#6451)
- @keshavvinayak01: Fixed remove_search_analytics issue (#6447)
- @saadm11: Fix logic to build internal/external versions on update_repos management command (#6442)
- @humitos: Refactor get_downloads to make one query for default_version (#6441)
- @humitos: Do not expose env variables on external versions (#6440)
- @humitos: Better ES settings on docker-compose (#6439)
- @humitos: Remove global pip cache (#6437)
- @humitos: Bring Azure storage backend classes to this repository (#6433)
- @stsewd: Show predefined match on automation rules admin (#6432)
- @stsewd: Override production domain explicitly (#6431)
- @humitos: inv tasks to use when developing with docker (#6418)
- @piyushpalawat99: Fix #6395 (#6402)
- @stsewd: Only pass public versions to html context (#6118)
- @ericholscher: Add an “Edit Versions” listing to the Admin menu (#6110)
- @saadm11: Extend webhook notifications with build status (#5621)

4.10.12 Version 3.11.0

Date December 03, 2019

- @davidfischer: Use media availability instead of querying the filesystem (#6428)
- @stsewd: Remove beta note about sharing by password and header auth (#6426)
- @humitos: Use trigger_build for update_repos command (#6422)
- @humitos: Add more supported field to APIv3 docs (#6417)
- @humitos: Add AuthenticationMiddleware to El Proxito tests (#6416)
- @stsewd: Update docs on sharing (#6410)
- @humitos: Use WORKDIR to cd into a directory in Dockerfile (#6409)
- @humitos: Use /data inside Azurite container to persist data (#6407)
- @humitos: Serve non-html files from nginx (X-Accel-Redirect) (#6404)
- @humitos: Perform redirects at DB level (#6398)
- @humitos: Allow to extend El Proxito views from commercial (#6397)
- @humitos: Migrate El Proxito views to class-based views (#6396)
- @agjohnson: Fix CSS and how we were handling html in automation rule UI (#6394)
- @ericholscher: Release 3.10.0 (#6391)
- @stsewd: Set privacy level explicitly (#6390)
- @ericholscher: Redirect index files in proxito instead of serving (#6387)
- @humitos: Fully working docker-compose file (#6295)
- @saadm11: Refactor Subproject validation to use it for Forms and API (#6285)

- @saadmkl1: Refactor Gold Views (#6272)
- @stsewd: Add docs for automatin rules (#6072)

4.10.13 Version 3.10.0

Date November 19, 2019

- @stsewd: Set privacy level explicitly (#6390)
- @ericholscher: Redirect index files in proxito instead of serving (#6387)
- @stsewd: Fix search indexing (#6380)
- @humitos: Include creditcard.png image (#6379)
- @stsewd: Silent curl (#6377)
- @stsewd: Use github actions to trigger tests in corporate (#6376)
- @saadmkl1: Show only users projects in the APIv3 browseable form (#6374)
- @humitos: Release 3.9.0 (#6371)
- @davidfischer: Pin the node dependencies with a package-lock (#6370)
- @ericholscher: Small optimization to not compute the highest version when it isn't displayed (#6360)
- @krptic07: remove rss feed (#6348)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 44 (#6347)
- @ericholscher: Port additional features to proxito (#6286)
- @stsewd: Add docs for automatin rules (#6072)
- @stsewd: Implement UI for automation rules (#5996)

4.10.14 Version 3.9.0

Date November 12, 2019

- @davidfischer: Pin the node dependencies with a package-lock (#6370)
- @humitos: Force PUBLIC_DOMAIN_USES_HTTPS on version compare tests (#6367)
- @segevrfiner: Remove a stray backtick from import-guide.rst (#6362)
- @stsewd: Don't compare inactive or non build versions (#6361)
- @stsewd: Fix test (#6358)
- @ericholscher: Change the default of proxied_api_host to api_host (#6355)
- @stsewd: Dont link to dashboard from footer (#6353)
- @humitos: Upgrade django-storages (#6339)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 43 (#6334)
- @KartikKapil: added previous year gsoc projects (#6333)
- @stsewd: Support 6.0rc1 build image (#6329)
- @stsewd: Don't error on non existing version (#6325)
- @stsewd: Remove files from storage and delete indexes from ES when no longer needed (#6323)

- @stsewd: Fix eslint (#6317)
- @humitos: Revert “Adding RTD prefix for docker only in setting.py and all... (#6315)
- @anindyamanna: Fixed Broken links (#6300)
- @stsewd: Use sync instead of copy for blob storage (#6298)
- @sciencewhiz: Fix missing word in wipe guide (#6294)
- @jaferkhan: Removed unused code from view and template (#6250) (#6288)
- @stsewd: Rename test name (#6283)
- @davidfischer: Store version media availability (#6278)
- @davidfischer: Link to the terms of service (#6277)
- @saadm11: API V3 Subproject Creation Bug fix (#6275)
- @stsewd: Fix footer (#6274)
- @stsewd: Fix tests (#6269)
- @stsewd: Refactor profile’s views (#6267)
- @humitos: Default to None when using the Serializer as Form for Browsable... (#6266)
- @ericholscher: Fix inactive version list not showing when no results returned (#6264)
- @ericholscher: Downgrade django-storages. (#6263)
- @ericholscher: Release 3.8.0 (#6262)
- @stsewd: Update docs version detail (api v3) (#6259)
- @stsewd: Merge #6176 to master (#6258)
- @humitos: Remove privacy_level field from APIv3 (#6257)
- @saadm11: Redirect /projects/ URL to /dashboard/ (#6255)
- @davidfischer: Allow project badges for private version (#6252)
- @stsewd: Add pub_date to project admin (#6244)
- @saadm11: Allow only post requests for delete views (#6242)
- @Iamshankhadeep: Changing created to modified time (#6234)
- @ericholscher: Initial stub of proxito (#6226)
- @saadm11: Add Better error message for lists in config file (#6200)
- @stsewd: Put view under login (#6193)
- @humitos: Ship API v3 (#6169)
- @stsewd: Protection against ReDoS (#6163)
- @dojutsu-user: Optimize json parsing (#6160)
- @tapaswenipathak: Added missing i18n for footer api (#6144)
- @stsewd: Use different setting for footer api url (#6131)
- @dojutsu-user: Remove ‘highlight’ URL param from search results (#6087)
- @Iamshankhadeep: Adding RTD prefix for docker only in setting.py and all other places where is needed (#6040)

- @stsewd: Design doc for organizations (#5958)

4.10.15 Version 3.8.0

Date October 09, 2019

- @stsewd: Update doccs version detail (api v3) (#6259)
- @stsewd: Merge #6176 to master (#6258)
- @humitos: Remove privacy_level field from APIv3 (#6257)
- @saadmkl1: Redirect /projects/ URL to /dashboard/ (#6255)
- @davidfischer: Allow project badges for private version (#6252)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 40 (#6251)
- @saadmkl1: Add note about specifying dependencies (#6248)
- @stsewd: Add pub_date to project admin (#6244)
- @humitos: Do not use --cache-dir for pip if CLEAN_AFTER_BUILD is enabled (#6239)
- @stsewd: Update pytest (#6233)
- @iambenzo: remove /projects/ (#6228)
- @ericholscher: Initial stub of proxito (#6226)
- @davidfischer: Improve the version listview (#6224)
- @stsewd: Override production media artifacts on test (#6220)
- @davidfischer: Customize default build media storage for the FS (#6215)
- @agjohnson: Release 3.7.5 (#6214)
- @stsewd: Remove dead code (#6213)
- @stsewd: Only use the sphinx way to mock (#6212)
- @saadmkl1: Only Build Active Versions from Build List Page Form (#6205)
- @saadmkl1: Make raw_config private (#6199)
- @Iamshankhadeep: moved expandable_fields to meta class (#6198)
- @stsewd: Put view under login (#6193)
- @dojutsu-user: Remove pie-chart from search analytics page (#6192)
- @stsewd: Refactor SearchAnalytics view (#6190)
- @stsewd: Refactor ProjectRedirects views (#6187)
- @stsewd: Refactor ProjectTranslations views (#6185)
- @stsewd: Refactor ProjectNotifications views (#6183)
- @stsewd: Refactor views ProjectUsers (#6178)
- @humitos: Create subproject relationship via APIv3 endpoint (#6176)
- @stsewd: Refactor views ProjectVersion (#6175)
- @davidfischer: Add terms of service (#6174)
- @davidfischer: Document connected account permissions (#6172)

