Read the Docs simplifies software documentation by building, versioning, and hosting of your docs, automatically. 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. Read more about VCS Integrations.

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. Read more about Versioned Documentation.

Open Source and User Focused Our code is free and open source. Our company is bootstrapped and 100% user focused. Read the Docs Community hosts documentation for over 100,000 large and small open source projects, in almost every human and computer language. Read the Docs for Business supports hundreds of organizations with product and internal documentation.

You can find out more about our all the Read the Docs features in these pages.
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.

- **Tutorial**: Read the Docs tutorial
- **Getting started**: With Sphinx | With MkDocs | Feature Overview | Choosing Between Our Two Sites | Glossary
- **Importing your existing documentation**: Import guide

## 1.1 Read the Docs tutorial

In this tutorial you will create a documentation project on Read the Docs by importing an Sphinx project from a GitHub repository, tailor its configuration, and explore several useful features of the platform.

The tutorial is aimed at people interested in learning how to use Read the Docs to host their documentation projects. You will fork a fictional software library similar to the one developed in the official Sphinx tutorial. No prior experience with Sphinx is required, and you can follow this tutorial without having done the Sphinx one.

The only things you will need to follow are a web browser, an Internet connection, and a GitHub account (you can register for a free account if you don’t have one). You will use Read the Docs Community, which means that the project will be public.

### 1.1.1 Getting started

**Preparing your project on GitHub**

To start, sign in to GitHub and navigate to the tutorial GitHub template, where you will see a green *Use this template* button. Click it to open a new page that will ask you for some details:

- Leave the default “Owner”, or change it to something better for a tutorial project.
- Introduce an appropriate “Repository name”, for example *rtd-tutorial*.
- Make sure the project is “Public”, rather than “Private”.

After that, click on the green *Create repository from template* button, which will generate a new repository on your personal account (or the one of your choosing). This is the repository you will import on Read the Docs, and it contains the following files:

- **README.rst** Basic description of the repository, you will leave it untouched.
- **pyproject.toml** Python project metadata that makes it installable. Useful for automatic documentation generation from sources.
Sign up for Read the Docs

To sign up for a Read the Docs account, navigate to the Sign Up page and choose the option Sign up with GitHub. On the authorization page, click the green Authorize readthedocs button.

Note: Read the Docs needs elevated permissions to perform certain operations that ensure that the workflow is as smooth as possible, like installing webhooks. If you want to learn more, check out Permissions for connected accounts.

After that, you will be redirected to Read the Docs, where you will need to confirm your e-mail and username. Clicking the Sign Up » button will create your account and redirect you to your dashboard.

By now, you should have two email notifications:

- One from GitHub, telling you that “A third-party OAuth application … was recently authorized to access your account”. You don’t need to do anything about it.
- Another one from Read the Docs, prompting you to “verify your email address”. Click on the link to finalize the process.

Finally, you created your account on Read the Docs and are ready to import your first project.

Welcome!

Note: Our commercial site offers some extra features, like support for private projects. You can learn more about our two different sites.
Authorize Read the Docs Community (readthedocs.org)

- **Read the Docs Community (readthedocs.org) by readthedocs**
  - wants to access your astrolj-test account

- **Repository webhooks and services**
  - Admin access

- **Organizations and teams**
  - Read-only access

- **Commit statuses**
  - Read and write access

- **Personal user data**
  - Email addresses (read-only)

[Authorize readthedocs]

Fig. 2: GitHub authorization page
1.1.2 First steps

**Importing the project to Read the Docs**

To import your GitHub project to Read the Docs, first click on the *Import a Project* button on your dashboard (or browse to the import page directly). You should see your GitHub account under the “Filter repositories” list on the right. If the list of repositories is empty, click the button, and after that all your repositories will appear on the center.

Fig. 4: Import projects workflow

Locate your *rtd-tutorial* project (possibly clicking *next ››* at the bottom if you have several pages of projects), and then click on the button to the right of the name. The next page will ask you to fill some details about your Read the Docs project:

- **Name** The name of the project. It has to be unique across all the service, so it is better if you prepend your username, for example *astrojuanlu-rtd-tutorial*.
- **Repository URL** The URL that contains the sources. Leave the automatically filled value.
- **Repository type** Version control system used, leave it as “Git”.
- **Default branch** Name of the default branch of the project, leave it as *main*.
- **Edit advanced project options** Leave it unchecked, we will make some changes later.

After hitting the *Next* button, you will be redirected to the *project home*. You just created your first project on Read the Docs!
Checking the first build

Read the Docs will try to build the documentation of your project right after you create it. To see the build logs, click on the Your documentation is building link on the project home, or alternatively navigate to the “Builds” page, then open the one on top (the most recent one).

If the build has not finished yet by the time you open it, you will see a spinner next to a “Installing” or “Building” indicator, meaning that it is still in progress.

When the build finishes, you will see a green “Build completed” indicator, the completion date, the elapsed time, and a link to see the corresponding documentation. If you now click on View docs, you will see your documentation live!

Note: Advertisement is one of our main sources of revenue. If you want to learn more about how do we fund our operations and explore options to go ad-free, check out our Sustainability page.

If you don’t see the ad, you might be using an ad blocker. Our Ethical Ads network respects your privacy, doesn’t target you, and tries to be as unobtrusive as possible, so we would like to kindly ask you to not block us
Configure your documentation builds! Adding a `readthedocs.yml` file to your project is the recommended way to configure your documentation builds. You can declare dependencies, set up submodules, and many other great features.

Fig. 6: First successful documentation build

Fig. 7: HTML documentation live on Read the Docs
Basic configuration changes

You can now proceed to make some basic configuration adjustments. Navigate back to the project page and click on the Admin button, which will open the Settings page.

First of all, add the following text in the description:

Lumache (/luˈmake/) is a Python library for cooks and food lovers that creates recipes mixing random ingredients.

Then set the project homepage to https://world.openfoodfacts.org/, and write food, python in the list of tags. All this information will be shown on your project home.

After that, configure your email so you get a notification if the build fails. To do so, click on the Notifications link on the left, type the email where you would like to get the notification, and click the Add button. After that, your email will be shown under “Existing Notifications”.

Trigger a build from a pull request

Read the Docs allows you to trigger builds from GitHub pull requests and gives you a preview of how the documentation would look like with those changes.

To enable that functionality, first click on the Advanced Settings link on the left under the Admin menu, check the “Build pull requests for this project” checkbox, and click the Save button at the bottom of the page.

Next, navigate to your GitHub repository, locate the file docs/source/index.rst, and click on the icon on the top-right with the tooltip “Edit this file” to open a web editor (more information on their documentation).

Fig. 8: File view on GitHub before launching the editor

In the editor, add the following sentence to the file:

Listing 1: docs/source/index.rst

Lumache has its documentation hosted on Read the Docs.

Write an appropriate commit message, and choose the “Create a new branch for this commit and start a pull request” option, typing a name for the new branch. When you are done, click the green Propose changes button, which will take you to the new pull request page, and there click the Create pull request button below the description.
After opening the pull request, a Read the Docs check will appear indicating that it is building the documentation for that pull request. If you click on the Details link while it is building, you will access the build logs, otherwise it will take you directly to the documentation. When you are satisfied, you can merge the pull request!

1.1.3 Customizing the build process

The Settings page of the project home allows you to change some global configuration values of your project. In addition, you can further customize the building process using the .readthedocs.yaml configuration file. This has several advantages:

- The configuration lives next to your code and documentation, tracked by version control.
- It can be different for every version (more on versioning in the next section).
- Some configurations are only available using the config file.

Read the Docs works without this configuration by making some decisions on your behalf. For example, what Python version to use, how to install the requirements, and others.

Tip: Settings that apply to the entire project are controlled in the web dashboard, while settings that are version or build specific are better in the YAML file.

Upgrading the Python version

For example, to explicitly use Python 3.8 to build your project, navigate to your GitHub repository, click on the Add file button, and add a .readthedocs.yaml file with these contents to the root of your project:

Listing 2: .readthedocs.yaml

```yaml
version: 2
build:
  os: "ubuntu-20.04"
  tools:
    python: "3.8"
```

The purpose of each key is:

version Mandatory, specifies version 2 of the configuration file.
build.os Required to specify the Python version, states the name of the base image.
**build.tools.python** Declares the Python version to be used.

After you commit these changes, go back to your project home, navigate to the “Builds” page, and open the new build that just started. You will notice that one of the lines contains `python3.8`: if you click on it, you will see the full output of the corresponding command, stating that it used Python 3.8.6 to create the virtual environment.

Fig. 10: Read the Docs build using Python 3.8

**Making warnings more visible**

If you navigate to your HTML documentation, you will notice that the index page looks correct, but actually the API section is empty. This is a very common issue with Sphinx, and the reason is stated in the build logs. On the build page you opened before, click on the `View raw` link on the top right, which opens the build logs in plain text, and you will see several warnings:

```plaintext
WARNING: [autosummary] failed to import 'lumache': no module named lumache
...
WARNING: autodoc: failed to import function 'get_random_ingredients' from module 'lumache
`'; the following exception was raised:
No module named 'lumache'
WARNING: autodoc: failed to import exception 'InvalidKindError' from module 'lumache';␣
`the following exception was raised:
No module named 'lumache'
```

To spot these warnings more easily and allow you to address them, you can add the `sphinx.fail_on_warning` option to your Read the Docs configuration file. For that, navigate to GitHub, locate the `.readthedocs.yaml` file you created earlier, click on the `icon, and add these contents:

```
Listing 3: .readthedocs.yaml

version: 2
build:
  os: "ubuntu-20.04"
  tools:
    python: "3.8"
sphinx:
  fail_on_warning: true
```

At this point, if you navigate back to your “Builds” page, you will see a **Failed build**, which is exactly the intended result: the Sphinx project is not properly configured yet, and instead of rendering an empty API page, now the build fails.
The reason `sphinx.ext.autosummary` and `sphinx.ext.autodoc` fail to import the code is because it is not installed. Luckily, the `.readthedocs.yaml` also allows you to specify which requirements to install.

To install the library code of your project, go back to editing `.readthedocs.yaml` on GitHub and modify it as follows:

```python
python:
  # Install our python package before building the docs
  install:
    - method: pip
      path: .
```

With this change, Read the Docs will install the Python code before starting the Sphinx build, which will finish seamlessly. If you go now to the API page of your HTML documentation, you will see the `lumache` summary!

### Enabling PDF and EPUB builds

Sphinx can build several other formats in addition to HTML, such as PDF and EPUB. You might want to enable these formats for your project so your users can read the documentation offline.

To do so, add this extra content to your `.readthedocs.yaml`:

```yaml
sphinx:
  fail_on_warning: true

formats:
  - pdf
  - epub
```

After this change, PDF and EPUB downloads will be available both from the “Downloads” section of the project home, as well as the flyout menu.

Fig. 11: Downloads available from the flyout menu
1.1.4 Versioning documentation

Read the Docs allows you to have several versions of your documentation, in the same way that you have several versions of your code. By default, it creates a latest version that points to the default branch of your version control system (main in the case of this tutorial), and that’s why the URLs of your HTML documentation contain the string /latest/.

Creating a new version

Let’s say you want to create a 1.0 version of your code, with a corresponding 1.0 version of the documentation. For that, first navigate to your GitHub repository, click on the branch selector, type 1.0.x, and click on “Create branch: 1.0.x from ‘main’” (more information on their documentation).

Next, go to your project home, click on the Versions button, and under “Active Versions” you will see two entries:

- The latest version, pointing to the main branch.
- A new stable version, pointing to the origin/1.0.x branch.

![Active Versions]

Fig. 12: List of active versions of the project

Right after you created your branch, Read the Docs created a new special version called stable pointing to it, and started building it. When the build finishes, the stable version will be listed in the flyout menu and your readers will be able to choose it.

**Note:** Read the Docs follows some rules to decide whether to create a stable version pointing to your new branch or tag. To simplify, it will check if the name resembles a version number like 1.0, 2.0.3 or 4.x.

Now you might want to set stable as the default version, rather than latest, so that users see the stable documentation when they visit the root URL of your documentation (while still being able to change the version in the flyout menu).

For that, go to the Advanced Settings link under the Admin menu of your project home, choose stable in the “Default version” dropdown, and hit Save at the bottom. Done!

Modifying versions

Both latest and stable are now active, which means that they are visible for users, and new builds can be triggered for them. In addition to these, Read the Docs also created an inactive 1.0.x version, which will always point to the 1.0.x branch of your repository.

Let’s activate the 1.0.x version. For that, go to the “Versions” on your project home, locate 1.0.x under “Activate a version”, and click on the Activate button. This will take you to a new page with two checkboxes, “Active” and “Hidden”. Check only “Active”, and click Save.

After you do this, 1.0.x will appear on the “Active Versions” section, and a new build will be triggered for it.

1.1. Read the Docs tutorial 13
Show a warning for old versions

When your project matures, the number of versions might increase. Sometimes you will want to warn your readers when they are browsing an old or outdated version of your documentation.

To showcase how to do that, let’s create a 2.0 version of the code: navigate to your GitHub repository, click on the branch selector, type 2.0.x, and click on “Create branch: 2.0.x from ‘main’”. This will trigger two things:

- Since 2.0.x is your newest branch, stable will switch to tracking it.
- A new 2.0.x version will be created on your Read the Docs project.
- Since you already have an active stable version, 2.0.x will be activated.

From this point, 1.0.x version is no longer the most up to date one. To display a warning to your readers, go to the Admin menu of your project home, click on the Advanced Settings link on the left, enable the “Show version warning” checkbox, and click the Save button.

If you now browse the 1.0.x documentation, you will see a warning on top encouraging you to browse the latest version instead. Neat!

1.1.5 Getting insights from your projects

Once your project is up and running, you will probably want to understand how readers are using your documentation, addressing some common questions like:

- what pages are the most visited pages?
- what search terms are the most frequently used?
- are readers finding what they look for?

Read the Docs offers you some analytics tools to find out the answers.
Welcome to Lumache's documentation!