- @stsewd: Refactor views projects (#6171)
- @dojutsu-user: Optimize json parsing (#6160)
- @humitos: APIv3 endpoint: allow to modify a Project once it's imported (#5952)

4.10.16 Version 3.7.5

Date September 26, 2019

- @davidfischer: Remove if storage blocks (#6191)
- @davidfischer: Update security docs (#6179)
- @davidfischer: Add the private spamfighting module to INSTALLED_APPS (#6177)
- @davidfischer: Document connected account permissions (#6172)
- @stsewd: Require login for old redirect (#6170)
- @humitos: Remove old and unused code (#6167)
- @stsewd: Clean up views (#6166)
- @stsewd: Update docs for sharing (#6164)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 36 (#6158)
- @saadm11: Remove PR Builder Project Idea from RTD GSoC Docs (#6147)
- @ericholscher: Serialize time in search queries properly (#6142)
- @humitos: Allow to extend DomainCreate view (#6139)
- @saadm11: Integration Re-sync Bug Fix (#6124)
- @stsewd: Don't log BuildEnvironmentWarning as error (#6112)
- @dojutsu-user: Add Search Guide (#6101)
- @saadm11: Add PR Builder guide to docs (#6093)
- @dojutsu-user: Record search queries smartly (#6088)
- @dojutsu-user: Remove 'highlight' URL param from search results (#6087)

4.10.17 Version 3.7.4

Date September 05, 2019

- @ericholscher: Remove paid support callout (#6140)
- @ericholscher: Fix IntegrationAdmin with raw_id_fields for Projects (#6136)
- @ericholscher: Fix link to html_extra_path (#6135)
- @stsewd: Move out authorization from FooterHTML view (#6133)
- @agjohnson: Add setting for always cleaning the build post-build (#6132)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 35 (#6129)
- @stsewd: Refactor footer_html view to class (#6125)
- @ericholscher: Use raw_id_fields in the TokenAdmin (#6116)
- @davidfischer: Fixed footer ads supported on all themes (#6115)

- @stsewd: Don't log BuildEnvironmentWarning as error (#6112)
- @pllim: Use the force when fetching with Git (#6109)
- @dojutsu-user: Record search queries smartly (#6088)
- @stsewd: Add move method to automation rule (#5998)
- @dojutsu-user: Index more domain data into elasticsearch (#5979)

4.10.18 Version 3.7.3

Date August 27, 2019

- @pllim: Use the force when fetching with Git (#6109)
- @davidfischer: Small improvements to the SEO guide (#6105)
- @davidfischer: Update intersphinx mapping with canonical sources (#6085)
- @davidfischer: Fix lingering 500 issues (#6079)
- @davidfischer: Technical docs SEO guide (#6077)
- @saadmkl1: GitLab Build Status Reporting for PR Builder (#6076)
- @davidfischer: Update ad details docs (#6074)
- @davidfischer: Gold makes projects ad-free again (#6073)
- @saadmkl1: Auto Sync and Re-Sync for Manually Created Integrations (#6071)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 32 (#6067)
- @saadmkl1: PR Builder GitLab Integration (#6066)
- @davidfischer: Send media downloads to analytics (#6063)
- @davidfischer: IPv6 in X-Forwarded-For fix (#6062)
- @humitos: Remove warning about beta state of conda support (#6056)
- @saadmkl1: Update GitLab Webhook creating to enable merge request events (#6055)
- @ericholscher: Release 3.7.2 (#6054)
- @dojutsu-user: Update feature flags docs (#6053)
- @saadmkl1: Add index.html filename to the external doc url (#6051)
- @dojutsu-user: Search analytics improvements (#6050)
- @stsewd: Sort versions taking into consideration the vcs type (#6049)
- @humitos: Avoid returning invalid domain when using USE_SUBDOMAIN=True in dev (#6026)
- @dojutsu-user: Search analytics (#6019)
- @tapaswenipathak: Remove django-guardian model (#6005)
- @stsewd: Add manager and description field to AutomationRule model (#5995)
- @davidfischer: Cleanup project tags (#5983)
- @davidfischer: Search indexing with storage (#5854)
- @wilvk: fix sphinx startup guide to not to fail on rtd build as per #2569 (#5753)

4.10.19 Version 3.7.2

Date August 08, 2019

- @dojutsu-user: Update feature flags docs (#6053)
- @saadmkl1: Add index.html filename to the external doc url (#6051)
- @dojutsu-user: Search analytics improvements (#6050)
- @stsewd: Sort versions taking into consideration the vcs type (#6049)
- @ericholscher: When called via SyncRepositoryTaskStep this doesn't exist (#6048)
- @davidfischer: Fix around community ads with an explicit ad placement (#6047)
- @ericholscher: Release 3.7.1 (#6045)
- @saadmkl1: Do not delete media storage files for external version (#6035)
- @tapaswenipathak: Remove django-guardian model (#6005)
- @davidfischer: Cleanup project tags (#5983)
- @davidfischer: Search indexing with storage (#5854)

4.10.20 Version 3.7.1

Date August 07, 2019

- @pyup-bot: pyup: Scheduled weekly dependency update for week 31 (#6042)
- @agjohnson: Fix issue with save on translation form (#6037)
- @saadmkl1: Do not delete media storage files for external version (#6035)
- @saadmkl1: Do not show wipe version message on build details page for External versions (#6034)
- @saadmkl1: Send site notification on Build status reporting failure and follow DRY (#6033)
- @davidfischer: Use Read the Docs for Business everywhere (#6029)
- @davidfischer: Remove project count on homepage (#6028)
- @stsewd: Fix missing arg in tests (#6022)
- @ericholscher: Update get_absolute_url for External Versions (#6020)
- @dojutsu-user: Search analytics (#6019)
- @saadmkl1: Fix issues around remote repository for sending Build status reports (#6017)
- @ericholscher: Expand the scope between before_vcs and after_vcs (#6015)
- @davidfischer: Handle .x in version sorting (#6012)
- @tapaswenipathak: Update note (#6008)
- @davidfischer: Link to Read the Docs for Business docs from relevant sections (#6004)
- @davidfischer: Note RTD for Biz requires SSL for custom domains (#6003)
- @davidfischer: Allow searching in the Django Admin for gold (#6001)
- @saadmkl1: More explicit tests for build managers (#6000)
- @dojutsu-user: Fix logic involving creation of Sphinx Domains (#5997)

- @dojutsu-user: Fix: no highlighting of matched keywords in search results (#5994)
- @saadmk11: Do not copy external version artifacts twice (#5992)
- @saadmk11: Update GitHub build status details URL (#5987)
- @humitos: Missing list.extend line when appending conda dependencies (#5986)
- @saadmk11: Fix github build status reporting bug (#5985)
- @dojutsu-user: Use try...catch block with underscore.js template. (#5984)
- @davidfischer: Cleanup project tags (#5983)
- @ericholscher: Release 3.7.0 (#5982)
- @stsewd: More explicit tests for version managers (#5981)
- @dojutsu-user: Search Fix: section_subtitle_link is not defined (#5980)
- @stsewd: More explicit setup for tests (#5977)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 29 (#5975)
- @stsewd: Update gitpython (#5974)
- @davidfischer: Community only ads for more themes (#5973)
- @darrowco: updated to pycopg2 (2.8.3) (#5965)
- @humitos: Append core requirements to Conda environment file (#5956)
- @humitos: Show APIv3 Token under Profile settings (#5954)
- @stsewd: Remove skip submodules flag (#5406)

4.10.21 Version 3.7.0

Date July 23, 2019

- @dojutsu-user: Search Fix: section_subtitle_link is not defined (#5980)
- @stsewd: More explicit setup for tests (#5977)
- @stsewd: Update gitpython (#5974)
- @davidfischer: Community only ads for more themes (#5973)
- @kittenking: Fix typos across readthedocs.org repository (#5971)
- @dojutsu-user: Fix: parse_json also including html in titles (#5970)
- @saadmk11: update external version check for notification task (#5969)
- @pranay414: Improve error message for invalid submodule URLs (#5957)
- @humitos: Append core requirements to Conda environment file (#5956)
- @Abhi-khandelwal: Exclude Spam projects count from total_projects count (#5955)
- @humitos: Show APIv3 Token under Profile settings (#5954)
- @ericholscher: Release 3.6.1 (#5953)
- @ericholscher: Missed a couple places to set READTHEDOCS_LANGUAGE (#5951)
- @dojutsu-user: Hotfix: Return empty dict when no highlight dict is present (#5950)
- @humitos: Use a cwd where the user has access inside the container (#5949)

- @saadmkl1: Small Changes to PR Builder Code (#5948)
- @saadmkl1: update build status message for github (#5947)
- @ericholscher: Integrate indoc search into our prod docs (#5946)
- @ericholscher: Explicitly delete SphinxDomain objects from previous versions (#5945)
- @ericholscher: Properly return None when there's no highlight on a hit. (#5944)
- @ericholscher: Add READTHEDOCS_LANGUAGE to the environment during builds (#5941)
- @ericholscher: Merge the GSOC 2019 in-doc search changes (#5919)
- @saadmkl1: Add check for external version in conf.py.tmpl for warning banner (#5900)
- @Abhi-khandelwal: Point users to commercial solution for their private repositories (#5849)
- @ericholscher: Merge initial work from Pull Request Builder GSOC (#5823)

4.10.22 Version 3.6.1

Date July 17, 2019

- @ericholscher: Missed a couple places to set READTHEDOCS_LANGUAGE (#5951)
- @dojutsu-user: Hotfix: Return empty dict when no highlight dict is present (#5950)
- @humitos: Use a cwd where the user has access inside the container (#5949)
- @saadmkl1: Small Changes to PR Builder Code (#5948)
- @ericholscher: Explicitly delete SphinxDomain objects from previous versions (#5945)
- @ericholscher: Properly return None when there's no highlight on a hit. (#5944)
- @ericholscher: Release 3.6.0 (#5943)
- @ericholscher: Bump the Sphinx extension to 1.0 (#5942)
- @ericholscher: Add READTHEDOCS_LANGUAGE to the environment during builds (#5941)
- @dojutsu-user: Small search doc fix (#5940)
- @dojutsu-user: Indexing speedup (#5939)
- @dojutsu-user: Small improvement in parse_json (#5938)
- @dojutsu-user: Use attrgetter in sorted function (#5936)
- @saadmkl1: Refine PR Builder Code (#5933)
- @dojutsu-user: Fix spacing between the results and add highlight url param (#5932)
- @ericholscher: Merge the GSOC 2019 in-doc search changes (#5919)
- @dojutsu-user: Add tests for section-linking (#5918)
- @saadmkl1: Update build list and detail page UX (#5916)
- @humitos: APIv3 endpoint to manage Environment Variables (#5913)
- @humitos: Split APIv3 tests on different files (#5911)
- @stsewd: Better msg when gitpython fails (#5903)
- @saadmkl1: Add check for external version in conf.py.tmpl for warning banner (#5900)
- @humitos: Update APIv3 documentation with latest changes (#5895)

4.10.23 Version 3.6.0

Date July 16, 2019

- @ericholscher: Bump the Sphinx extension to 1.0 (#5942)
- @ericholscher: Add READTHEDOCS_LANGUAGE to the environment during builds (#5941)
- @dojutsu-user: Small search doc fix (#5940)
- @dojutsu-user: Indexing speedup (#5939)
- @dojutsu-user: Small improvement in parse_json (#5938)
- @dojutsu-user: Use attrgetter in sorted function (#5936)
- @saadmkl1: Refine PR Builder Code (#5933)
- @dojutsu-user: Fix spacing between the results and add highlight url param (#5932)
- @Abhi-khandelwal: remove the usage of six (#5930)
- @dojutsu-user: Fix count value of docsearch REST api (#5926)
- @ericholscher: Merge the GSOC 2019 in-doc search changes (#5919)
- @dojutsu-user: Add tests for section-linking (#5918)
- @saadmkl1: Update build list and detail page UX (#5916)
- @humitos: These Project's methods are not used (#5915)
- @saadmkl1: Github Status reporting Test fix (#5914)
- @humitos: APIv3 endpoint to manage Environment Variables (#5913)
- @humitos: Split APIv3 tests on different files (#5911)
- @saadmkl1: Add Feature Flag to Enable External Version Building (#5910)
- @ericholscher: Pass the build_pk to the task instead of the build object itself (#5904)
- @stsewd: Better msg when gitpython fails (#5903)
- @saadmkl1: Exclude external versions from get_latest_build (#5901)
- @humitos: Update conda at startup (#5897)
- @humitos: Update APIv3 documentation with latest changes (#5895)
- @stsewd: Add tests for version and project querysets (#5894)
- @davidfischer: Rework on documentation guides (#5893)
- @humitos: Lint (pep257: D415) (#5892)
- @davidfischer: Fix spaces in email subject link (#5891)
- @saadmkl1: Build only HTML and Save external version artifacts in different directory (#5886)
- @humitos: APIv3 CRUD for Redirect objects (#5879)
- @ericholscher: Add config to Build and Version admin (#5877)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 26 (#5874)
- @stsewd: Call distinct to the end of the querysets (#5872)
- @pranay414: Change rtd to readthedocs (#5871)
- @humitos: APIv3 refactor some fields (#5868)

- @saadmkl1: Send Build Status Report Using GitHub Status API (#5865)
- @humitos: APIv3 “Import Project” endpoint (#5857)
- @stsewd: Remove django guardian from querysets (#5853)
- @humitos: Hide “Protected” privacy level from users (#5833)
- @dojutsu-user: Add section linking for the search result (#5829)

4.10.24 Version 3.5.3

Date June 19, 2019

- @davidfischer: Treat docs warnings as errors (#5825)
- @davidfischer: Fix some unclear verbiage (#5820)
- @davidfischer: Rework documentation index page (#5819)
- @davidfischer: Upgrade intersphinx to Django 1.11 (#5818)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 24 (#5817)
- @humitos: Disable changing domain when editing the object (#5816)
- @saadmkl1: Update docs with sitemap sort order change (#5815)
- @davidfischer: Optimize requests to APIv3 (#5803)
- @ericholscher: Show build length in the admin (#5802)
- @stsewd: Move search functions (#5801)
- @ericholscher: A few small improvements to help with search admin stuff (#5800)
- @stsewd: Simplify es indexing (#5798)
- @humitos: Use a real SessionBase object on FooterNoSessionMiddleware (#5797)
- @stsewd: Add logging in magic methods (#5795)
- @stsewd: Fix unbound var in search view (#5794)
- @davidfischer: Mention security issue in the changelog (#5790)
- @stsewd: Index path with original path name (#5785)
- @stsewd: Use querysets from the class not from an instance (#5783)
- @saadmkl1: Add Build managers and Update Build Querysets. (#5779)
- @davidfischer: Project advertising page/form update (#5777)
- @davidfischer: Update docs around opt-out of ads (#5776)
- @saadmkl1: Sitemap sort order priorities updated (#5724)
- @dojutsu-user: [Design Doc] In Doc Search UI (#5707)
- @saadmkl1: Pull Request Builder Design Doc (#5705)
- @humitos: Support single version subprojects URLs to serve from Django (#5690)
- @agjohnson: Add a contrib Dockerfile for local build image on Linux (#4608)

4.10.25 Version 3.5.2

This is a quick hotfix to the previous version.

Date June 11, 2019

- @ericholscher: Fix version of our sphinx-ext we're installing (#5789)
- @stsewd: Get version from the api (#5788)

4.10.26 Version 3.5.1

This version contained a [security fix](#) for an open redirect issue. The problem has been fixed and deployed on readthedocs.org. For users who depend on the Read the Docs code line for a private instance of Read the Docs, you are encouraged to update to 3.5.1 as soon as possible.