Lumache (/luˈmaʃe/) is a Python library for cooks and food lovers that creates recipes mixing random ingredients. It pulls data from the Open Food Facts database and offers a simple and intuitive API.

![Warning for old versions](image)

**Browsing Traffic Analytics**

The *Traffic Analytics* view shows the top viewed documentation pages of the past 30 days, plus a visualization of the daily views during that period. To generate some artificial views on your newly created project, you can first click around the different pages of your project, which will be accounted immediately for the current day statistics.

To see the Traffic Analytics view, go back the *project page* again, click on the *Admin* button, and then click on the *Traffic Analytics* section. You will see the list of pages in descending order of visits, as well as a plot similar to the one below.

![Traffic Analytics plot](image)

**Note:** The Traffic Analytics view explained above gives you a simple overview of how your readers browse your documentation. It has the advantage that it stores no identifying information about your visitors, and therefore it respects their privacy. However, you might want to get more detailed data by enabling Google Analytics. Notice though that we take some extra measures to **respect user privacy** when they visit projects that have Google Analytics enabled, and this might reduce the number of visits counted.
Finally, you can also download this data for closer inspection. To do that, scroll to the bottom of the page and click on the *Download all data* button. That will prompt you to download a CSV (Comma-Separated Values) file that you can process any way you want.

**Browsing Search Analytics**

Apart from traffic analytics, Read the Docs also offers the possibility to inspect *what search terms your readers use* on your documentation. This can inform decisions on what areas to reinforce, or what parts of your project are less understood or more difficult to find.

To generate some artificial search statistics on the project, go to the HTML documentation, locate the Sphinx search box on the left, type `ingredients`, and press the *Enter* key. You will be redirected to the search results page, which will show two entries.

Next, go back to the *Admin* section of your project page, and then click on the *Search Analytics* section. You will see a table with the most searched queries (including the `ingredients` one you just typed), how many results did each query return, and how many times it was searched. Below the queries table, you will also see a visualization of the daily number of search queries during the past 30 days.

![Search Analytics](image)

*Fig. 16: Most searched terms*

Like the Traffic Analytics, you can also download the whole dataset in CSV format by clicking on the *Download all data* button.

**1.1.6 Where to go from here**

This is the end of the tutorial. You started by forking a GitHub repository and importing it on Read the Docs, building its HTML documentation, and then went through a series of steps to customize the build process, tweak the project configuration, and add new versions.

Here you have some resources to continue learning about documentation and Read the Docs:

- You can learn more about the functionality of the platform by going over our *Read the Docs features* page.
- To make the most of the documentation generators that are supported, you can read the *Sphinx tutorial* or the *MkDocs User Guide*.
- Whether you are a documentation author, a project administrator, a developer, or a designer, you can follow our how-to guides that cover specific tasks, available under `/guides/index`.
• You can check out some of the Advanced features of Read the Docs, like Subprojects or Automation Rules, to name a few.

• For private project support and other enterprise features, you can use our commercial service (and if in doubt, check out Choosing Between Our Two Sites).

• Do you want to join a global community of fellow documentarians? Check out Write the Docs and its Slack workspace.

• Do you want to contribute to Read the Docs? We greatly appreciate it! Check out Building and Contributing to Documentation.

Happy documenting!

1.2 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

If you want to learn more about how to create your first Sphinx project, read on. If you are interested in exploring the Read the Docs platform using an already existing Sphinx project, check out Read the Docs tutorial.

1.2.1 Quick start

See also:

If you already have a Sphinx project, check out our Importing Your Documentation guide.

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.

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

1.2.2 Using Markdown with Sphinx

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

$ pip install myst-parser

Then in your conf.py:

```
extensions = ['myst_parser']
```

You can now continue writing your docs in .md files and it will work with Sphinx. Read the Getting started with MyST in Sphinx docs for additional instructions.
1.2.3 External resources

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

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

1.3 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.

1.3.1 Quick start

See also:

If you already have a Mkdocs project, check out our Importing Your Documentation guide.

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.

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

For full documentation visit mkdocs.org.

Commands

- `mkdocs new [dir-name]` - Create a new project.
- `mkdocs serve` - Start the live-reloading docs server.
- `mkdocs build` - Build the documentation site.
- `mkdocs help` - Print this help message.

Project layout

```
mkdocs.yml  # The configuration file.
docs/
    index.md # The documentation homepage.
    ...      # Other markdown pages, images and other files.
```

Fig. 18: Your MkDocs project is built
1.3.2 External resources

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

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

1.4 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.

1.4.1 Automatically import your docs

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.

1.4.2 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.
1.4.3 Building your documentation

Within a few seconds of completing the import process, your code will automatically be fetched from your 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.yaml 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. We recommend to set the version your project uses explicitly.

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

If you have any more trouble, don’t hesitate to reach out to us. The Site Support page has more information on getting in touch.
1.5 Read the Docs features

Read the Docs offers a number of platform features that are possible because we both build and host documentation for you.

1.5.1 Automatic Documentation Deployment

We integrate with GitHub, BitBucket, and GitLab. We automatically create webhooks in your repository, which tell us whenever you push a commit. We will then build and deploy your docs every time you push a commit. This enables a workflow that we call Continuous Documentation:

Once you set up your Read the Docs project, your users will always have up-to-date documentation.

Learn more about VCS Integrations.

1.5.2 Custom Domains & White Labeling

When you import a project to Read the Docs, we assign you a URL based on your project name. You are welcome to use this URL, but we also fully support custom domains for all our documentation projects.

Learn more about Custom Domains and White Labeling.

1.5.3 Versioned Documentation

We support multiple versions of your documentation, so that users can find the exact docs for the version they are using. We build this on top of the version control system that you’re already using. Each version on Read the Docs is just a tag or branch in your repository.

You don’t need to change how you version your code, we work with whatever process you are already using. If you don’t have a process, we can recommend one.

Learn more about Versioned Documentation.
1.5.4 Downloadable Documentation

Read the Docs supports building multiple formats for Sphinx-based projects:

• PDF
• ePub
• Zipped HTML

This means that every commit that you push will automatically update your PDFs as well as your HTML.

This feature is great for users who are about to get on a plane and want offline docs, as well as being able to ship your entire set of documentation as one file.

Learn more about Downloadable Documentation.

1.5.5 Full-Text Search

We provide search across all the projects that we host. This actually comes in two different search experiences: dashboard search on the Read the Docs dashboard and in-doc search on documentation sites, using your own theme and our search results.

We offer a number of search features:

• Search across subprojects
• Search results land on the exact content you were looking for
• Search across projects you have access to (available on Read the Docs for Business)
• A full range of search operators including exact matching and excluding phrases.

Learn more about Server Side Search.

1.5.6 Open Source and Customer Focused

Read the Docs cares deeply about our customers and our community. As part of that commitment, all of the source code for Read the Docs is open source. This means there’s no vendor lock-in, and you are welcome to contribute the features you want or run your own instance.

Our bootstrapped company is owned and controlled by the founders, and fully funded by our customers and advertisers. That allows us to focus 100% of our attention on building the best possible product for you.

Learn more About Read the Docs.
1.6 Choosing Between Our Two Sites

A question our users often have is what the difference is between Read the Docs Community and Read the Docs for Business. This page will lay out the functional and philosophical differences between the two sites, which should help you choose which is a better fit for your organization.

The features available on both platforms are the same. The primary difference is the audience and use cases that are supported.

1.6.1 Read the Docs Community

Read the Docs Community is meant for open source projects to use for documentation hosting. This is great for user and developer documentation for your project.

Important points:

• All documentation sites have advertising
• Only supports public VCS repositories
• All documentation is publicly accessible to the world
• Less build time and fewer build resources (memory & CPU)
• Documentation is organized by projects

You can sign up for an account at https://readthedocs.org.

1.6.2 Read the Docs for Business

Read the Docs for Business is meant for companies and users who have private documentation. It works well for product documentation as well as internal docs for your developers.

Important points:

• No advertising
• Allows importing private and public repositories from VCS
• Supports private versions that only your organization or people you give access to can see
• More build time and more build resources (memory & CPU)
• Documentation is organized by organization, giving more control over permissions

You can sign up for an account at https://readthedocs.com.
1.6.3 Questions?

If you have a question about which platform would be best, you can email us at support@readthedocs.org.

1.7 Glossary

dashboard Main page where you can see all your projects with their build status and import a new project.

flyout menu Menu displayed on the documentation, readily accessible for readers, containing the list active versions, links to the static downloads, and other useful information.

profile page Page where you can see the projects of a certain user.

project home Page where you can access all the features of Read the Docs, from having an overview to browsing the latest builds or administering your project.

project page Another name for project home.

root URL Home URL of your documentation without the /<lang> and /<version> segments. For projects without custom domains, the one ending in .readthedocs.io/ (for example, https://docs.readthedocs.io as opposed to https://docs.readthedocs.io/en/latest).
READ THE DOCS FEATURE OVERVIEW

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

• **Overview of core features:** VCS Integrations | Custom Domains and White Labeling | Versioned Documentation | Downloadable Documentation | Documentation Hosting Features | Server Side Search | Traffic Analytics | Preview Documentation from Pull Requests | Build Notifications and Webhooks | User-defined Redirects | Security Log

• **Connecting with GitHub, BitBucket, or GitLab:** Connecting your VCS account

• **Read the Docs build process:** Configuration reference | Build process | Environment Variables | Badges

• **Troubleshooting:** Site Support | Frequently asked questions

### 2.1 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 should be named `.readthedocs.yaml`.

**Note:** Some other variants like readthedocs.yaml, .readthedocs.yml, etc are deprecated.

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

• Settings are per version rather than per project.

• Settings live in your VCS.

• They enable reproducible build environments over time.

• Some settings are only available using a configuration file

**Tip:** Using a configuration file is the recommended way of using Read the Docs.
2.1.1 Configuration File V2

Read the Docs supports configuring your documentation builds with a YAML file. The configuration file must be in the root directory of your project and be named .readthedocs.yaml.

All options are applied to the version containing this file. Below is an example YAML file which shows the most common configuration options:

Sphinx

MkDocs

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

# Required
version: 2

# Set the version of Python and other tools you might need
build:
  os: ubuntu-20.04
  tools:
    python: "3.9"
    # You can also specify other tool versions:
    # nodejs: "16"
    # rust: "1.55"
    # golang: "1.17"

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

# If using Sphinx, optionally build your docs in additional formats such as PDF
# formats:
#   - pdf

# Optionally declare the Python requirements required to build your docs
python:
  install:
    - requirements: docs/requirements.txt
```

(continues on next page)
mkdocs:
  configuration: mkdocs.yml

# Optionally declare the Python requirements required to build your docs
python:
  install:
    - requirements: docs/requirements.txt

Supported settings

Read the Docs validates every configuration file. Any configuration option that isn’t supported will make the build fail. This is to avoid typos and provide feedback on invalid configurations.

Warning: When using a v2 configuration file, the local settings from the web interface are ignored.

- version
- formats
- python
  - python.version
  - python.install
  * Requirements file
  * Packages
  - python.system_packages
- conda
  - conda.environment
- build
  - build.os
  - build.tools
  - build.tools.python
  - build.tools.nodejs
  - build.tools.rust
  - build.tools.golang
  - build.apt_packages
- sphinx
  - sphinx.builder
  - sphinx.configuration
  - sphinx.fail_on_warning
- mkdocs
version

Required true

Example:

```yaml
version: 2
```

Warning: If you don’t provide the version, v1 will be used.

formats

Additional formats of the documentation to be built, apart from the default HTML.

Type list

Options htmlzip, pdf, epub, all

Default []

Example:

```yaml
version: 2

# Default
formats: []
```

```yaml
version: 2

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

Note: You can use the all keyword to indicate all formats.
version: 2

# Build all formats
formats: all

Warning: At the moment, only Sphinx supports additional formats. pdf, epub, and htmlzip output is not yet supported when using MkDocs.

python

Configuration of the Python environment to be used.

version: 2

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

python.version

Warning: This option is now deprecated and replaced by build.tools.python. See python.version (legacy) for the description of this option.

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:**

```yaml
version: 2
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:**
version: 2

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

**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.

```
version: 2

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

Warning: This functionality is in beta. If you find any inconsistencies or have feedback, please open an issue.

Configuration for the documentation build process. This allows you to specify the base Read the Docs image used to build the documentation, and control the versions of several tools: Python, Node.js, Rust, and Go.

```yaml
version: 2
build:
  os: ubuntu-20.04
  tools:
    python: "3.9"
    nodejs: "16"
    rust: "1.55"
    golang: "1.17"
```

build.os

The Docker image used for building the docs. Image names refer to the operating system Read the Docs uses to build them.

Note: Arbitrary Docker images are not supported.

<table>
<thead>
<tr>
<th>Type</th>
<th>string</th>
</tr>
</thead>
<tbody>
<tr>
<td>Options</td>
<td>ubuntu-20.04</td>
</tr>
<tr>
<td>Required</td>
<td>true</td>
</tr>
</tbody>
</table>

build.tools

Version specifiers for each tool. It must contain at least one tool.

<table>
<thead>
<tr>
<th>Type</th>
<th>dict</th>
</tr>
</thead>
<tbody>
<tr>
<td>Options</td>
<td>python, nodejs, rust, golang</td>
</tr>
<tr>
<td>Required</td>
<td>true</td>
</tr>
</tbody>
</table>

build.tools.python

Python version to use. You can use several interpreters and versions, from CPython, PyPy, Miniconda, and Mamba.

Note: If you use Miniconda3 or Mambaforge, you can select the Python version using the environment.yml file. See our Conda Support guide for more information.

<table>
<thead>
<tr>
<th>Type</th>
<th>string</th>
</tr>
</thead>
</table>
Options

- 2.7
- 3 (last stable CPython version)
- 3.6
- 3.7
- 3.8
- 3.9
- 3.10
- pypy3.7
- miniconda3-4.7
- mambaforge-4.10

**build.tools.nodejs**

Node.js version to use.