Date June 11, 2019

- @stsewd: Update build images in docs (#5782)
- @saadmkl1: Validate dict when parsing the mkdocs.yml file (#5775)
- @stsewd: Pin textclassifier dependencies (#5773)
- @stsewd: Fix tests on master (#5769)
- @stsewd: Don't use implicit relative import (#5767)
- @stsewd: Use version_pk to trigger builds (#5765)
- @davidfischer: Domain UI improvements (#5764)
- @ericholscher: Try to fix Elastic connection pooling issues (#5763)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 22 (#5762)
- @ericholscher: Try to fix Elastic connection pooling issues (#5760)
- @davidfischer: Escape variables in mkdocs data (#5759)
- @humitos: Serve 404/index.html file for htmldir Sphinx builder (#5754)
- @wilvk: fix sphinx startup guide to not to fail on rtd build as per #2569 (#5753)
- @stsewd: Fix mkdocs relpath (#5749)
- @stsewd: Call lock per task (#5748)
- @stsewd: Pin kombu to 4.3.0 (#5747)
- @agjohnson: Clarify latexmk option usage (#5745)
- @ericholscher: Hotfix latexmk builder to ignore error codes (#5744)
- @ericholscher: Hide the Code API search in the UX for now. (#5743)
- @davidfischer: Add init.py under readthedocs/api (#5742)
- @dojutsu-user: Fix design docs missing from toctree (#5741)
- @ericholscher: Release 3.5.0 (#5740)
- @saadmkl1: Pytest Timezone Warning Fixed (#5739)
- @humitos: Filter by projects with no banned users (#5733)
- @davidfischer: Fix the sidebar ad color (#5731)

- @saadmkl1: Permanent redirect feature added (#5727)
- @humitos: Move version “Clean” button to details page (#5706)
- @gorshunovr: Update flags documentation (#5701)
- @davidfischer: Storage updates (#5698)
- @stsewd: Remove files after build (#5680)
- @stsewd: Move community support to email (#5651)
- @davidfischer: Optimizations and UX improvements to the dashboard screen (#5637)
- @chrisjsewell: Use `--upgrade` instead of `--force-reinstall` for pip installs (#5635)
- @stsewd: Move file validations out of the config module (#5627)
- @humitos: Remove old/deprecated build endpoints (#5479)
- @shivanshu1234: Add link to in-progress build from dashboard. (#5431)
- @stsewd: Downgrade pytest-django (#5294)

4.10.27 Version 3.5.0

Date May 30, 2019

- @pyup-bot: pyup: Scheduled weekly dependency update for week 21 (#5737)
- @humitos: Update feature flags exposed to user in docs (#5734)
- @davidfischer: Fix the sidebar ad color (#5731)
- @davidfischer: Create a funding file (#5729)
- @davidfischer: Small commercial hosting page rework (#5728)
- @mattparrilla: Add note about lack of support for private repos (#5726)
- @humitos: Canonical consistency example (#5722)
- @humitos: Use nonstopmode for latexmk (#5714)
- @cclauss: Identity is not the same thing as equality in Python (#5713)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 20 (#5712)
- @humitos: Move version “Clean” button to details page (#5706)
- @ericholscher: Explicitly mention a support email (#5703)
- @davidfischer: Storage updates (#5698)
- @humitos: Enable auth validate passwords (#5696)
- @stsewd: Simplify lock acquire (#5695)
- @stsewd: Simplify update docs task (#5694)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 19 (#5692)
- @saadmkl1: Warning about using sqlite 3.26.0 for development (#5681)
- @davidfischer: Configure the security middleware (#5679)
- @stsewd: Fix bug in notifications (#5678)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 18 (#5667)

- @saadmkl1: pylint fix for notifications, restapi and config (#5664)
- @saadmkl1: pylint fix for readthedocs.search (#5663)
- @saadmkl1: pylint fix for readthedocs.projects (#5662)
- @saadmkl1: pylint fix for readthedocs.doc_builder (#5660)
- @humitos: Support Docker 5.0 image (#5657)
- @humitos: Use latexmk if Sphinx > 1.6 (#5656)
- @humitos: Upgrade docker python package to latest release (#5654)
- @saadmkl1: pylint fix for readthedocs.core (#5650)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 17 (#5645)
- @saadmkl1: Serve custom 404 pages from Django (#5644)
- @yarons: Typo fix (#5642)
- @saadmkl1: Sitemap hreflang syntax invalid for regional language variants fix (#5638)
- @davidfischer: Optimizations and UX improvements to the dashboard screen (#5637)
- @davidfischer: Redirect project slugs with underscores (#5634)
- @saadmkl1: Standardizing the use of settings directly (#5632)
- @saadmkl1: Note for Docker image size in Docker instructions (#5630)
- @davidfischer: UX improvements around SSL certificates (#5629)
- @davidfischer: Gold project sponsorship changes (#5628)
- @davidfischer: Make sure there's a contact when opting out of advertising (#5626)
- @stsewd: Remove unused volume from docker (#5625)
- @dojutsu-user: hotfix: correct way of getting environment variables (#5622)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 16 (#5619)
- @ericholscher: Release 3.4.2 (#5613)
- @ericholscher: Add explicit egg version to unicode-slugify (#5612)
- @dojutsu-user: Remove ProxyMiddleware (#5607)
- @dojutsu-user: Remove 'Versions' tab from Admin Dashboard. (#5600)
- @dojutsu-user: Notify the user when deleting a superproject (#5596)
- @saadmkl1: Handle 401, 403 and 404 when setting up webhooks (#5589)
- @saadmkl1: Unify usage of settings and remove the usage of getattr for settings (#5588)
- @saadmkl1: Note about admin page in the docs (#5585)
- @humitos: Remove USE_SETUPTOOLS_LATEST feature flag (#5578)
- @saadmkl1: Validate docs dir before writing custom js (#5569)
- @rshrc: Added note in YAML docs (#5565)
- @shivanshu1234: Specify python3 in installation instructions. (#5552)
- @davidfischer: Write build artifacts to (cloud) storage from build servers (#5549)
- @saadmkl1: "Default branch: latest" does not exist Fix. (#5547)

- @dojutsu-user: Update `readthedocs-environment.json` file when env vars are added/deleted (#5540)
- @humitos: Update common to its latest version (#5517)
- @saadmkl1: Profile page performance issue Fix (#5472)
- @stsewd: Remove unused form (#5443)
- @stsewd: Use relative paths in config module (#5377)
- @humitos: Initial structure for APIv3 (#5356)
- @stsewd: Add models for automation rules (#5323)
- @stsewd: Downgrade pytest-django (#5294)
- @ericholscher: Add search for DomainData objects (#5290)
- @gorshunovr: Change version references to :latest tag (#5245)
- @dojutsu-user: Fix buttons problems in 'Change Email' section. (#5219)

4.10.28 Version 3.4.2

Date April 22, 2019

- @ericholscher: Add explicit egg version to unicode-slugify (#5612)
- @saadmkl1: Update Environmental Variable character limit (#5597)
- @davidfischer: Add meta descriptions to top documentation (#5593)
- @stsewd: Ignore pytest-xdist from pyupdate (#5590)
- @saadmkl1: Note about admin page in the docs (#5585)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 14 (#5580)
- @stsewd: Use downloads.html in template (#5579)
- @ihnorton: Fix typo in conda.rst (#5576)
- @davidfischer: Fix for Firefox to close the ad correctly (#5571)
- @davidfischer: Non mobile fixed footer ads (#5567)
- @ericholscher: Release 3.4.1 (#5566)
- @dojutsu-user: Update `readthedocs-environment.json` file when env vars are added/deleted (#5540)
- @stsewd: Allow build mkdocs outside root (#5539)
- @saadmkl1: Sitemap assumes that all versions are translated Fix. (#5535)
- @saadmkl1: Remove Header Login button from login page (#5534)
- @davidfischer: Optimize database performance of the footer API (#5530)
- @stsewd: Don't depend of enabled pdf/epub to show downloads (#5502)
- @saadmkl1: Don't allow to create subprojects with same alias (#5404)
- @saadmkl1: Improve project translation listing Design under admin tab (#5380)