*Type* string

*Options* 14, 16

**build.tools.rust**

Rust version to use.

*Type* string

*Options* 1.55

**build.tools.golang**

Go version to use.

*Type* string

*Options* 1.17

**build.apt_packages**

List of APT packages to install. Our build servers run Ubuntu 18.04, with the default set of package repositories installed. We don’t currently support PPA’s or other custom repositories.

*Type* list

*Default* []
Note: When possible avoid installing Python packages using apt (python3-numpy for example), use pip or Conda instead.

sphinx

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

Note: If you want to pin Sphinx to a specific version, use a requirements.txt or environment.yml file (see Requirements file and conda.environment). If you are using a metadata file to describe code dependencies like setup.py, pyproject.toml, or similar, you can use the extra_requirements option (see Packages). This also allows you to override the default pinning done by Read the Docs if your project was created before October 2020.

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 (`-W` and `--keep-going` options). This means the build fails if there is a warning and exits with exit status 1.

  * **Type**: bool
  * **Default**: false

**mkdocs**

Configuration for MkDocs documentation.

```yaml
version: 2

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

**Note:** If you want to pin MkDocs to a specific version, use a `requirements.txt` or `environment.yml` file (see *Requirements file* and *conda.environment*). If you are using a metadata file to describe code dependencies like `setup.py`, `pyproject.toml`, or similar, you can use the `extra_requirements` option (see *Packages*). This also allows you to override the default pinning done by Read the Docs if your project was created before March 2021.

**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.

**Warning:** You can’t use `include` and `exclude` settings for submodules at the same time.

```yaml
version: 2
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.

```yaml
version: 2
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: []`.

```
version: 2
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.

**search**

Settings for more control over *Server Side Search*.

```
version: 2
search:
  ranking:
    api/v1/*: -1
    api/v2/*: 4
  ignore:
    - 404.html
```

**search.ranking**

Set a custom search rank over pages matching a pattern.

  - **Type**: map of patterns to ranks
  - **Default**: {}

Patterns are matched against the final html pages produced by the build (you should try to match `index.html`, not `docs/index.rst`). Patterns can include some special characters:

- `*` matches everything
- `?` matches any single character
• [seq] matches any character in seq

The rank can be an integer number between -10 and 10 (inclusive). Pages with a rank closer to -10 will appear further down the list of results, and pages with a rank closer to 10 will appear higher in the list of results. Note that 0 means normal rank, not no rank.

If you are looking to completely ignore a page, check `search.ignore`.

```yaml
version: 2

search:
  ranking:
    # Match a single file
tutorial.html: 2

    # Match all files under the api/v1 directory
api/v1/*/: -5

    # Match all files that end with tutorial.html
'*/tutorial.html': 3
```

Note: The final rank will be the last pattern to match the page.

Tip: Is better to decrease the rank of pages you want to deprecate, rather than increasing the rank of the other pages.

```
search.ignore

Don't index files matching a pattern. This is, you won’t see search results from these files.

Type  list of patterns
Default ['search.html', 'search/index.html', '404.html', '404/index.html']

Patterns are matched against the final html pages produced by the build (you should try to match index.html, not docs/index.rst). Patterns can include some special characters:

• * matches everything
• ? matches any single character
• [seq] matches any character in seq
```

```yaml
version: 2

search:
  ignore:
    # Ignore a single file
    - 404.html

    # Ignore all files under the search/ directory
    - search/*
```
# Ignore all files that end with ref.html
- '*/ref.html'

version: 2

search:
ignore:
  # Custom files to ignore
  - file.html
  - api/v1/*
  
  # Defaults
  - search.html
  - search/index.html
  - 404.html
  - 404/index.html'

Note: Since Read the Docs fallbacks to the original search engine when no results are found, you may still see search results from ignored pages.

Schema

You can see the complete schema here.

Legacy build specification

The legacy build specification used a different set of Docker images, and only allowed you to specify the Python version. It remains supported for backwards compatibility reasons. Check out the build above for an alternative method that is more flexible.

version: 2

build:
  image: latest
  apt_packages:
    - libclang
    - cmake

python:
  version: "3.7"

The legacy build specification also supports the apt_packages key described above.

Warning: When using the new specifcation, the build.os and python.version options cannot be used. Doing so will error the build.
build (legacy)

build.image (legacy)

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

python.version (legacy)

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

Type  string
Default  3

Note: Make sure to use quotes (") to make it a string. We previously supported using numbers here, but that approach is deprecated.

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.

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 has been replaced by build.os. See build.os. Alternatively, you can use the legacy build.image that now has only two options: latest (default) and stable.
- 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
- search

**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

**2.1.2 Configuration File V1 (Deprecated)**

Read the Docs 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:
Supported settings

**Warning:** When using a v1 configuration file, the local settings from the web interface are overridden.

### version

- Default: 1

```yaml
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.

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

```yaml
# 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.

<table>
<thead>
<tr>
<th>Warning:</th>
<th>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 <strong>build.image</strong> for more information on supported Python versions.</th>
</tr>
</thead>
</table>

```yaml
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 **Build Process** for more details.

```yaml
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.

```yaml
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
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:

```python
from setuptools import setup

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

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

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

Behind the scene the following Pip command will be run:

```
$ pip install .[tests,docs]
```
2.2 VCS Integrations

Read the Docs provides integrations with several VCS providers to detect changes to your documentation and versions, mainly using webhooks. Integrations are configured with your repository provider, such as GitHub, Bitbucket or GitLab, and with each change to your repository, Read the Docs is notified. When we receive an integration 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.

You’ll find a list of configured 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, including the Payload URL needed by the repository provider such as GitHub, GitLab, or Bitbucket.

2.2.1 Integration Creation

If you have connected your Read the Docs account to GitHub, Bitbucket, or GitLab, an integration 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 an integration for your project.

To manually set up an integration, 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 integration 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 integration automatically. If something fails, you can still setup the integration 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, Pull Requests 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

**Gitea**

These instructions apply to any Gitea instance.

| Warning: This isn’t officially supported, but using the “GitHub webhook” is an effective workaround, because Gitea uses the same payload as GitHub. The generic webhook is not compatible with Gitea. See issue #8364 for more details. Official support may be implemented in the future. |

On Read the Docs:
- Manually create a “GitHub webhook” integration (this will show a warning about the webhook not being correctly set up, that will go away when the webhook is configured in Gitea)

On your Gitea instance:
- Go to the Settings > Webhooks page for your project on your Gitea instance
- Create a new webhook of type “Gitea”
- For URL, use the URL of the integration on Read the Docs, found on the Admin > Integrations page
- Leave the default HTTP Method as POST
- For Content type, both application/json and application/x-www-form-urlencoded work
- Leave the Secret field blank
- Select Choose events, and mark Branch or tag creation, Branch or tag deletion and Push events
- Ensure Active is enabled; it is by default
- Finish by clicking Add Webhook
- Test the webhook with Delivery test

Finally, on Read the Docs, check that the warnings have disappeared and the delivery test triggered a build.
2.2.2 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:

```bash
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.

2.2.3 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.

2.2.4 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.
2.2.5 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.

2.2.6 Troubleshooting

Webhook activation failed. Make sure you have the necessary permissions

If you find this error, make sure your user has permissions over the repository. In case of GitHub, check that you have granted access to the Read the Docs OAuth App to your organization.

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.

2.3 Custom Domains and White Labeling

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, see Custom domain support.

2.3.1 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 your documentation. A good example is https://pip.readthedocs.io For Read the Docs for Business the subdomain looks like <slug>.readthedocs-hosted.com.

2.3.2 Custom domain support

You can also host your documentation from your own domain.

Note: We don’t currently support pointing subdomains or root domains to a project using A records. DNS A records require a static IP address and our IPs may change without notice.

Read the Docs Community
Read the Docs for Business

In order to setup your custom domain, follow these steps:

1. For a subdomain like docs.example.com, add a CNAME record in your DNS that points the domain to readthedocs.io. For a root domain like example.com use an ANAME or ALIAS record pointing to readthedocs.io.
2. Go the Admin tab of your project
3. Click on Domains
4. Enter your domain and click on Add

By default, we provide a validated SSL certificate for the domain. This service is generously provided by Cloudflare. The SSL certificate issuance can take about one hour, you can see the status of the certificate on the domain page in your project.

As an example, fabric’s DNS record looks like this:

$ dig CNAME +short docs.fabfile.org
     readthedocs.io.

In order to setup your custom domain, follow these steps:

1. Go the Admin tab of your project
2. Click on Domains
3. Enter your domain and click on Add
4. Follow the steps shown on the domain page. This will require adding 2 DNS records, one pointing your custom domain to our servers, and another allowing us to provision an SSL certificate.

By default, we provide a validated SSL certificate for the domain. The SSL certificate issuance can take a few days, you can see the status of the certificate on the domain page in your project.

---

**Note:** Some older setups configured a CNAME record pointing to `<organization-slug>.users.readthedocs.com`. These domains will continue to resolve.

---

**Note:** You will need to keep the extra DNS records after the setup is complete. If you remove the verification record, the certificate won’t renew automatically, and if you remove the CNAME pointing to us, your domain won’t serve the documentation.

---

**Strict Transport Security**

By default, we do not return a Strict Transport Security header (HSTS) for user custom domains. This is a conscious decision as it can be misconfigured in a not easily reversible way. For both Read the Docs Community and Read the Docs for Business, HSTS for custom domains can be set upon request.

We always return the HSTS header with a max-age of at least one year for our own domains including `*.readthedocs.io`, `*.readthedocs-hosted.com`, `readthedocs.org` and `readthedocs.com`.

---

**Troubleshooting**

**SSL certificate issue delays**

The status of your domain validation and certificate can always be seen on the details page for your domain under Admin > Domains > YOURDOMAIN.TLD (details).

- For Read the Docs Community, domains are usually validated and a certificate issued within minutes. However, if you setup the domain in Read the Docs without provisioning the necessary DNS changes and then update DNS hours or days later, this can cause a delay in validating because there is an exponential back-off in validation. Loading the domain details in the Read the Docs dashboard and saving the domain again will force a revalidation.
• For Read the Docs for Business, domains can take up to a couple days to validate and issue a certificate. To avoid any downtime in moving a domain from somewhere else to Read the Docs, it is possible to validate the domain and provision the certificate before pointing your domain to Read the Docs.

Certificate authority authorization

Certificate authority authorization (CAA) is a security feature that allows domain owners to limit which certificate authorities (CAs) can issue certificates for a domain. This is done by setting CAA DNS records for your domain. CAA records are typically on the root domain, not subdomains since you can’t have a CNAME and CAA record for the same subdomain. Here’s an example for palletsprojects.com:

$ dig CAA +short palletsprojects.com
0 issue "digicert.com"
0 issue "comodoca.com"
0 issue "letsencrypt.org"

If there are CAA records for your domain that do not allow the certificate authorities that Read the Docs uses, you may see an error message like pending_validation: caa_error: YOURDOMAIN.TLD in the Read the Docs dashboard for your domain. You will need to update your CAA records to allow us to issue the certificate.

• For Read the Docs Community, we use Cloudflare which uses Digicert as a CA. See the Cloudflare CAA FAQ for details.

• For Read the Docs for Business, we use AWS Certificate Manager as a CA. See the Amazon CAA guide for details.

Note: If your custom domain was previously used in GitBook, contact GitBook support (via live chat in their website) to remove the domain name from their DNS Zone in order for your domain name to work with Read the Docs, else it will always redirect to GitBook.

2.3.3 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 are using *Sphinx*, Read the Docs will set the value of the html_baseurl setting (if isn’t already set) to your canonical domain.

If you are using *MkDocs*, you can use the site_url setting.

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/" />
```

2.4 Versioned Documentation

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

If your project has any tags or branches with a name following semantic versioning, we also create a *stable* version, tracking your most recent release. If you want a custom stable version, create either a tag or branch in your project with that name.

When you have *VCS Integrations* configured for your repository, we will automatically build each version when you push a commit.

2.4.1 How we envision versions working

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

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

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

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

If you have documentation changes on a *long-lived branch*, you can build those too. This will allow you to see how the new docs will be built in this branch of the code. Generally you won’t have more than 1 active branch over a long period of time. The main exception here would be release branches, which are branches that are maintained over time for a specific release number.
2.4.2 Version States

States define the visibility of a version across the site. You can change the states of a version from the Versions tab of your project.

**Active**

- **Active**
  - Docs for this version are visible
  - Builds can be triggered for this version

- **Inactive**
  - Docs for this version aren’t visible
  - Builds can’t be triggered for this version

When you deactivate a version, its docs are removed.

**Hidden**

- **Not hidden and Active**
  - This version is listed on the version (flyout) menu on the docs site
  - This version is shown in search results on the docs site

- **Hidden and Active**
  - This version isn’t listed on the version (flyout) menu on the docs site
  - This version isn’t shown in search results from another version on the docs site (like on search results from a superproject)

Hiding a version doesn’t make it private, any user with a link to its docs would be able to see it. This is useful when:

- You no longer support a version, but you don’t want to remove its docs.
- You have a work in progress version and don’t want to publish its docs just yet.

_Note:_ Active versions that are hidden will be listed as `Disallow: /path/to/version/` in the default `robots.txt` file created by Read the Docs.

2.4.3 Privacy levels

_Note:_ Privacy levels are only supported on Read the Docs for Business.
Public

It means that everything is available to be seen by everyone.

Private

Private versions 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. If you want to share your docs temporarily, see Sharing.

In addition, if you want other users to view the build page of your public versions, you’ll need to set the project level of your project to public.

Logging out

When you log in to a documentation site, you will be logged in until close your browser. To log out, click on the Log out link in your documentation’s flyout menu. This is usually located in the bottom right or bottom left, depending on the theme design. This will log you out from the current domain, but not end any other session that you have active.
2.4.4 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.

Version Control Support Matrix

<table>
<thead>
<tr>
<th></th>
<th>git</th>
<th>hg</th>
<th>bzt</th>
<th>svn</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tags</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Branches</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Default</td>
<td>master</td>
<td>default</td>
<td>trunk</td>
<td></td>
</tr>
</tbody>
</table>

2.4.5 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).

Note: The banner will be injected in an HTML element with the main role or in the main tag. For example:

```html
<div role="main">
  <!-- The banner would be injected here -->
  ...
</div>

<main>
  <!-- The banner would be injected here -->
  ...
</main>
```

2.4.6 Redirects on root URLs

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

2.5 Downloadable Documentation

Read the Docs supports building multiple formats for Sphinx-based projects:

- PDF
- ePub
- Zipped HTML

This means that every commit that you push will automatically update your PDFs as well as your HTML.

This is enabled by the formats key in our config file. A simple example is here:
# Build PDF & ePub formats

- epub
- pdf

If you want to see an example, you can download the Read the Docs documentation in the following formats:

- PDF
- ePub
- Zipped HTML

## 2.5.1 Use cases

This functionality is great for anyone who needs documentation when they aren’t connected to the internet. Users who are about to get on a plane can grab a PDF and have the entire doc set ready for their trip.

The other value is having the entire docset in a single file. You can send a user an email with a single PDF or ePub and they’ll have all the docs in one place.

## 2.5.2 Deleting downloadable content

The entries in the Downloads section of your project dashboard reflect the formats specified in your config file for each active version.

This means that if you wish to remove downloadable content for a given version, you can do so by removing the matching `formats` key from your config file.

## 2.6 Documentation Hosting Features

The main way that users interact with your documentation is via the hosted HTML that we serve. We support a number of important features that you would expect for a documentation host.
2.6.1 Content Delivery Network (CDN)

A CDN is used for making documentation pages faster for your users. This is done by caching the documentation page content in multiple data centers around the world, and then serving docs from the data center closest to the user.

We support CDN’s on both of our sites, as we talk about below.

Read the Docs Community

On Read the Docs Community, we are able to provide a CDN to all the projects that we host. This service is graciously sponsored by CloudFlare.

We bust the cache on the CDN when the following actions happen:

• Your Project is saved
• Your Domain is saved
• A new version is built

Read the Docs for Business

On Read the Docs for Business, we offer a CDN as part of our Enterprise plan. Please contact support@readthedocs.com to discuss how we can enable this for you.

2.6.2 Sitemaps

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 semantic versioning.

2.6.3 Custom Not Found (404) Pages

If you want your project to use a custom page for not found pages instead of the “Maze Found” default, 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 corresponding to the current version and uses it if it exists. Otherwise, it tries to fall back to the 404.html page corresponding to the default version of the project.

We recommend the sphinx-notfound-page extension, which Read the Docs maintains. It automatically creates a 404.html page for your documentation, matching the theme of your project. See its documentation for how to install and customize it.
2.6.4 Custom robots.txt Pages

robots.txt files allow you to customize how your documentation is indexed in search engines. We automatically generate one for you, which automatically hides versions which are set to Hidden.

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.

2.7 Server Side Search

Read the Docs provides full-text search across all of the pages of all projects, this is powered by Elasticsearch. You can search all projects at https://readthedocs.org/search/, or search only on your project from the Search tab of your project.

We override the default search engine of your project with our search engine to provide you better results within your project. We fallback to the built-in search engine from your project if our search engine doesn’t return any results, just in case we missed something.

2.7.1 Search features

We offer a number of benefits compared to other documentation hosts:

**Search across subprojects** Subprojects allow you to host multiple discrete projects on a single domain. Every subproject hosted on that same domain is included in the search results of the main project.

**Search results land on the exact content you were looking for** We index every heading in the document, allowing you to get search results exactly to the content that you are searching for. Try this out by searching for “full-text search”.

**Full control over which results should be listed first** Set a custom rank per page, allowing you to deprecate content, and always show relevant content to your users first. See search.ranking.

**Search across projects you have access to (Read the Docs for Business)** This allows you to search across all the projects you access to in your Dashboard. Don’t remember where you found that document the other day? No problem, you can search across them all.

**Special query syntax for more specific results.** We support a full range of search queries. You can see some examples in our Search query syntax guide.

**Configurable.** Tweak search results according to your needs using a configuration file.
2.7.2 Search Analytics

Know what your users are looking for in your docs. To see a list of the top queries and an overview from the last month, go to the Admin tab of your project, and then click on Search Analytics.

![Search Analytics](image)

2.7.3 API

If you are using Read the Docs for Business you will need to replace https://readthedocs.org/ with https://readthedocs.com/ in all the URLs used in the following examples. Check Authentication and authorization if you are using private versions.

**Warning:** This API isn’t stable yet, some small things may change in the future.

**GET /api/v2/search/**

Return a list of search results for a project, including results from its Subprojects. Results are divided into sections with highlights of the matching term.

**Query Parameters**

- **q** – Search query
- **project** – Project slug
• **version** – Version slug
• **page** – Jump to a specific page
• **page_size** – Limits the results per page, default is 50

**Response JSON Object**
• **type** *(string)* – The type of the result, currently page is the only type.
• **project** *(string)* – The project slug
• **project_alias** *(string)* – Alias of the project if it’s a subproject.
• **version** *(string)* – The version slug
• **title** *(string)* – The title of the page
• **domain** *(string)* – Canonical domain of the resulting page
• **path** *(string)* – Path to the resulting page
• **highlights** *(object)* – An object containing a list of substrings with matching terms. Note that the text is HTML escaped with the matching terms inside a `<span>` tag.
• **blocks** *(object)* – A list of block objects containing search results from the page. Currently, there are two types of blocks:
  – section: A page section with a linkable anchor (id attribute).
  – domain: A Sphinx domain with a linkable anchor (id attribute).

**Example request**

**Bash**

```
$ curl "https://readthedocs.org/api/v2/search/?project=docs&version=latest&q=server side search"
```

**Python**

```python
import requests

URL = 'https://readthedocs.org/api/v2/search/'
params = {
    'q': 'server side search',
    'project': 'docs',
    'version': 'latest',
}
response = requests.get(URL, params=params)
print(response.json())
```

**Example response**:

```
{
    "count": 41,
    "next": "https://readthedocs.org/api/v2/search/?page=2&project=read-the-docs&
    q=server+side+search&version=latest",
    "previous": null,
    "results": [
        {
            "type": "page",
```
Read the Docs provides full-text search across all of the pages of all projects, this is powered by Elasticsearch.

You can search all projects at https://readthedocs.org/search/

Retrieve search results for docs

Retrieve search results for docs
Authentication and authorization

If you are using private versions, users will only be allowed to search projects they have permissions over. Authentication and authorization is done using the current session, or any of the valid sharing methods.

To be able to use the user’s current session you need to use the API from the domain where your docs are being served (<you-docs-domain>/_/api/v2/search/). This is https://docs.readthedocs-hosted.com/_/api/v2/search/ for the https://docs.readthedocs-hosted.com/ project, for example.

2.8 Traffic Analytics

Traffic Analytics lets you see which documents your users are reading. This allows you to understand how your documentation is being used, so you can focus on expanding and updating parts people are reading most.

To see a list of the top pages from the last month, go to the Admin tab of your project, and then click on Traffic Analytics.

You can also access to analytics data from search results.

Note: The amount of analytics data stored for download depends which site you’re using:

- On the Community site, the last 90 days are stored.
- On the Commercial one, it goes from 30 to infinite storage (check out the pricing page).
2.8.1 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-123456789-1)

![Analytics code field]

Fig. 3: Options to manage Google Analytics

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.

Disabling Google Analytics on your project

Google Analytics can be completely disabled on your own projects. To disable Google Analytics:

• Going to Admin > Advanced Settings in your project.
• Check the box Disable Analytics.

Your documentation will need to be rebuilt for this change to take effect.

2.9 Preview Documentation from Pull Requests

Read the Docs allows you to build and preview your documentation from pull requests. To enable this feature:

1. Go to your project dashboard
2. Go to Admin > Advanced settings
3. Enable the Build pull requests for this project option
4. Click on Save
2.9.1 Features

- **Build on Pull Request Events**: We create and build a new version when a pull request is open, and when a new commit has been pushed.

- **Build Status Report**: When a build is triggered, a build pending notification is sent with a link to the build log. When the build finishes we send a success notification with the link to the preview or a failure notification with a link to the build log.

- **Warning Banner**: A warning banner is shown at the top of the documentation to let users know that this isn’t the main documentation for the project.

  **Note**: Warning banners are available only for *Sphinx projects*.

Fig. 4: GitHub build status reporting

2.9.2 Privacy levels

  **Note**: Privacy levels are only supported on *Read the Docs for Business*.

By default all docs built from pull requests are private. To change their privacy level:

1. Go to your project dashboard
2. Go to *Admin > Advanced settings*
3. Select your option in *Privacy level of builds from pull requests*
4. Click on *Save*

Privacy levels work the same way as for *normal versions*.

2.9.3 Limitations

- Builds from pull requests have the same memory and time limitations as *regular builds*.

- Only available for GitHub and GitLab.

- Additional formats like PDF and Epub aren’t built to produce results quicker.

- Searches will default to the default experience for your tool. This is a feature we plan to add, but don’t want to overwhelm our search indexes used in production.

- The built documentation is kept for 90 days after the pull request has been closed or merged.
2.9.4 Troubleshooting

1. **Pull Requests builds are not triggered.** We only support GitHub and GitLab currently. You need to make sure that your Read the Docs account is connected with that provider. You can check this by going to your Username dropdown > Settings > Connected Services.

2. **Build status is not being reported to your VCS provider.** You need to make sure that you have granted access to the Read the Docs OAuth App to your personal or organization GitHub account. Learn more about this in our GitHub permission troubleshooting section.

   Also make sure your webhook integration is properly setup to handle events related to pull requests. You can setup or re-sync the integration from your projects admin dashboard. Learn more about setting up integrations from our integrations documentation.

2.10 Build Notifications and Webhooks

---

**Note:** Currently we don’t send notifications or trigger webhooks on builds from pull requests.

---

2.10.1 Email notifications

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!

2.10.2 Build Status Webhooks

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 > Webhooks in your project.
- Fill in the URL field and select what events will trigger the webhook
- Modify the payload or leave the default (see below)
- Click on Save

Every time one of the checked events triggers, Read the Docs will send a POST request to your webhook URL. The default payload will look like this:

```json
{
    "event": "build:triggered",
    "name": "docs",
    "slug": "docs",
}
```

(continues on next page)
Webhooks

We'll send a POST request to the URL with the JSON payload below on the selected events.

URL:
https://hooks.slack.c

URL to send the webhook to

Events:
- build:triggered
- build:passed
- build:failed

Events to subscribe

Fig. 5: URL and events for a webhook

```
"version": "latest",
"commit": "2552bb609ca46865dc36401dee0b1865a0ae52d",
"build": "15173336",
"start_date": "2021-11-03T16:23:14",
"build_url": "https://readthedocs.org/projects/docs/builds/15173336/",
"docs_url": "https://docs.readthedocs.io/en/latest/"
```

When a webhook is sent, a new entry will be added to the “Recent Activity” table. By clicking on each individual entry, you will see the server response, the webhook request, and the payload.

**Custom payload examples**

You can customize the payload of the webhook to suit your needs, as long as it is valid JSON. Below you have a couple of examples, and in the following section you will find all the available variables.
Fig. 6: Activity of a webhook

JSON payload:

```
{
"event": "{{ event }}",
"name": "{{ project.name }}",
"slug": "{{ project.slug }}",
"version": "{{ version.slug }}",
"commit": "{{ build.commit }}",
"build": "{{ build.id }}",
"start_date": "{{ build.start_date }}",
"build_url": "{{ build.url }}",
"docs_url": "{{ build.docs_url }}"
}
```

JSON payload to send to the webhook. Check the docs for available substitutions.

Fig. 7: Custom payload
Slack

```
{
  "attachments": [
    {
      "color": "#db3238",
      "blocks": [
        {
          "type": "section",
          "text": {
            "type": "mrkdwn",
            "text": "*Read the Docs build failed*"
          }
        },
        {
          "type": "section",
          "fields": [
            {
              "type": "mrkdwn",
              "text": "*Project*: <{{ project.url }}|{{ project.name }}>"
            },
            {
              "type": "mrkdwn",
              "text": "*Version*: {{ version.name }} ({{ build.commit }})"
            },
            {
              "type": "mrkdwn",
              "text": "*Build*: <{{ build.url }}|{{ build.id }}>"
            }
          ]
        }
      ]
    }
  ]
}
```

More information on the Slack Incoming Webhooks documentation.

Discord

```
{
  "username": "Read the Docs",
  "content": "Read the Docs build failed",
  "embeds": [
    {
      "title": "Build logs",
      "url": "{{ build.url }}",
      "color": 15258703,
      "fields": [
        {
          "name": "*Project*",
          ...
```

(continues on next page)
More information on the Discord webhooks documentation.

Variable substitutions reference

{{ event }} Event that triggered the webhook, one of build:triggered, build:failed, or build:passed.

{{ build.id }} Build ID.

{{ build.commit }} Commit corresponding to the build, if present (otherwise '').

{{ build.url }} URL of the build, for example https://readthedocs.org/projects/docs/builds/15173336/.

{{ build.docs_url }} URL of the documentation corresponding to the build, for example https://docs.readthedocs.io/en/latest/.

{{ build.start_date }} Start date of the build (UTC, ISO format), for example 2021-11-03T16:23:14.

{{ organization.name }} Organization name (Commercial only).

{{ organization.slug }} Organization slug (Commercial only).

{{ project.slug }} Project slug.

{{ project.name }} Project name.

{{ project.url }} URL of the project dashboard.

{{ version.slug }} Version slug.

{{ version.name }} Version name.
Validating the payload

After you add a new webhook, Read the Docs will generate a secret key for it and uses it to generate a hash signature (HMAC-SHA256) for each payload that is included in the X-Hub-Signature header of the request.