4.10.29 Version 3.4.1

Date April 03, 2019

- @pyup-bot: pyup: Scheduled weekly dependency update for week 13 (#5558)
- @stsewd: Fix advanced settings form (#5544)
- @stsewd: Call mkdocs using -m (#5542)
- @stsewd: Allow build mkdocs outside root (#5539)
- @stsewd: Use patch method to update has_valid_clone (#5538)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 12 (#5536)
- @saadm11: Sitemap assumes that all versions are translated Fix. (#5535)
- @saadm11: Remove Header Login button from login page (#5534)
- @stevepiercy: Add pylons-sphinx-themes to list of supported themes (#5533)
- @davidfischer: Optimize database performance of the footer API (#5530)
- @stsewd: Fix extra origin in urls (#5523)
- @davidjb: Update contributing docs for RTD's own docs (#5522)
- @davidjb: Use HTTPS for intersphinx mappings (#5521)
- @davidjb: Fix formatting for CentOS/RHEL installs (#5520)
- @davidfischer: Guide users to the YAML config from the build detail page (#5519)
- @davidjb: Add to and reorder GitHub webhook docs (#5514)
- @stsewd: Link to the docdir of the remote repo in non-rtd themes for mkdocs (#5513)
- @stevepiercy: Tidy up grammar, promote Unicode characters (#5511)
- @stsewd: Catch specific exception for config not found (#5510)
- @dojutsu-user: Use ValueError instead of InvalidParamsException (#5509)
- @humitos: Force Sphinx to not use xindy (#5507)
- @stsewd: Update mkdocs (#5505)
- @stsewd: Don't depend of enabled pdf/epub to show downloads (#5502)
- @ericholscher: Remove search & API from robots.txt (#5501)
- @saadm11: Make /random/ path work (#5496)
- @humitos: Typo on conf.py.tmpl (#5495)
- @rshrc: Added note warning about using sqlite 3.26.0 in development (#5491)
- @stsewd: Regroup advanced settings (#5489)
- @ericholscher: Fix bug that caused search objects not to delete (#5487)
- @ericholscher: Release 3.4.0 (#5486)
- @davidfischer: Promote the YAML config (#5485)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 11 (#5483)
- @saadm11: Dashboard screen performance fix (#5471)
- @saadm11: Build List Screen Performance Issue Fix. (#5470)

- @saadmk11: Remove Haystack reference in Docs (#5469)
- @davidfischer: Enable Django Debug Toolbar in development (#5464)
- @davidfischer: Optimize the version list screen (#5460)
- @stsewd: Regroup settings (#5459)
- @humitos: Guide to build PDF for non-ASCII language (#5453)
- @dojutsu-user: Remove asserts from code. (#5452)
- @davidfischer: Optimize the repos API query (#5451)
- @stsewd: Update version of setuptools (#5450)
- @stsewd: Remove unused validator (#5442)
- @humitos: Build PDF files using latexmk (#5437)
- @stsewd: Always update the commit of the stable version (#5421)
- @stsewd: Share doctree between builders (#5407)
- @stsewd: Remove unused template (#5401)
- @orlnub123: Fix pip installs (#5386)
- @davidfischer: Add an application form for community ads (#5379)

4.10.30 Version 3.4.0

Date March 18, 2019

- @davidfischer: Promote the YAML config (#5485)
- @saadmk11: Dashboard screen performance fix (#5471)
- @saadmk11: Build List Screen Performance Issue Fix. (#5470)
- @saadmk11: Remove Haystack reference in Docs (#5469)
- @mashrikt: gitignore dev.db-journal file #5463 (#5466)
- @davidfischer: Enable Django Debug Toolbar in development (#5464)
- @davidfischer: Optimize the version list screen (#5460)
- @stsewd: Regroup settings (#5459)
- @Mariatta: Fix typo: leave the field black -> blank (#5457)
- @stsewd: Use Ubuntu xenial on travis (#5456)
- @dojutsu-user: Update links to point to stable version. (#5455)
- @dojutsu-user: Fix inconsistency in footer links (#5454)
- @davidfischer: Optimize the repos API query (#5451)
- @stsewd: Update version of setuptools (#5450)
- @stsewd: Remove unused validator (#5442)
- @humitos: Build PDF files using latexmk (#5437)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 10 (#5432)
- @shivanshu1234: Remove invalid example from v2.rst (#5430)

- @saadmkl1: Removed unused constant from core.models (#5424)
- @stsewd: Fix reraise of exception (#5423)
- @stsewd: Always update the commit of the stable version (#5421)
- @stsewd: Fix warnings in code (#5419)
- @stsewd: Refactor move_files (#5418)
- @agarwalrounak: Document that people can create a version named stable (#5417)
- @agarwalrounak: Update installation guide to include submodules (#5416)
- @stsewd: Update docs for building with markdown (#5415)
- @stsewd: Share doctree between builders (#5407)
- @humitos: Communicate the project slug can be changed by requesting it (#5403)
- @stsewd: Remove unused template (#5401)
- @stsewd: Remove view docs dropdown (#5400)
- @humitos: Minimum upgrade of the builds docs (#5398)
- @stsewd: Update internal requirements (#5396)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 09 (#5395)
- @dojutsu-user: Trigger build on default branch when saving a project (#5393)
- @Mike-Dai: Removed un-needed python dependencies (#5389)
- @orlnub123: Fix pip installs (#5386)
- @rshrc: Addressed Issue #5327 (#5383)
- @ericholscher: Be extra explicit about the CNAME (#5382)
- @stsewd: Better MkDocs integration as GSoC idea (#5378)
- @ericholscher: Release 3.3.1 (#5376)
- @ericholscher: Add a GSOC section for openAPI (#5375)
- @dojutsu-user: Make 'default_version' field as readonly if no active versions are found. (#5374)
- @ericholscher: Be more defensive with our storage uploading (#5371)
- @ericholscher: Check for two paths for each file (#5370)
- @ericholscher: Don't show projects in Sphinx Domain Admin sidebar (#5367)
- @stsewd: Start building with sphinx 1.8 (#5366)
- @saadmkl1: Remove pytest warnings (#5346)
- @davidfischer: Remove the v1 API (#5293)
- @stsewd: Remove doctype from resolver (#5230)
- @humitos: Implementation of APIv3 (#4863)

4.10.31 Version 3.3.1

Date February 28, 2019

- @ericholscher: Be more defensive with our storage uploading (#5371)
- @ericholscher: Check for two paths for each file (#5370)
- @stsewd: Protect against anchors with # (#5369)
- @ericholscher: Don't show projects in Sphinx Domain Admin sidebar (#5367)
- @ericholscher: Fix sphinx domain models and migrations (#5363)
- @stsewd: Try to put back codecov integration (#5362)
- @ericholscher: Release 3.3.0 (#5361)
- @ericholscher: Fix search bug when an empty list of objects_id was passed (#5357)
- @dojutsu-user: Add admin methods for reindexing versions from project and version admin. (#5343)
- @stsewd: Cleanup a little of documentation_type from footer (#5315)
- @ericholscher: Add modeling for intersphinx data (#5289)
- @stsewd: Remove doctype from resolver (#5230)
- @stsewd: Validate webhook's payload (#4940)
- @stsewd: Start testing config v2 on our project (#4838)
- @ericholscher: Revert "Merge pull request #4636 from readthedocs/search_upgrade" (#4716)
- @safwanrahman: [GSoC 2018] All Search Improvements (#4636)
- @stsewd: Add schema for configuration file with yamale (#4084)
- @stsewd: Add note about mercurial on tests (#3358)

4.10.32 Version 3.3.0

Date February 27, 2019

- @ericholscher: Fix search bug when an empty list of objects_id was passed (#5357)
- @agjohnson: Update UI translations (#5354)
- @ericholscher: Update GSOC page to mention we're accepted. (#5353)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 08 (#5352)
- @dojutsu-user: Increase path's max_length for ImportedFile model to 4096 (#5345)
- @saadmkl1: improvement on inserting mkdocs media (#5344)
- @dojutsu-user: Add admin methods for reindexing versions from project and version admin. (#5343)
- @stsewd: Initialize local variable before using it (#5342)
- @dojutsu-user: Remove deprecated code (#5341)
- @stsewd: Require conda.file when using conda in v1 (#5338)
- @stsewd: Remove unused setting (#5336)
- @stsewd: Fix comment (#5329)