![Secret: zb68RILVC2yn2dqKh4xzbUmcP79joy](image)

Secret used to sign the payload of the webhook

Fig. 8: Webhook secret

We highly recommend using this signature to verify that the webhook is coming from Read the Docs. To do so, you can add some custom code on your server, like this:

```python
import hashlib
import hmac
import os

def verify_signature(payload, request_headers):
    """
    Verify that the signature of payload is the same as the one coming from request_headers.
    """
    digest = hmac.new(
        key=os.environ['WEBHOOK_SECRET'].encode(),
        msg=payload.encode(),
        digestmod=hashlib.sha256,
    )
    expected_signature = digest.hexdigest()

    return hmac.compare_digest(
        request_headers['X-Hub-Signature'].encode(),
        expected_signature.encode(),
    )
```

Legacy webhooks

Webhooks created before the custom payloads functionality was added to Read the Docs send a payload with the following structure:

```
{
    "name": "Read the Docs",
    "slug": "rtd",
    "build": {
        "id": 6321373,
        "commit": "e8dd17a3f1627dd206d721e4be08ae6766fda40",
    }
}
```
To migrate to the new webhooks and keep a similar structure, you can use this payload:

```json
{
  "name": "{{ project.name }}",
  "slug": "{{ project.slug }}",
  "build": {
    "id": "{{ build.id }}",
    "commit": "{{ build.commit }}",
    "state": "{{ event }}",
    "date": "{{ build.start_date }}"
  }
}
```

2.11 Security Log

Security logs allow you to see what has happened recently in your organization or account. We store the IP address and the browser’s User-Agent on each event, so that you can confirm this access was from the intended person.

2.11.1 User security log

We store user security logs from the last 90 days, and track the following events:

- Authentication on the dashboard
- Authentication on documentation pages (Read the Docs for Business only)

Authentication failures and successes are both tracked.

To access your logs:

- Click on Username dropdown
- Click on Settings
- Click on Security Log

2.11.2 Organization security log

Note: This feature exists only on Read the Docs for Business.

We store logs according to your plan, check our pricing page for more details. We track the following events:

- Authentication on documentation pages from your organization
- User access to every documentation page from your organization (Enterprise plans only)
Authentication failures and successes are both tracked.

To access your organization logs:

- Click on *Organizations* from your user dropdown
- Click on your organization
- Click on *Settings*
- Click on *Security Log*

## 2.12 Connecting Your VCS 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 *VCS Integrations* 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 a social account, go to your *Username dropdown > Settings > 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.

### 2.12.1 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 *VCS Integrations* 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.

**GitHub permission troubleshooting**

**Repositories not in your list to import.**

Many organizations require approval for each OAuth application that is used, or you might have disabled it in the past for your personal account. This can happen at the personal or organization level, depending on where the project you are trying to access has permissions from.

Personal Account

Organization Account

You need to make sure that you have granted access to the Read the Docs OAuth App to your **personal GitHub account**. If you do not see Read the Docs in the OAuth App settings, you might need to disconnect and reconnect the GitHub service.

See also:

GitHub docs on requesting access to your personal OAuth for step-by-step instructions.

You need to make sure that you have granted access to the Read the Docs OAuth App to your **organization GitHub account**. If you don’t see “Read the Docs” listed, then you might need to connect GitHub to your social accounts as noted above.

See also:

GitHub doc on requesting access to your organization OAuth for step-by-step instructions.

**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
2.13 Build Process

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

Read the Docs Community

- 15 minutes build time
- 3GB of memory
- 2 concurrent builds

We can increase build limits on a per-project basis. Send 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.

- 30 minutes build time
- 7GB of memory
- Concurrent builds vary based on your pricing plan

If you are having trouble with your documentation builds, you can reach our support at support@readthedocs.com.

2.13.1 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, this is done inside a Docker container.

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 <format> . _build/<format>` We also create pdf’s and ePub’s automatically based on your project. For MkDocs, we run `mkdocs build`.

Once these files are built, they are deployed to our file hosting backend and will be visible to users.

An example in code:

```python
update_docs_from_vcs(version)
config = get_config(project)
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)
deploy_docs(version)
```
Note: Regardless of whether you build your docs with Sphinx or MkDocs, we recommend you *pinning the version* of Sphinx or Mkdocs you want us to use. Some examples of pinning versions might be `sphinx<2.0` or `mkdocs>=1.0`.

## 2.13.2 Build environment

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

<table>
<thead>
<tr>
<th>Environment variable</th>
<th>Description</th>
<th>Example value</th>
</tr>
</thead>
<tbody>
<tr>
<td>READTHEDOCS</td>
<td>Whether the build is running inside RTD</td>
<td>True</td>
</tr>
<tr>
<td>READTHEDOCS_VERSION</td>
<td>The RTD name of the version which is being built</td>
<td>latest</td>
</tr>
<tr>
<td>READTHEDOCS_PROJECT</td>
<td>The RTD slug of the project which is being built</td>
<td>my-example-project</td>
</tr>
<tr>
<td>READTHEDOCS_LANGUAGE</td>
<td>The RTD language slug of the project which is being built</td>
<td>en</td>
</tr>
</tbody>
</table>

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 *Environment Variables*.

### 2.13.3 Docker images

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*.

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.

### 2.13.4 Default versions of dependencies

Read the Docs supports two tools to build your documentation: *Sphinx* and *MkDocs*. In order to provide *several features*, Read the Docs injects or modifies some content while building your docs.

In particular, if you don’t specify the dependencies of your project, we install some of them on your behalf. In the past we used to pin these dependencies to a specific version and update them after some time, but doing so would break some builds and make it more difficult for new projects to use new versions. For this reason, we are now installing their latest version by default.
Note: In order to keep your builds reproducible, it’s highly recommended declaring its dependencies and versions explicitly. See *Reproducible Builds*.

**External dependencies**

**Python**

These are the dependencies that are installed by default when using a Python environment:

- **Sphinx**: Projects created before Oct 20, 2020 use 1.8.x. New projects use the latest version.
- **Mkdocs**: Projects created before April 3, 2019 (April 23, 2019 for *Read the Docs for Business*) use 0.17.3. Projects created before March 9, 2021 use 1.0.4. New projects use the latest version.
- **sphinx-rtd-theme**: Projects created before October 20, 2020 (January 21, 2021 for *Read the Docs for Business*) use 0.4.3. New projects use the latest version.
- **pip**: Latest version by default.
- **setuptools**: 58.2.0 or older.
- **mock**: 1.0.1 (could be removed in the future).
- **pillow**: 5.4.1 (could be removed in the future).
- **alabaster**: 0.7.x (could be removed in the future).
- **commonmark**: 0.8.1 (could be removed in the future).
- **recommonmark**: 0.5.0 (could be removed in the future).

**Conda**

These are the dependencies that are installed by default when using a Conda environment:

- **Conda**: Miniconda2 4.6.14 (could be updated in the future to use the latest version by default).
- **Mkdocs**: Latest version by default installed via conda.
- **Sphinx**: Latest version by default installed via conda.
- **sphinx-rtd-theme**: Latest version by default installed via conda.
- **mock**: Latest version by default installed via pip (could be removed in the future).
- **pillow**: Latest version by default installed via pip (could be removed in the future).
- **recommonmark**: Latest version by default installed via conda (could be removed in the future).
Internal dependencies

Internal dependencies are needed to integrate your docs with Read the Docs. We guarantee that these dependencies will work with all current supported versions of our tools, you don’t need to specify them in your requirements.

- readthedocs-sphinx-ext

2.14 Environment Variables

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

For example, 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.

To define an environment variable, you need to

1. Go to your project’s Admin > Environment Variables
2. Click on Add Environment Variable
3. Fill the Name and Value (your secret)
4. Check the Public option if you want to expose this environment variable to builds from pull requests.

**Warning:** If you mark this option, any user that can create a pull request on your repository will be able to see the value of this environment variable.

5. Click on Save

**Note:** Once you create an environment variable, you won’t be able to see its value anymore.

After adding an environment variable, you can read it from your build process, for example in your Sphinx’s conf.py file:

```python
# 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),
)
```
2.15 Badges

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

2.15.1 Status Badges

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

Here are a few examples:

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

2.15.2 Style

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

<table>
<thead>
<tr>
<th>STYLE</th>
<th>BADGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>flat</td>
<td></td>
</tr>
<tr>
<td>flat-square</td>
<td></td>
</tr>
<tr>
<td>for-the-badge</td>
<td></td>
</tr>
<tr>
<td>plastic</td>
<td></td>
</tr>
<tr>
<td>social</td>
<td></td>
</tr>
</tbody>
</table>

2.15.3 Project Pages

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

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

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

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

https://readthedocs.org/projects/pip/badge/
2.16 Site Support

2.16.1 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?”

You might also find the answers you are looking for in our documentation guides. These provide step-by-step solutions to common user requirements.

2.16.2 User Support

If you need a specific request for your project or account, like more resources, change of the project’s slug or username.

Read the Docs Community
Read the Docs for Business

Please fill out the form at https://readthedocs.org/support/, and we will reply as soon as possible.

Please fill out the form at https://readthedocs.com/support/, and we will reply within 1 business day for most plans.

2.16.3 Bug Reports

If you have an issue with the actual functioning of the site, you can file bug reports on our GitHub issue tracker. You can also contribute to Read the Docs, as the code is open source.

2.16.4 Priority Support

We offer priority support with Read the Docs for Business and we have a dedicated team that responds to support requests during business hours.

2.17 Frequently Asked Questions

2.17.1 My project isn’t building correctly

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, see our section on My documentation requires additional dependencies.

If you are still seeing errors because of C library dependencies, please see I get import errors on libraries that depend on C modules.
2.17.2 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).

2.17.3 My documentation requires additional dependencies

For most Python dependencies, you can specify a requirements file which details your dependencies. See our guide on Using a configuration file. You can also set your project documentation to install your project itself as a dependency.

Your build may depend on extensions that require additional system packages to be installed. If you are using a Configuration File you can add libraries with apt to the Ubuntu-based builder. If your project or its dependencies rely on C libraries that cannot be installed this way, see I get import errors on libraries that depend on C modules.

2.17.4 My project requires some additional settings

If this is just a dependency issue, see My documentation requires additional dependencies.

Read the Docs offers some settings which can be used for a variety of purposes. 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.

2.17.5 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 the build system does not have the dependencies for building your project, such as C libraries needed by some Python packages (e.g. libevent or mysql). For libraries that cannot be installed via apt in the builder there is another way to successfully build the documentation despite missing dependencies.

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.
2.17.6 How do I change behavior when building with 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:

```python
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:

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

2.17.7 Client Error 401 when building documentation

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

```
```

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'
```

2.17.8 Deleting a stale or broken build environment

See Wiping a Build Environment.

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

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

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

```
```

This also works the same for custom domains:

```
```

You can add subprojects in the project admin dashboard.
For details on custom domains, see our documentation on *Custom Domains and White Labeling*.

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

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

You can specify an exact path to your documentation using a Read the Docs [Configuration File](https://docs.readthedocs.io/en/latest/rtd-configuration.html).

### 2.17.11 I want to use the Blue/Default Sphinx theme

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

### 2.17.12 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.

### 2.17.13 Image scaling doesn’t work in my documentation

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

### 2.17.14 I want comments in my docs

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

### 2.17.15 How do I support multiple languages of documentation?

See the section on *Localization of Documentation*.

### 2.17.16 Does Read The Docs work well with “legible” docstrings?

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

**Note:** To use these extensions you need to specify the dependencies on your project by following this [guide](https://docs.readthedocs.io/en/latest/requirements.html).
2.17.17 Can I document a Python package that is not at the root of my repository?

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

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

```
src/python/
```

Please note that the path must be relative to the working directory where `pip` is launched, rather than the directory where the requirements file is located. Therefore, even if you want to move the requirements file to a `requirements/` directory, the example path above would work.

You can customize the path to your requirements file and any other installed dependency using a Read the Docs Configuration File.

2.17.18 I need to install a package in a environment with pinned versions

To ensure proper installation of a Python package, the `pip install` method will automatically upgrade every dependency to its most recent version in case they aren’t pinned by the package definition. If instead you’d like to pin your dependencies outside the package, you can add this line to your requirements or environment file (if you are using Conda).

In your `requirements.txt` file:

```
# path to the directory containing setup.py relative to the project root
-e .
```

In your Conda environment file (`environment.yml`):

```
# path to the directory containing setup.py relative to the environment file
-e ..
```

2.17.19 Can I use Anaconda Project and `anaconda-project.yml`?

Yes. With `anaconda-project>=0.8.4` you can use the Anaconda Project configuration file `anaconda-project.yaml` (or `anaconda-project.yml`) directly in place of a Conda environment file by using `dependencies:` as an alias for `packages:`.

I.e., your `anaconda-project.yaml` file can be used as a `conda.environment` config in the `.readthedocs.yaml` config file if it contains:

```
dependencies:
  - python=3.9
  - scipy
  ...
```
2.17.20 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). We have documented how to set this up in our Custom robots.txt Pages docs.

2.17.21 Can I remove advertising from my documentation?

See Opting out of advertising.

2.17.22 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.

2.17.23 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.

2.17.24 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

We also keep an up-to-date changelog.

2.17.25 How can I deploy Jupyter Book projects on Read the Docs?

According to its own documentation,

Jupyter Book is an open source project for building beautiful, publication-quality books and documents from computational material.

Even though Jupyter Book leverages Sphinx “for almost everything that it does”, it purposely hides Sphinx conf.py files from the user, and instead generates them on the fly from its declarative _config.yml. As a result, you need to follow some extra steps to make Jupyter Book work on Read the Docs.

As described in the official documentation, you can manually convert your Jupyter Book project to Sphinx with the following command:

```
$ jupyter-book config sphinx path/to/book
```
and then commit the resulting `conf.py` to git. Alternatively, you can set up some automation that does it for every change, for example using pre-commit.
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.

- **For documentation authors**: Cross-referencing with Sphinx | Link to Other Projects’ Documentation With Intersphinx | How to use Jupyter notebooks in Sphinx | More guides for authors
- **For project administrators**: Technical Documentation Search Engine Optimization (SEO) Guide | Manage Translations for Sphinx projects | Using advanced search features | Using Private Git Submodules | More guides for administrators
- **For developers and designers**: Installing Private Python Packages | Adding Custom CSS or JavaScript to Sphinx Documentation | Reproducible Builds | Embedding Content From Your Documentation | Conda Support | More guides for developers and designers

### 3.1 Guides for documentation authors

These guides offer some tips and tricks to author documentation with the tools supported on Read the Docs. Only reStructuredText or Markdown knowledge and minimal configuration tweaking are needed.

For an introduction to Sphinx and Mkdocs, have a look at our *Getting Started with Sphinx* and *Getting Started with MkDocs*.

#### 3.1.1 Cross-referencing with Sphinx

When writing documentation you often need to link to other pages of your documentation, other sections of the current page, or sections from other pages.

An easy way is just to use the raw URL that Sphinx generates for each page/section. This works, but it has some disadvantages:

- Links can change, so they are hard to maintain.
- Links can be verbose and hard to read, so it is unclear what page/section they are linking to.
- There is no easy way to link to specific sections like paragraphs, figures, or code blocks.
- URL links only work for the html version of your documentation.

Instead, Sphinx offers a powerful way to linking to the different elements of the document, called *cross-references*. Some advantages of using them:

- Use a human-readable name of your choice, instead of a URL.
- Portable between formats: html, PDF, ePub.
• Sphinx will warn you of invalid references.
• You can cross reference more than just pages and section headers.

This page describes some best-practices for cross-referencing with Sphinx with two markup options: reStructuredText and MyST (Markdown).

• If you are not familiar with reStructuredText, check reStructuredText Primer for a quick introduction.
• If you want to learn more about the MyST Markdown dialect, check out The MyST Syntax Guide.

---

### Table of contents

- Getting started
  - Explicit targets
  - Implicit targets
- Cross-referencing using roles
  - The ref role
  - The doc role
  - The numref role
- Automatically label sections
- Invalid targets
- Finding the reference name
- Cross-referencing targets in other documentation sites

### Getting started

### Explicit targets

Cross referencing in Sphinx uses two components, references and targets.

- references are pointers in your documentation to other parts of your documentation.
- targets are where the references can point to.

You can manually create a target in any location of your documentation, allowing you to reference it from other pages. These are called explicit targets.

For example, one way of creating an explicit target for a section is:

reStructuredText

MyST (Markdown)

```markdown
.. _My target:

Explicit targets

Reference `My target`
```

## Explicit targets

Reference [(My_target)](My_target).

Then the reference will be rendered as *My target*.

You can also add explicit targets before paragraphs (or any other part of a page).

Another example, add a target to a paragraph:

reStructuredText

MyST (Markdown)

```
.. _target to paragraph:

An easy way is just to use the final link of the page/section.
This works, but it has :ref:`some disadvantages <target to paragraph>`:
```

Then the reference will be rendered as: *some disadvantages*.

You can also create in-line targets within an element on your page, allowing you to, for example, reference text *within* a paragraph.

For example, an in-line target inside a paragraph:

reStructuredText

MyST (Markdown)

```
You can also create `_in-line targets` within an element on your page, allowing you to, for example, reference text *within* a paragraph.
```

Then you can reference it using `_in-line targets`_, that will be rendered as: *in-line targets*.

### Implicit targets

You may also reference some objects by name without explicitly giving them one by using *implicit targets*.

When you create a section, a footnote, or a citation, Sphinx will create a target with the title as the name:

reStructuredText

MyST (Markdown)

```
For example, to reference the previous section you can use `Explicit targets`_.
```

For example, to reference the previous section you can use [](#explicit-targets).

**Note:** This requires setting `myst_heading_anchors = 2` in your `conf.py`, see Auto-generated header anchors.
The reference will be rendered as: *Explicit targets*.

### Cross-referencing using roles

All targets seen so far can be referenced only from the same page. Sphinx provides some roles that allow you to reference any explicit target from any page.

**Note:** Since Sphinx will make all explicit targets available globally, all targets must be unique.

You can see the complete list of cross-referencing roles at [Cross-referencing syntax](#). Next, you will explore the most common ones.

#### The ref role

The `ref` role can be used to reference any explicit targets. For example:

```rst
MyST (Markdown)
```

- :ref:`my target`.
- :ref:`Target to paragraph <target to paragraph>`.
- :ref:`Target inside a paragraph <in-line targets>`.

- {ref}`my target`.
- {ref}`Target to paragraph <target_to_paragraph>`.

That will be rendered as:

- *Explicit targets.*
- *Target to paragraph.*
- *Target inside a paragraph.*

The `ref` role also allow us to reference code blocks:

```bash
.. _target to code:

.. code-block:: python

    # Add the extension
    extensions = [
        'sphinx.ext.autosectionlabel',
    ]

    # Make sure the target is unique
    autosectionlabel_prefix_document = True
```

We can reference it using :ref:`code <target to code>`, that will be rendered as: *code.*
The doc role

The doc role allows us to link to a page instead of just a section. The target name can be relative to the page where the role exists, or relative to your documentation’s root folder (in both cases, you should omit the extension).

For example, to link to a page in the same directory as this one you can use:

reStructuredText

MyST (Markdown)

- :doc:`intersphinx`
- :doc:`/guides/intersphinx`
- :doc:`Custom title </guides/intersphinx>`

That will be rendered as:

- *Link to Other Projects’ Documentation With Intersphinx*
- *Link to Other Projects’ Documentation With Intersphinx*
- *Custom title*

_tip_: Using paths relative to your documentation root is recommended, so you avoid changing the target name if the page is moved.

The numref role

The numref role is used to reference numbered elements of your documentation. For example, tables and images.

To activate numbered references, add this to your conf.py file:

```bash
# Enable numref
numfig = True
```

Next, ensure that an object you would like to reference has an explicit target.

For example, you can create a target for the next image:

reStructuredText

MyST (Markdown)

```markdown
.. _target to image:

.. figure:: /img/logo.png
   :alt: Logo
   :align: center
   :width: 240px

   Link me!
```

3.1. Guides for documentation authors
Automatically label sections

Manually adding an explicit target to each section and making sure it is unique is a big task! Fortunately, Sphinx includes an extension to help us with that problem, `autosectionlabel`.

To activate the `autosectionlabel` extension, add this to your `conf.py` file:

```python
# Add the extension
extensions = [
    'sphinx.ext.autosectionlabel',
]

# Make sure the target is unique
autosectionlabel_prefix_document = True
```

Sphinx will create explicit targets for all your sections, the name of target has the form `{path/to/page}:{title-of-section}`.

For example, you can reference the previous section using:

- reStructuredText
- MyST (Markdown)

- `{ref}`: `guides/cross-referencing-with-sphinx:explicit targets`.
- `{ref}`: `Custom title <guides/cross-referencing-with-sphinx:explicit targets>`.

That will be rendered as:

- Explicit targets.
- Custom title.

### Invalid targets

If you reference an invalid or undefined target Sphinx will warn us. You can use the `-W` option when building your docs to fail the build if there are any invalid references. On Read the Docs you can use the `sphinx.fail_on_warning` option.

### Finding the reference name

When you build your documentation, Sphinx will generate an inventory of all explicit and implicit links called `objects.inv`. You can list all of these targets to explore what is available for you to reference.

List all targets for built documentation with:

```bash
$ python -m sphinx.ext.intersphinx <link>
```

Where `<link>` is either a URL or a local path that points to your inventory file (usually in `_build/html/objects.inv`). For example, to see all targets from the Read the Docs documentation:

```bash
$ python -m sphinx.ext.intersphinx https://docs.readthedocs.io/en/stable/objects.inv
```

### Cross-referencing targets in other documentation sites

You can reference to docs outside your project too! See [Link to Other Projects' Documentation With Intersphinx](#).

#### 3.1.2 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.

```bash
$ 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.

```python
# 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.

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

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

reStructuredText

MyST (Markdown)

- :ref:`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:

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:

```python
# 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:

```python
# 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-fr': ('https://<token-for-fr-translation>:@readthedocs-docs.readthedocs-hosted.com/fr/latest', None),
    # Links to the private "apis" subproject of the "docs" project
}
```

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 Environment Variables 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:

```
3.1. Guides for documentation authors 97
```
# conf.py file

```python
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

```python
intersphinx_mapping = {
    # Links to a private project named "docs" using a local inventory file.
}
```

## 3.1.3 How to use Jupyter notebooks in Sphinx

Jupyter notebooks are a popular tool to describe computational narratives that mix code, prose, images, interactive components, and more. Embedding them in your Sphinx project allows using these rich documents as documentation, which can provide a great experience for tutorials, examples, and other types of technical content. There are a few extensions that allow integrating Jupyter and Sphinx, and this document will explain how to achieve some of the most commonly requested features.

### Including classic `.ipynb` notebooks in Sphinx documentation

There are two main extensions that add support Jupyter notebooks as source files in Sphinx: nbsphinx and MyST-NB. They have similar intent and basic functionality: both can read notebooks in .ipynb and additional formats supported by jupyter, and are configured in a similar way (see Existing relevant extensions for more background on their differences).

First of all, create a Jupyter notebook using the editor of your liking (for example, JupyterLab). For example, source/notebooks/Example 1.ipynb:

Next, you will need to enable one of the extensions, as follows:

nbsphinx

MyST-NB
Introduction

This is an example .ipynb notebook.

[1]:
```python
import sys
print(sys.version)
```

3.9.5 (default, May 19 2021, 11:32:47)
[GCC 9.3.0]

[2]:
```python
from IPython.display import Image
```

[3]: Image("http://sipi.usc.edu/database/preview/misc/4.2.03.png")

![Image of a mandrill](http://sipi.usc.edu/database/preview/misc/4.2.03.png)

Fig. 2: Example Jupyter notebook created on JupyterLab
Finally, you can include the notebook in any `toctree`. For example, add this to your root document:

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

    notebooks/Example 1
```

The notebook will render as any other HTML page in your documentation after doing `make html`.

Fig. 3: Example Jupyter notebook rendered on HTML by nbsphinx

To further customize the rendering process among other things, refer to the nbsphinx or MyST-NB documentation.
Rendering interactive widgets

Widgets are eventful python objects that have a representation in the browser and that you can use to build interactive GUIs for your notebooks. Basic widgets using ipywidgets include controls like sliders, textboxes, and buttons, and more complex widgets include interactive maps, like the ones provided by ipyleaflet.

You can embed these interactive widgets on HTML Sphinx documentation. For this to work, it’s necessary to save the widget state before generating the HTML documentation, otherwise the widget will appear as empty. Each editor has a different way of doing it:

- The classical Jupyter Notebook interface provides a “Save Notebook Widget State” action in the “Widgets” menu, as explained in the ipywidgets documentation. You need to click it before exporting your notebook to HTML.
- JupyterLab provides a “Save Widget State Automatically” option in the “Settings” menu. You need to leave it checked so that widget state is automatically saved.
- In Visual Studio Code it’s not possible to save the widget state at the time of writing (June 2021).

For example, if you create a notebook with a simple IntSlider widget from ipywidgets and save the widget state, the slider will render correctly in Sphinx.

To see more elaborate examples:

- ipyleaflet provides several widgets for interactive maps, and renders live versions of them in their documentation.
- PyVista is used for scientific 3D visualization with several interactive backends and examples in their documentation as well.

**Warning:** Although widgets themselves can be embedded in HTML, events require a backend (kernel) to execute. Therefore, @interact, .observe, and related functionalities relying on them will not work as expected.
Using notebooks in other formats

For example, this is how a simple notebook looks like in MyST Markdown format:

```
---
jupyter:
text_representation:
  extension: .md
  format_name: myst
  format_version: 0.13
  jupyter_version: 1.10.3
kernelspec:
display_name: Python 3
language: python
name: python3
---

# Plain-text notebook formats

This is an example of a Jupyter notebook stored in MyST Markdown format.

```
```
```
```
Notice that the Markdown format does not store the outputs of the computation. Sphinx will automatically execute notebooks without outputs, so in your HTML documentation they appear as complete.

**Creating galleries of examples using notebooks**

`nbsphinx` has support for creating thumbnail galleries from a list of Jupyter notebooks. This functionality relies on Sphinx-Gallery and extends it to work with Jupyter notebooks rather than Python scripts.

To use it, you will need to install both `nbsphinx` and Sphinx-Gallery, and modify your `conf.py` as follows:

```python
Listing 6: conf.py
extensions = [
    'nbsphinx',
    'sphinx_gallery.load_style',
]
```

After doing that, there are two ways to create the gallery:

- From a reStructuredText source file, using the .. `nbgallery::` directive, as showcased in the documentation.
- From a Jupyter notebook, adding a "nbsphinx-gallery" tag to the metadata of a cell. Each editor has a different way of modifying the cell metadata (see figure below).

For example, this reST markup would create a thumbnail gallery with generic images as thumbnails, thanks to the Sphinx-Gallery default style:

```
Thumbnails gallery
==================
.. nbgallery::
```

![Panel to modify cell metadata in JupyterLab](image)

Fig. 6: Panel to modify cell metadata in JupyterLab

For example, this reST markup would create a thumbnail gallery with generic images as thumbnails, thanks to the Sphinx-Gallery default style:
To see some examples of notebook galleries in the wild:

- poliastro offers tools for interactive Astrodynamics in Python, and features several examples and how-to guides using notebooks and displays them in an appealing thumbnail gallery. In addition, poliastro uses unpaired MyST Notebooks to reduce repository size and improve integration with git.

- Qiskit is a framework for quantum computing that leverages nbgallery for its tutorials and uses a highly customized style to match the branding of the website.