- @stsewd: Don't depend on specific data when catching exception (#5326)
- @regisb: Fix "clean_builds" command argument parsing (#5320)
- @stsewd: Cleanup a little of documentation_type from footer (#5315)
- @humitos: Warning note about running ES locally for tests (#5314)
- @humitos: Update documentation on running test for python environment (#5313)
- @ericholscher: Release 3.2.3 (#5312)
- @ericholscher: Add basic auth to the generic webhook API. (#5311)
- @ericholscher: Fix an issue where we were not properly filtering projects (#5309)
- @stsewd: Rstrip repo url (#5308)
- @rexzing: Incompatible dependency for prospector with pylint-django (#5306)
- @davidfischer: Allow extensions to control URL structure (#5296)
- @stsewd: Downgrade pytest-django (#5294)
- @ericholscher: Add modeling for intersphinx data (#5289)
- @ovc: Tweek css for sphinx_prompt (#5281)
- @saadm11: #4036 Updated build list to include an alert state (#5222)
- @humitos: Use unicode-slugify to generate Version.slug (#5186)
- @dojutsu-user: Add admin functions for wiping a version (#5140)
- @humitos: Generate general sitemap.xml for projects (#5122)
- @humitos: Logging exceptions rework (#5118)
- @davidfischer: Store ePubs and PDFs in media storage (#4947)
- @stsewd: Validate webhook's payload (#4940)
- @ericholscher: Revert "Merge pull request #4636 from readthedocs/search_upgrade" (#4716)
- @safwanrahman: [GSoC 2018] All Search Improvements (#4636)

4.10.33 Version 3.2.3

Date February 19, 2019

- @ericholscher: Add basic auth to the generic webhook API. (#5311)
- @ericholscher: Fix an issue where we were not properly filtering projects (#5309)
- @stsewd: Rstrip repo url (#5308)
- @stsewd: Use autosectionlabel for docs in security (#5307)
- @rexzing: Incompatible dependency for prospector with pylint-django (#5306)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 07 (#5305)
- @davidfischer: Allow extensions to control URL structure (#5296)
- @stsewd: Downgrade pytest-django (#5294)
- @rexzing: Docs reformatting with :guilabel: (#5161)

4.10.34 Version 3.2.2

Date February 13, 2019

- @ericholscher: Support old jquery where responseJSON doesn't exist (#5285)
- @humitos: pyup.yml syntax fixed (#5284)
- @dojutsu-user: Fix error of travis (rename migration file) (#5282)
- @humitos: pyup YAML configuration file (#5279)
- @pyup-bot: Pin ipdb to latest version 0.11 (#5278)
- @pyup-bot: Pin datadiff to latest version 2.0.0 (#5277)
- @pyup-bot: Pin pytest-cov to latest version 2.6.1 (#5276)
- @pyup-bot: Pin pillow to latest version 5.4.1 (#5275)
- @pyup-bot: Update elasticsearch to 6.3.1 (#5274)
- @discover: clarify github integration needs `https://` prepended (#5273)
- @humitos: Setup and configure pyup.io (#5272)
- @humitos: Update all Python dependencies (#5269)
- @davidfischer: Add note about security issue (#5263)
- @ericholscher: Don't delay search delete on project delete (#5262)
- @agjohnson: Automate docs version from our setup.cfg (#5259)
- @agjohnson: Add admin actions for building versions (#5255)
- @ericholscher: Give the 404 page a title. (#5252)
- @humitos: Make our SUFFIX default selection py2/3 compatible (#5251)
- @ericholscher: Release 3.2.1 (#5248)
- @ericholscher: Remove excluding files on search. (#5246)
- @gorshunovr: Change version references to :latest tag (#5245)
- @humitos: Remove py2 compatibility (#5241)
- @stsewd: Allow to override trigger_build from demo project (#5236)
- @ericholscher: Change some info logging to debug to clean up build output (#5233)
- @stsewd: Fake auth middleware in tests (#5206)
- @EJEP: Clarify 'more info' link in admin settings page (#5180)
- @rexzing: Docs reformatting with :guilabel: (#5161)

4.10.35 Version 3.2.1

Date February 07, 2019

- @ericholscher: Remove excluding files on search. (#5246)
- @ericholscher: Don't update search on HTMLFile save (#5244)
- @ericholscher: Be more defensive in our 404 handler (#5243)

- @humitos: Install sphinx-notfound-page for building 404.html custom page (#5242)
- @humitos: Remove py2 compatibility (#5241)
- @ericholscher: Release 3.2.0 (#5240)

4.10.36 Version 3.2.0

Date February 06, 2019

- @ericholscher: Support passing an explicit `index_name` for search indexing (#5239)
- @davidfischer: Tweak some ad styles (#5237)
- @stsewd: Fix conda issue link (#5226)
- @humitos: Add Santos to the development team (#5224)
- @ericholscher: Update our GSOC page for 2019 (#5210)
- @humitos: Do not allow to merge 'Status: blocked' PRs (#5205)
- @stsewd: Inject user to middleware tests (#5203)
- @ericholscher: Remove approvals requirement from mergable (#5200)
- @agjohnson: Update project notification copy to past tense (#5199)
- @stsewd: Remove feature flag for v2 (#5198)
- @ericholscher: Refactor search code (#5197)
- @stsewd: Update mergeable settings to v2 (#5196)
- @stsewd: Fix mergeable bot (#5195)
- @stsewd: Fix broken links for badges (#5190)
- @dojutsu-user: Change badge style (#5189)
- @humitos: Allow `source_suffix` to be a dictionary (#5183)
- @humitos: Upgrade all packages removing py2 compatibility (#5179)
- @dojutsu-user: Small docs fix (#5176)
- @stsewd: Sync all services even if one social account fails (#5171)
- @ericholscher: Release 3.1.0 (#5170)
- @rvmzes: SyntaxError caused by comma in python3 (#5156)
- @humitos: Use latest docker images as default (#5155)
- @stsewd: Remove logic for guessing slug from an unregistered domain (#5143)
- @humitos: Allow custom 404.html on projects (#5130)
- @dojutsu-user: Docs for feature flag (#5043)
- @stsewd: Remove usage of `project.documentation_type` in tasks (#4896)
- @ericholscher: Reapply the Elastic Search upgrade to `master` (#4722)
- @stsewd: Config file v2 docs (#4451)
- @stsewd: Set python3 as default interpreter (#3581)

4.10.37 Version 3.1.0

This version greatly improves our search capabilities, thanks to the Google Summer of Code. We're hoping to have another version of search coming soon after this, but this is a large upgrade moving to the latest Elastic Search.

Date January 24, 2019

- @ericholscher: Fix docs build (#5164)
- @ericholscher: Release 3.0.0 (#5163)
- @stsewd: Fix tests on master (#5162)
- @dojutsu-user: Sort versions smartly everywhere (#5157)
- @stsewd: Allow query params in redirects (#5081)
- @dojutsu-user: Implement get objects or log (#4900)
- @stsewd: Remove usage of project.documentation_type in tasks (#4896)
- @ericholscher: Reapply the Elastic Search upgrade to master (#4722)

4.10.38 Version 3.0.0

Read the Docs now only supports Python 3.6+. This is for people running the software on their own servers, builds continue to work across all supported Python versions.

Date January 23, 2019

- @stsewd: Fix tests on master (#5162)
- @dojutsu-user: Sort versions smartly everywhere (#5157)
- @rvmzes: SyntaxError caused by comma in python3 (#5156)
- @ericholscher: Fix Sphinx conf.py inserts (#5150)
- @ericholscher: Upgrade recommonmark to latest and fix integration (#5146)
- @stsewd: Fix requirements for local installation (#5138)
- @ericholscher: Fix local-docs-build requirements (#5136)
- @humitos: Upgrade all dependencies (#5134)
- @humitos: Configuration file for ProBot Mergeable Bot (#5132)
- @xavfernandez: docs: fix integration typos (#5128)
- @Hamdy722: Update LICENSE (#5125)
- @stsewd: Remove doctype from search (#5121)
- @humitos: Validate mkdocs.yml config on values that we manipulate (#5119)
- @humitos: Use 2019 in our README (#5117)
- @stsewd: Remove dead code from config module (#5116)
- @ericholscher: Check that the repo exists before trying to get a git commit (#5115)
- @ericholscher: Release 2.8.5 (#5111)
- @stsewd: Use the python path from virtualenv in Conda (#5110)
- @humitos: Feature flag to use readthedocs/build:testing image (#5109)

- @stsewd: Use python from virtualenv's bin directory when executing commands (#5107)
- @dojutsu-user: Split requirements/pip.txt (#5100)
- @humitos: Do not list banned projects under /projects/ (#5097)
- @humitos: Do not build projects from banned users (#5096)
- @humitos: Support custom robots.txt (#5086)
- @stsewd: Allow query params in redirects (#5081)
- @davidfischer: Fire a signal for domain verification (eg. for SSL) (#5071)
- @humitos: Upgrade all code to be Python3 only (#5065)
- @dojutsu-user: Use default settings for Config object (#5056)
- @agjohnson: Allow large form posts via multipart encoded forms to command API (#5000)
- @dojutsu-user: Validate url from webhook notification (#4983)
- @dojutsu-user: Display error, using inbuilt notification system, if primary email is not verified (#4964)
- @dojutsu-user: Implement get objects or log (#4900)
- @humitos: CRUD for EnvironmentVariables from Project's admin (#4899)
- @stsewd: Remove usage of project.documentation_type in tasks (#4896)
- @dojutsu-user: Fix the failing domain deletion task (#4891)
- @stsewd: Remove unused validations from v1 config (#4883)
- @humitos: Appropriate logging when a LockTimeout for VCS is reached (#4804)
- @stsewd: Implement extended install option (#4740)
- @bansalnitish: Added a link to open new issue with prefilled details (#3683)

4.10.39 Version 2.8.5

Date January 15, 2019

- @stsewd: Use the python path from virtualenv in Conda (#5110)
- @humitos: Feature flag to use readthedocs/build:testing image (#5109)
- @stsewd: Use python from virtualenv's bin directory when executing commands (#5107)
- @humitos: Do not build projects from banned users (#5096)
- @agjohnson: Fix common pieces (#5095)
- @rainwoodman: Suppress progress bar of the conda command. (#5094)
- @humitos: Remove unused suggestion block from 404 pages (#5087)
- @humitos: Remove header nav (Login/Logout button) on 404 pages (#5085)
- @stsewd: Fix little typo (#5084)
- @agjohnson: Split up deprecated view notification to GitHub and other webhook endpoints (#5083)
- @humitos: Install ProBot (#5082)
- @stsewd: Update docs about contributing to docs (#5077)
- @humitos: Declare and improve invoke tasks (#5075)

- @davidfischer: Fire a signal for domain verification (eg. for SSL) (#5071)
- @agjohnson: Update copy on notifications for github services deprecation (#5067)
- @humitos: Upgrade all packages with pur (#5059)
- @dojutsu-user: Reduce logging to sentry (#5054)
- @discover: fixed missing apostrophe for possessive “project’s” (#5052)
- @dojutsu-user: Template improvements in “gold/subscription_form.html” (#5049)
- @merwok: Fix link in features page (#5048)
- @stsewd: Update webhook docs (#5040)
- @stsewd: Remove sphinx static and template dir (#5039)
- @stephenfin: Add temporary method for disabling shallow cloning (#5031) (#5036)
- @stsewd: Raise exception in failed checkout (#5035)
- @dojutsu-user: Change default_branch value from Version.slug to Version.identifier (#5034)
- @humitos: Make wipe view not CSRF exempt (#5025)
- @humitos: Convert an IRI path to URI before setting as NGINX header (#5024)
- @safwanrahman: index project asynchronously (#5023)
- @stsewd: Keep command output when it’s killed (#5015)
- @stsewd: More hints for invalid submodules (#5012)
- @ericholscher: Release 2.8.4 (#5011)
- @stsewd: Remove auto doctype (#5010)
- @davidfischer: Tweak sidebar ad priority (#5005)
- @stsewd: Replace git status and git submodules status for gitpython (#5002)
- @davidfischer: Backport jquery 2432 to Read the Docs (#5001)
- @stsewd: Refactor remove_dir (#4994)
- @humitos: Skip builds when project is not active (#4991)
- @dojutsu-user: Make \$ unselectable in docs (#4990)
- @dojutsu-user: Remove deprecated “models.permalink” (#4975)
- @dojutsu-user: Add validation for tags of length greater than 100 characters (#4967)
- @dojutsu-user: Add test case for send_notifications on VersionLockedError (#4958)
- @dojutsu-user: Remove trailing slashes on svn checkout (#4951)
- @stsewd: Safe symlink on version deletion (#4937)
- @humitos: CRUD for EnvironmentVariables from Project’s admin (#4899)
- @humitos: Notify users about the usage of deprecated webhooks (#4898)
- @dojutsu-user: Disable django guardian warning (#4892)
- @humitos: Handle 401, 403 and 404 status codes when hitting GitHub for webhook (#4805)

4.10.40 Version 2.8.4

Date December 17, 2018

- @davidfischer: Tweak sidebar ad priority (#5005)
- @davidfischer: Backport jquery 2432 to Read the Docs (#5001)
- @ericholscher: Remove codecov comments and project coverage CI status (#4996)
- @stsewd: Remove LOCAL_GIT_BRANCHES from settings (#4993)
- @dojutsu-user: Link update on FAQ page (#4988)
- @ericholscher: Only use remote branches for our syncing. (#4984)
- @humitos: Sanitize output and chunk it at DATA_UPLOAD_MAX_MEMORY_SIZE (#4982)
- @humitos: Modify DB field for container_time_limit to be an integer (#4979)
- @dojutsu-user: Remove deprecated imports from “urlresolvers” (#4976)
- @davidfischer: Workaround for a django-storages bug (#4963)
- @ericholscher: Release 2.8.3 (#4961)
- @monsij: Remove -e option (#4960)
- @nutann3: Update “install Sphinx” URL (#4959)
- @stsewd: Shallow git clone (#4939)
- @dojutsu-user: Validate profile form fields (#4910)
- @davidfischer: Calculate actual ad views (#4885)
- @humitos: Allow all /api/v2/ CORS if the Domain is known (#4880)
- @dojutsu-user: Disable django.security.DisallowedHost from logging (#4879)
- @dojutsu-user: Remove ‘Sphinx Template Changes’ From Docs (#4878)
- @dojutsu-user: Make form for adopting project a choice field (#4841)
- @dojutsu-user: Add ‘Branding’ under the ‘Business Info’ section and ‘Guidelines’ on ‘Design Docs’ (#4830)
- @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)
- @dojutsu-user: Remove /embed API endpoint (#4771)
- @dojutsu-user: Improve unexpected error message when build fails (#4754)
- @dojutsu-user: Change the way of using login_required decorator (#4723)
- @dojutsu-user: Fix the form for adopting a project (#4721)

4.10.41 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)

4.10.42 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 'VersionLockedTimeout' to 'VersionLockedError' in comment. (#4859)
- @stsewd: Hide "edit on" when the version is a tag (#4851)
- @stsewd: Delete untracked tags on fetch (#4811)

- @humitos: Appropriate 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)
- @chirath: Webhook notification URL size validation check (#3680)

4.10.43 Version 2.8.1

Date November 06, 2018

- @ericholscher: Fix migration name on modified date migration (#4867)
- @dojutsu-user: Change 'VersionLockedTimeout' 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)

4.10.44 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 RequestContext (#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)

4.10.45 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)

4.10.46 Version 2.7.1

Date October 04, 2018

- @ericholscher: Revert “Merge pull request #4636 from readthedocs/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)

4.10.47 Version 2.7.0

Date September 29, 2018

Reverted, do not use

4.10.48 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 hiding “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)

4.10.49 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)

4.10.50 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 multiline 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/readthedocs/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)

4.10.51 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)

4.10.52 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)

4.10.53 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)

4.10.54 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: Clarify 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)

4.10.55 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)

4.10.56 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)

4.10.57 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 AttributeError: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)

4.10.58 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)

4.10.59 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)

4.10.60 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)
- @techtunik: 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)

4.10.61 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)

4.10.62 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)

4.10.63 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)

4.10.64 Version 2.3.10

Date April 24, 2018

- @humitos: Downgrade docker to 3.1.3 (#4003)

4.10.65 Version 2.3.9

Date April 20, 2018

- @agjohnson: Fix recursion problem more generally (#3989)

4.10.66 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)

4.10.67 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 pip 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)

4.10.68 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)

4.10.69 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)

4.10.70 Version 2.3.4

- Release for static assets

4.10.71 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)

4.10.72 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.

4.10.73 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)
- @rix: Fix page redirect preview (#2711)

4.10.74 Version 2.3.0

Warning: Version 2.3.0 includes a security fix for project translations. See [Release 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)

4.10.75 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)

4.10.76 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)

4.10.77 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)

4.10.78 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)

4.10.79 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 psycpg2 (#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)

4.10.80 Version 2.1.3

date Dec 21, 2017

- @ericholscher: Upgrade psycpg2 (#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)

4.10.81 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)

4.10.82 Version 2.1.1

Release information missing

4.10.83 Version 2.1.0

- @ericholscher: Revert "Merge pull request #3336 from readthedocs/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)
- @josejrobes: 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)

4.10.84 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 preference 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)

4.10.85 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/lukaszbdjango-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 psycpg2 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 psycpg2. Also it's only a minor version upgrade. Release notes: <http://initd.org/psycpg/docs/news.html#what-s-new-in-psycpg-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
```

4.11 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:

- *Support Team*
- *Operations Team*
- *Development Team*

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.

4.11.1 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

Support team members

- [Eric Holscher](#) (Pacific Time)
- [Anthony Johnson](#) (Mountain Time)
- [Manuel Kaufmann](#) (Central Time)
- Your Name Here

Please don't email us personally for support on Read the Docs. You can email support@readthedocs.org for any issues you may have.

Joining the support team

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.

4.11.2 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.

Ops team 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 the ops team

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.

4.11.3 Development Team

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

Dev team members

- Eric Holscher
- Anthony Johnson
- Manuel Kaufmann
- David Fischer
- Santos Gallegos
- Your name here

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

Joining the dev team

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.

4.12 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.

4.12.1 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>

4.12.2 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

4.12.3 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.

4.13 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.

4.14 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* although you will still see *community ads*. You can go ad-free by [becoming a Gold member](#) or a [Supporter](#) of Read the Docs. Gold members can also remove advertising from their projects for all visitors.

For businesses looking to remove advertising, please consider *[Read the Docs for Business](#)*.

4.14.1 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 Community Ads program will highlight some of these projects.

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. Please [complete an application](#) to be considered for our Community Ads program. If you have any questions about our community ads program, feel free to [send us an email](#).

Opting Out

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

1. You can go completely ad-free by becoming a [Gold member](#) or a [Supporter](#). Additionally, Gold members may remove advertising from their projects for all visitors.
2. You 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**.

You will still see *community ads* for open source projects and conferences.

3. Project owners can also opt out of paid advertisements for their projects. You can change these options:
 - Go to your **project** page (`/projects/<slug>/`)
 - Go to *Admin > Advertising*

- Change your advertising settings
4. If you are part of a company that uses Read the Docs to host documentation for a commercial product, we offer [Read the Docs for Business](#) that offers a completely ad-free experience, additional build resources, and other great features like CDN support and private documentation.
 5. If you would like to completely remove advertising from your open source project, but our commercial plans don't seem like the right fit, please [get in touch](#) to discuss alternatives to advertising.

4.14.2 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
- 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.

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.

4.14.3 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*.

4.15 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.

4.15.1 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)

4.15.2 Past sponsors

- [Python Software Foundation](#)
- [Revsys](#)
- [Mozilla Web Dev](#)
- [Django Software Foundation](#)
- [Lab305](#)
- [Twilio](#)
- [Rackspace](#)
- [Mozilla](#)

4.15.3 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.

4.16 Read the Docs for Business

Read the Docs is our community solution for open source projects at readthedocs.org and we offer Read the Docs for Business for building and hosting commercial documentation at readthedocs.com. Features in this section are specific to Read the Docs for Business.

Private repositories and private documentation The largest difference between the community solution and our commercial offering is the ability to connect to private repositories, to restrict documentation access to certain users, or to share private documentation via private hyperlinks.

Additional build resources Do you have a complicated build process that uses large amounts of CPU, memory, disk, or networking resources? Our commercial offering has much higher default resources that result in faster documentation build times and we can increase it further for very demanding projects.

Priority support We have a dedicated support team that responds to support requests during business hours. If you need a quick turnaround, please signup for readthedocs.com.

Advertising-free All commercially hosted documentation is always ad-free.

4.16.1 Custom Domains

Note: These directions are for projects hosted on Read the Docs for Business. For setting up a custom domain on a project hosted on readthedocs.org, please read our [community documentation](#).

Subdomain support

Once a project is imported into Read the Docs, by default it's hosted under a subdomain on one of our domains. If you need a custom domain, continue on to custom domain setup.

Serving documentation with a custom domain

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 *Admin > Advance Settings*.
- Add a Domain in the *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 require SSL for custom domains. During the setup process, you will need to add a record to your DNS which will allow us to issue an SSL certificate for your custom domain.

4.16.2 Organizations

Note: This feature only exists on [Read the Docs for Business](#).

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.

4.16.3 Sharing

Note: This feature only exists on [Read the Docs for Business](#).

You can share your project with users outside of your company:

- by sending them a *secret link*,
- by giving them a *password*.

These methods will allow them to view a specific project inside your company.

Additionally, you can use a HTTP Authorization Header. This is useful to have access from a script.

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, password, or HTTP header token), 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
 - For HTTP header token, you need to pass the `Authorization` header in your HTTP request.
- 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 an *Authorization required* page asking them for the password you generated. When the user enters the password, they will have access to view your project.

HTTP Authorization Header

Token Authorization

You need to send the `Authorization` header with the token on each HTTP request. The header has the form `Authorization: Token <ACCESS_TOKEN>`. For example:

```
$ curl -H "Authorization: Token 19okmz5k0i6yk17jp70jlnv91v" https://docs.example.com/en/latest/example.html
```

Basic Authorization

You can also use basic authorization, with the token as user and an empty password. For example:

```
$ curl --url https://docs.example.com/en/latest/example.html --user '19okmz5k0i6yk17jp70jlnv91v:'
```

4.16.4 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.

4.17 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.

If your main reasons for considering a custom installation are to connect to private repositories or to have fine grained access control over your documentation, please consider [Read the Docs for Business](#). It has those features and more!

Warning: These documents are maintained by the community, and might not be up to date. Read the Docs developers do not support custom installs of our software, as mentioned in our [Read the Docs Open Source Philosophy](#).

4.17.1 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 = '%s'.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](#).

4.17.2 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/readthedocs/readthedocs.org.git
$ cd readthedocs.org
$ sudo pip install -r requirements.txt
```

Possible errors with a local RTD setup

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 errors administering a RTD server

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

	POST /api/v3/projects/pip/environmentvariables/, 105
GET /api/v2/build/,83	
GET /api/v2/build/(int:id)/,84	POST /api/v3/projects/pip/redirects/, 101
GET /api/v2/project/,80	
GET /api/v2/project/(int:id)/,80	PUT /api/v3/projects/(str:project_slug)/redirects/, 102
GET /api/v2/project/(int:id)/active_versions/, 81	DELETE /api/v3/projects/(str:project_slug)/environmentvariables/, 106
GET /api/v2/version/,82	
GET /api/v2/version/(int:id)/,82	DELETE /api/v3/projects/(str:project_slug)/redirects/, 103
GET /api/v3/projects/,87	
GET /api/v3/projects/(str:project_slug)/builds/, 95	DELETE /api/v3/projects/(str:project_slug)/subprojects/, 99
GET /api/v3/projects/(str:project_slug)/builds/(int:build_id)/,82	PATCH /api/v3/projects/(string:project_slug)/, 90
GET /api/v3/projects/(str:project_slug)/environmentvariables/, 104	PATCH /api/v3/projects/(string:project_slug)/versions/, 93
GET /api/v3/projects/(str:project_slug)/environmentvariables/(int:environmentvariable_id)/, 104	
GET /api/v3/projects/(str:project_slug)/redirects/, 101	POST /repos/:owner/:repo/statuses/:sha, 131
GET /api/v3/projects/(str:project_slug)/redirects/(int:redirect_id)/, 100	
GET /api/v3/projects/(str:project_slug)/subprojects/, 97	
GET /api/v3/projects/(str:project_slug)/subprojects/(str:alias_slug)/, 97	
GET /api/v3/projects/(str:project_slug)/translations/, 99	
GET /api/v3/projects/(string:project_slug)/, 88	
GET /api/v3/projects/(string:project_slug)/versions/, 91	
GET /api/v3/projects/(string:project_slug)/versions/(string:version_slug)/, 92	
POST /api/v3/projects/,89	
POST /api/v3/projects/(str:project_slug)/subprojects/, 98	
POST /api/v3/projects/(string:project_slug)/versions/(string:version_slug)/builds/, 96	

E

environment variable
 READTHEDOCS, [36](#)

P

Python Enhancement Proposals
 PEP 440, [49](#)

R

READTHEDOCS, [36](#)