### Background

#### Existing relevant extensions

In the first part of this document we have seen that nbsphinx and MyST-NB are similar. However, there are some differences between them:

- nbsphinx uses pandoc to convert the Markdown from Jupyter notebooks to reStructuredText and then to docutils AST, whereas MyST-NB uses MyST-Parser to directly convert the Markdown text to docutils AST. Therefore, nbsphinx assumes pandoc flavored Markdown, whereas MyST-NB uses MyST flavored Markdown. Both Markdown flavors are mostly equal, but they have some differences.

- nbsphinx executes each notebook during the parsing phase, whereas MyST-NB can execute all notebooks up front and cache them with jupyter-cache. This can result in shorter build times when notebooks are modified if using MyST-NB.

- nbsphinx provides functionality to create thumbnail galleries, whereas MyST-NB does not have such functionality at the moment (see Creating galleries of examples using notebooks for more information about galleries).

- MyST-NB allows embedding Python objects coming from the notebook in the documentation (read their “glue” documentation for more information) and provides more sophisticated error reporting than the one nbsphinx has.

- The visual appearance of code cells and their outputs is slightly different: nbsphinx renders the cell numbers by default, whereas MyST-NB doesn’t.

Deciding which one to use depends on your use case. As general recommendations:
• If you want to use other notebook formats or generate a thumbnail gallery from your notebooks, nbsphinx is the right choice.

• If you want to leverage a more optimized execution workflow and a more streamlined parsing mechanism, as well as some of the unique MyST-NB functionalities (g1ue, better error reporting) you should use MyST-NB.

Alternative notebook formats

Jupyter notebooks in .ipynb format (as described in the nbformat documentation) are by far the most widely used for historical reasons.

However, to compensate some of the disadvantages of the .ipynb format (like cumbersome integration with version control systems), jupyter offers other formats based on plain text rather than JSON.

As a result, there are three modes of operation:

• Using classic .ipynb notebooks. It’s the most straightforward option, since all the tooling is prepared to work with them, and does not require additional pieces of software. It is therefore simpler to manage, since there are fewer moving parts. However, it requires some care when working with Version Control Systems (like git), by doing one of these things:
  – Clear outputs before commit. Minimizes conflicts, but might defeat the purpose of notebooks themselves, since the computation results are not stored.
  – Use tools like nbdime (open source) or ReviewNB (proprietary) to improve the review process.
  – Use a different collaboration workflow that doesn’t involve notebooks.

• Replace .ipynb notebooks with a text-based format. These formats behave better under version control and they can also be edited with normal text editors that do not support cell-based JSON notebooks. However, text-based formats do not store the outputs of the cells, and this might not be what you want.

• Pairing .ipynb notebooks with a text-based format, and putting the text-based file in version control, as suggested in the jupyter documentation. This solution has the best of both worlds. In some rare cases you might experience synchronization issues between both files.

These approaches are not mutually exclusive, nor you have to use a single format for all your notebooks. For the examples in this document, we have used the MyST Markdown format.

If you are using alternative formats for Jupyter notebooks, you can include them in your Sphinx documentation using either nbsphinx or MyST-NB (see Existing relevant extensions for more information about the differences between them).

3.2 Guides for project administrators

These guides cover common use cases relevant for managing documentation projects, using the Read the Docs web interface, and making changes to the configuration files.

For an introduction to Read the Docs, have a look at our Read the Docs tutorial.
3.2.1 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
```

(continues on next page)
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:

```rst
.. 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.
```

![Google search engine results showing a customized meta description](image)

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

3.2.2 Manage Translations for Sphinx projects

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:

```shell
$ 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:

```shell
$ 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:

```shell
$ 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 ""
msgid "Read the Docs hosts documentation for the open source community."
msgid "It supports :ref:`Sphinx <sphinx>` docs written with reStructuredText."
```

(continues on next page)
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:

```bash
$ 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:

```bash
$ 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:

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

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

```bash
$ 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:

```bash
$ 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.
3.2.3 Using advanced search features

Read the Docs uses *Server Side Search* to power our search. This guide explains how to add a “search as you type” feature to your documentation, and how to use advanced query syntax to get more accurate results.

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

### Table of contents

- Enable “search as you type” in your documentation
- Search query syntax
  - Exact phrase search
  - Exact phrase search with slop value
  - Prefix query
  - Fuzzy query
  - Build complex queries

### Enable “search as you type” in your documentation

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

To try this feature, 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=contributing)
- [https://docs.readthedocs.io/?rtd_search=api/v3/projects/](https://docs.readthedocs.io/?rtd_search=api/v3/projects/)

### Search query syntax

Read the Docs uses the Simple Query String feature from Elasticsearch. This means that as the search query becomes more complex, the results yielded become more specific.

### Exact phrase search

If a query is wrapped in " (double quotes), 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=%22custom%20css%22)
- [https://docs.readthedocs.io/?rtd_search=%22adding%20a%20subproject%22](https://docs.readthedocs.io/?rtd_search=%22adding%20a%20subproject%22)
- [https://docs.readthedocs.io/?rtd_search=%22when%20a%20404%20is%20returned%22](https://docs.readthedocs.io/?rtd_search=%22when%20a%20404%20is%20returned%22)
**Exact phrase search with slop value**

~N (tilde N) after a phrase signifies slop amount. It can be used to match words that 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**

* (asterisk) 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 the exact spelling of the keyword is unknown. 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=reedthedocs~2
- https://docs.readthedocs.io/?rtd_search=authentation~3
- https://docs.readthedocs.io/?rtd_search=configurtion~1

**Build complex queries**

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

For example:
- 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*
3.2.4 Hide a Version and Keep its Docs Online

If you manage a project with a lot of versions, the version (flyout) menu of your docs can be easily overwhelmed and hard to navigate.

![Overwhelmed flyout menu](image)

You can deactivate the version to remove its docs, but removing its docs isn’t always an option. To not list a version in the flyout menu while keeping its docs online, you can mark it as hidden. Go to the Versions tab of your project, click on Edit and mark the Hidden option.

Users that have a link to your old version will still be able to see your docs. And new users can see all your versions (including hidden versions) in the versions tab of your project at https://readthedocs.org/projects/<your-project>/versions/

Check the docs about versions’ states for more information.
3.2.5 Deprecating Content

When you deprecate a feature from your project, you may want to deprecate its docs as well, and stop your users from reading that content.

Deprecating content may sound as easy as delete it, but doing that will break existing links, and you don’t necessary want to make the content inaccessible. Here you’ll find some tips on how to use Read the Docs to deprecate your content progressively and in non harmful ways.

**Deprecating versions**

If you have multiple versions of your project, it makes sense to have its documentation versioned as well. For example, if you have the following versions and want to deprecate v1:

- https://project.readthedocs.io/en/v1/
- https://project.readthedocs.io/en/v2/

For cases like this you can hide a version. Hidden versions won’t be listed in the versions menu of your docs, and they will be listed in a robots.txt file to stop search engines of showing results for that version.

Users can still see all versions in the dashboard of your project. To hide a version go to your project and click on Versions > Edit, and mark the Hidden option. Check Version States for more information.

**Note:** If the versions of your project follow the semver convention, you can activate the Version warning option for your project. A banner with a warning and linking to the stable version will be shown on all versions that are lower than the stable one.

**Deprecating pages**

You may not always want to deprecate a version, but deprecate some pages. For example, if you have documentation about two APIs and you want to deprecate v1:

- https://project.readthedocs.io/en/latest/api/v2.html

A simple way is just adding a warning at the top of the page, this will warn users visiting that page, but it won’t stop users from being redirected to that page from search results. You can add an entry of that page in a custom robots.txt file to avoid search engines of showing those results. For example:

```plaintext
# robots.txt
User-agent: *
Disallow: /en/latest/api/v1.html # Deprecated API
```

But your users will still see search results from that page if they use the search from your docs. With Read the Docs you can set a custom rank per pages. For example:

```yaml
# .readthedocs.yaml
version: 2
```

(continues on next page)
This won’t hide results from that page, but it will give priority to results from other pages.

**Tip:** You can make use of Sphinx directives (like `warning`, `deprecated`, `versionchanged`) or MkDocs admonitions to warn your users about deprecated content.

### Moving and deleting pages

After you have deprecated a feature for a while, you may want to get rid of its documentation, that’s OK, you don’t have to maintain that content forever. But be aware that users may have links of that page saved, and it will be frustrating and confusing for them to get a 404.

To solve that problem you can create a redirect to a page with a similar feature/content, like redirecting to the docs of the v2 of your API when your users visit the deleted docs from v1, this is a page redirect from `/api/v1.html` to `/api/v2.html`. See [User-defined Redirects](#).

#### 3.2.6 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`:

```python
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:

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

And for **Japanese** projects:

```python
latex_engine = 'platex'
l 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.

### 3.2.7 Manually Importing Private Repositories

**Warning:** This guide is for users of *Read the Docs for Business*. If you are using GitHub, GitLab, or Bitbucket, we recommend connecting your account and importing your project from [https://readthedocs.com/dashboard/import](https://readthedocs.com/dashboard/import) instead of importing it manually.

If you are using an unsupported integration, or don’t want to connect your account, you’ll need to do some extra steps in order to have your project working.

1. Manually import your project using an SSH URL
2. Allow access to your project using an SSH key
3. Setup a webhook to build your documentation on every commit

#### Table of contents

- Importing your project
- Giving access to your project with an SSH key
  - Copy your project’s public key
  - Add the public key to your project
    - GitHub
    - GitLab
    - Bitbucket
    - Azure DevOps
    - Others
- Webhooks

#### Importing your project

2. Fill the Repository URL field with the SSH form of your repository’s URL, e.g `git@github.com:readthedocs/readthedocs.org.git`
3. Fill the other required fields
4. Click Next
Giving access to your project with an SSH key

After importing your project the build will fail, because Read the Docs doesn’t have access to clone your repository. To give access, you’ll need to add your project’s public SSH key to your VCS provider.

Copy your project’s public 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 public key to your project

GitHub

For GitHub, you can use deploy keys with read only access.

1. Go to your project on GitHub
2. Click on Settings
3. Click on Deploy Keys
4. Click on Add deploy key
5. Put a descriptive title and paste the public SSH key from your Read the Docs project
6. Click on Add key

GitLab

For GitLab, you can use deploy keys with read only access.

1. Go to your project on GitLab
2. Click on Settings
3. Click on Repository
4. Expand the Deploy Keys section
5. Put a descriptive title and paste the public SSH key from your Read the Docs project
6. Click on Add key
Bitbucket

For Bitbucket, you can use access keys with read only access.

1. Go your project on Bitbucket
2. Click on Repository Settings
3. Click on Access keys
4. Click on Add key
5. Put a descriptive label and paste the public SSH key from your Read the Docs project
6. Click on Add SSH key

Azure DevOps

For Azure DevOps, you can use SSH key authentication.

1. Go your Azure DevOps page
2. Click on User settings
3. Click on SSH public keys
4. Click on New key
5. Put a descriptive name and paste the public SSH key from your Read the Docs project
6. Click on Add

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 your repository. Refer to your provider’s documentation for the steps required to do this.

Webhooks

To build your documentation on every commit, you’ll need to manually add a webhook, see VCS Integrations. If you are using an unsupported integration, you may need to setup a custom integration using our generic webhook.

3.2.8 Wiping a Build Environment

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

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

Follow these steps to wipe the build environment:

• Go to Versions
• Click on the Edit button of the version you want to wipe on the right side of the page
• Go to the bottom of the page and click the wipe link, next to the “Save” button
Note: By wiping the documentation build environment, all the rst, md, and code files associated with it will be removed but not the documentation already built (HTML and PDF files). Your documentation will still be online after wiping the build environment.

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

### 3.3 Guides for developers and designers

These guides are helpful for developers and designers seeking to extend the authoring tools or customize the documentation appearance.

#### 3.3.1 Installing Private Python Packages

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

Read the Docs uses pip to install your Python packages. If you have private dependencies, you can install them from a private Git repository or a private repository manager.

**From a Git repository**

Pip supports installing packages from a Git repository using the URI form:

```bash
git+https://gitprovider.com/user/project.git@{version}
```

Or if your repository is private:

```bash
git+https://{token}@gitprovider.com/user/project.git@{version}
```

Where version can be a tag, a branch, or a commit. And token is a personal access token with read only permissions from your provider.

To install the package, you need to add the URI in your requirements file. Pip will automatically expand environment variables in your URI, so you don’t have to hard code the token in the URI. See *using environment variables in Read the Docs* for more information.

**Note:** You have to use the POSIX format for variable names (only uppercase letters and _ are allowed), and including a dollar sign and curly brackets around the name ({$API_TOKEN}) for pip to be able to recognize them.

Below you can find how to get a personal access token from our supported providers. We will be using environment variables for the token.
GitHub

You need to create a personal access token with the repo scope. Check the GitHub documentation on how to create a personal token.

URI example:

```
git+https://${GITHUB_TOKEN}@github.com/user/project.git@{version}
```

**Warning:** GitHub doesn’t support tokens per repository. A personal token will grant read and write access to all repositories the user has access to. You can create a machine user to give read access only to the repositories you need.

GitLab

You need to create a deploy token with the read_repository scope for the repository you want to install the package from. Check the GitLab documentation on how to create a deploy token.

URI example:

```
git+https://${GITLAB_TOKEN_USER}:${GITLAB_TOKEN}@gitlab.com/user/project.git@{version}
```

Here GITLAB_TOKEN_USER is the user from the deploy token you created, not your GitLab user.

Bitbucket

You need to create an app password with Read repositories permissions. Check the Bitbucket documentation on how to create an app password.

URI example:

```
git+https://${BITBUCKET_USER}:${BITBUCKET_APP_PASSWORD}@bitbucket.org/user/project.git@{version}
```

Here BITBUCKET_USER is your Bitbucket user.

**Warning:** Bitbucket doesn’t support app passwords per repository. An app password will grant read access to all repositories the user has access to.

From a repository manager other than PyPI

Pip by default will install your packages from PyPI. If you are using a repository manager like pypiserver, or Nexus Repository, you need to set the `--index-url` option. You have two ways of set that option:

- Put `--index-url=https://my-index-url.com/` at the top of your requirements file. See Requirements File Format.
3.3.2 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`.
- Make sure you are using SSH URLs for your submodules (git@github.com:readthedocs/readthedocs.org.git for example) in your `.gitmodules` file, not http URLs.

**Table of contents**
- GitHub
- Azure DevOps
- Others

**GitHub**

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 *public SSH key from your Read the Docs project*
5. Click on *Add SSH key*

**Azure DevOps**

Azure DevOps does not have per-repository SSH keys, but keys can be added to a user instead. As long as this user has access to your main repository and all its submodules, Read the Docs can clone all the repositories with the same key.

*See also:*

*Allow access to your Azure DevOps repository with an SSH key.*

**Others**

GitLab and Bitbucket allow you to reuse the same 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 submodule repository.

*See also:*

*Giving access to your project with an SSH key*

### 3.3.3 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`:

```python
## 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:

```python
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`:

```python
## 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
/** 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 {
    /* ... */
}
```

3.3.4 Reproducible Builds

Your docs depend on tools and other dependencies to be built. If your docs don’t have reproducible builds, an update in a dependency can break your builds when least expected, or make your docs look different from your local version. This guide will help you to keep your builds working over time, and in a reproducible way.

Contents

- Building your docs
- Using a configuration file
- Don’t rely on implicit dependencies
- Pinning dependencies

Building your docs

To test your build process, you can build them locally in a clean environment (this is without any dependencies installed). Then you should make sure you are running those same steps on Read the Docs.

You can configure how your project is built from the web interface (`Admin` tab), or by using a configuration file (recommended). If you aren’t familiar with these tools, check our docs:

- Getting Started with Sphinx
- Getting Started with MkDocs
- Configuration File V2
Note: You can see the exact commands that are run on Read the Docs by going to the Builds tab of your project.

Using a configuration file

If you use the web interface to configure your project, the options are applied to all versions and builds of your docs, and can be lost after changing them over time. Using a configuration file provides you per version settings, and those settings live in your repository.

A configuration file with explicit dependencies looks like this:

```yaml
# File: .readthedocs.yaml

version: 2

build:
  os: "ubuntu-20.04"
  tools:
    python: "3.9"

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

# Explicitly set the version of Python and its requirements
python:
  install:
    - requirements: docs/requirements.txt

# File: docs/requirements.txt

# Defining the exact version will make sure things don't break
sphinx==4.2.0
sphinx_rtd_theme==1.0.0
readthedocs-sphinx-search==0.1.1
```

Don't rely on implicit dependencies

By default Read the Docs will install the tool you chose to build your docs, and other dependencies, this is done so new users can build their docs without much configuration.

We highly recommend not to assume these dependencies will always be present or that their versions won't change. Always declare your dependencies explicitly using a configuration file, for example:

**Good**: Your project is declaring the Python version explicitly, and its dependencies using a requirements file.

```yaml
# File: .readthedocs.yaml

version: 2

build:
  os: "ubuntu-20.04"
```

(continues on next page)
tools:
  python: "3.9"

sphinx:
  configuration: docs/conf.py

python:
  install:
    - requirements: docs/requirements.txt

Bad: Your project is relying on the default Python version and default installed dependencies.

# File: .readthedocs.yaml

version: 2
sphinx:
  configuration: docs/conf.py

Pinning dependencies

As you shouldn’t rely on implicit dependencies, you shouldn’t rely on undefined versions of your dependencies. Some examples:

Good: The specified versions will be used for all your builds, in all platforms, and won’t be updated unexpectedly.

# File: docs/requirements.txt

sphinx==4.2.0
sphinx_rtd_theme==1.0.0
readthedocs-sphinx-search==0.1.1

# File: docs/environment.yaml

name: docs
channels:
  - conda-forge
  - defaults
dependencies:
  - sphinx==4.2.0
  - nbsphinx==0.8.1
  - pip:
    - sphinx_rtd_theme==1.0.0

Bad: The latest or any other already installed version will be used, and your builds can fail or change unexpectedly any time.

# File: docs/requirements.txt

sphinx
sphinx_rtd_theme
readthedocs-sphinx-search
Check the pip user guide for more information about requirements files, or our Conda docs about environment files.

**Tip:** Remember to update your docs’ dependencies from time to time to get new improvements and fixes. It also makes it easy to manage in case a version reaches its end of support date.

### 3.3.5 Embedding Content From Your Documentation

Read the Docs allows you to embed content from any of the projects we host and specific allowed external domains (currently, ['docs\.\python\.org', 'docs\.\scipy\.org', 'docs\.\sympy\.org', 'www\.sphinx-doc\.org', 'numpy\.org']). This allows reuse of content across sites, making sure the content is always up to date.

There are a number of uses cases for embedding content, so we’ve built our integration in a way that enables users to build on top of it. This guide will show you some of our favorite integrations:

- **Contextualized tooltips on documentation pages**
- **Inline help on application website**
- **Calling the Embed API directly**

**Contextualized tooltips on documentation pages**

Tooltips on your own documentation are really useful to add more context to the current page the user is reading. You can embed any content that is available via reference in Sphinx, including:

- Python object references
- Full documentation pages
- Sphinx references
- Term definitions

We built a Sphinx extension called `sphinx-hoverxref` on top of our Embed API you can install in your project with minimal configuration.

Here is an example showing a tooltip when you hover with the mouse a reference:

You can find more information about this extension, how to install and configure it in the `hoverxref` documentation.
Importing Your Documentation

To import a public documentation, visit the Import page. For private repositories, please contact support.

If you have connected your project with Read the Docs, you will see a list of your repositories that can be imported. The import icon next to the repository name should be filled in, along with your project's information.

Manually import your documentation

If you do not have a connection with Read the Docs, you need to select Import Manual. You will then need to manually configure your documentation. You need to know the location of your repository as well. When importing your documentation, you will be asked for the repository URL, including the branch or the commit ID (if you cannot supply the URL). The URL is normally the URL on the project's repository.

Some examples:

```
# import the default documentation
Docs > Importing Your Documentation

# import a branch
Docs > Importing Your Documentation

# import a specific commit
Docs > Importing Your Documentation
```

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 `0.5.2`. If you are using a different version, you will need to specify it in your `readthedocs.yml` file.

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

Fig. 10: Tooltip shown when hovering on a reference using sphinx-hoverxref.
Inline help on application website

This allows us to keep the official documentation as the single source of truth, while having great inline help in our application website as well. On the “Automation Rules” admin page we could embed the content of our Automation Rules documentation page and be sure it will be always up to date.

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

The following example will fetch the section “Creating an automation rule” in page automation-rules.html from our own docs and will populate the content of it into the #help-container div element.

```javascript
<script type="text/javascript">
var params = {
  // 'doctool': 'sphinx',
  // 'doctoolversion': '4.2.0',
};
var url = 'https://readthedocs.org/api/v3/embed/?' + $.param(params);
$.get(url, function(data) {
  $('#help-container').content(data['content']);
});
</script>
<div id="help-container"></div>
```

You can modify this example to subscribe to .on click Javascript event, and show a modal when the user clicks in a “Help” link.

**Tip:** Take into account that if the title changes, your section argument will break. To avoid that, you can manually define Sphinx references above the sections you don’t want to break. For example,

```
.. in your .rst document file

.. _unbreakable-section-reference:

Creating an automation rule
---------------------------

This is the text of the section.

```

To link to the section “Creating an automation rule” you can send section=unbreakable-section-reference. If you change the title it won’t break the embedded content because the label for that title will still be unbreakable-section-reference.

Please, take a look at the Sphinx :ref: role documentation for more information about how to create references.

3.3. Guides for developers and designers 131
Calling the Embed API directly

Embed API lives under https://readthedocs.org/api/v3/embed/ URL and accept the URL of the content you want to embed. Take a look at its own documentation to find out more details.

You can click on the following links and check a live response directly in the browser as examples:

•  https://readthedocs.org/api/v3/embed/?url=https://docs.sympy.org/latest/tutorial/gotchas.html%23equals-signs

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

3.3.6 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.

Our Docker images use Miniconda, a minimal conda installer. After specifying your project requirements using a conda environment.yml file, Read the Docs will create the environment (using conda env create) and add the core dependencies needed to build the documentation.

Creating the environment.yml

There are several ways of exporting a conda environment:

•  conda env export will produce a complete list of all the packages installed in the environment with their exact versions. This is the best option to ensure reproducibility, but can create problems if done from a different operative system than the target machine, in our case Ubuntu Linux (check out our Docker images for further information).

•  conda env export --from-history will only include packages that were explicitly requested in the environment, excluding the transitive dependencies. This is the best option to maximize cross-platform compatibility, however it may include packages that are not needed to build your docs.

•  And finally, you can also write it by hand. This allows you to pick exactly the packages needed to build your docs (which also results in faster builds) and overcomes some limitations in the conda exporting capabilities.

For example, using the second method for an existing environment:

$ conda activate rtd38
(rtd38) $ conda env export --from-history | tee environment.yml
name: rtd38
channels:
Read the Docs will override the name and prefix of the environment when creating it, so they can have any value, or not be present at all.

**Tip:** Bear in mind that `rasterio==1.2` (double `==`) will install version 1.2.0, whereas `python=3.8` (single `=`) will fetch the latest 3.8.* version, which is 3.8.8 at the time of writing.

**Warning:** Pinning Sphinx and other Read the Docs core dependencies is not yet supported by default when using conda (see this GitHub issue for discussion). If your project needs it, request that we enable the `CONDA_APPEND_CORE_REQUIREMENTS` feature flag.

### Effective use of channels

Conda packages are usually hosted on https://anaconda.org/, a registration-free artifact archive maintained by Anaconda Inc. Contrary to what happens with the Python Package Index, different users can potentially host the same package in the same repository, each of them using their own *channel*. Therefore, when installing a conda package, conda also needs to know which channels to use, and which ones take precedence.

If not specified, conda will use `defaults`, the channel maintained by Anaconda Inc. and subject to Anaconda Terms of Service. It contains well-tested versions of the most widely used packages. However, some packages are not available on the `defaults` channel, and even if they are, they might not be on their latest versions.

As an alternative, there are channels maintained by the community that have a broader selection of packages and more up-to-date versions of them, the most popular one being `conda-forge`.

To use the `conda-forge` channel when specifying your project dependencies, include it in the list of channels in `environment.yml`, and conda will rank them in order of appearance. To maximize compatibility, we recommend putting `conda-forge` above `defaults`:

```yaml
name: rtd38
channels:
  - conda-forge
  - defaults
dependencies:
  - python=3.8
# Rest of the dependencies
```

**Tip:** If you want to opt out the `defaults` channel completely, replace it by `nodefaults` in the list of channels. See the relevant conda docs for more information.
Making builds faster with mamba

One important thing to note is that, when enabling the conda-forge channel, the conda dependency solver requires a large amount of RAM and long solve times. This is a known issue due to the sheer amount of packages available in conda-forge.

As an alternative, you can instruct Read the Docs to use mamba, a drop-in replacement for conda that is much faster and reduces the memory consumption of the dependency solving process.

To do that, add a .readthedocs.yaml configuration file with these contents:

```yaml
version: 2
build:
  os: "ubuntu-20.04"
  tools:
    python: "mambaforge-4.10"
conda:
  environment: environment.yml
```

You can read more about the build.tools.python configuration in our documentation.

Mixing conda and pip packages

There are valid reasons to use pip inside a conda environment: some dependency might not be available yet as a conda package in any channel, or you might want to avoid precompiled binaries entirely. In either case, it is possible to specify the subset of packages that will be installed with pip in the environment.yml file. For example:

```yaml
name: rtd38
channels:
  - conda-forge
  - defaults
dependencies:
  - rasterio>=1.2
  - python=3.8
  - pytorch-cpu=1.7
  - pip>=20.1  # pip is needed as dependency
  - pip:
    - black>=20.8b1
```

The conda developers recommend in their best practices to install as many requirements as possible with conda, then use pip to minimize possible conflicts and interoperability issues.

**Warning:** Notice that conda env export --from-history does not include packages installed with pip, see this conda issue for discussion.
Compiling your project sources

If your project contains extension modules written in a compiled language (C, C++, FORTRAN) or server-side JavaScript, you might need special tools to build it from source that are not readily available on our Docker images, such as a suitable compiler, CMake, Node.js, and others.

Luckily, conda is a language-agnostic package manager, and many of these development tools are already packaged on conda-forge or more specialized channels.

For example, this conda environment contains the required dependencies to compile Slycot on Read the Docs:

```
name: slycot38
channels:
  - conda-forge
  - defaults
dependencies:
  - python=3.8
  - cmake
  - numpy
  - compilers
```

Troubleshooting

If you have problems on the environment creation phase, either because the build runs out of memory or time or because some conflicts are found, you can try some of these mitigations:

- Reduce the number of channels in environment.yml, even leaving conda-forge only and opting out of the defaults adding nodefaults.
- Constrain the package versions as much as possible to reduce the solution space.
- Use mamba, an alternative package manager fully compatible with conda packages.
- And, if all else fails, request more resources.

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.
3.3.7 Specifying your dependencies with Poetry

Declaring your project metadata

Poetry is a PEP 517-compliant build backend, which means that it can generate your project metadata using a standardized interface that can be consumed directly by pip. To do that, first make sure that the build-system section of your pyproject.toml declares the build backend as follows:

Listing 8: pyproject.toml

```
[build-system]
requires = ["poetry_core>=1.0.0"]
build-backend = "poetry.core.masonry.api"
```

Then, you will be able to install it on Read the Docs just using pip, with a configuration like this:

Listing 9: .readthedocs.yaml

```
version: 2
build:
  os: ubuntu-20.04
  tools:
    python: "3.9"

python:
  install:
    - method: pip
      path: .
```

For example, the rich Python library uses Poetry to declare its library dependencies and installs itself on Read the Docs with pip.

Locking your dependencies

With your pyproject.toml file you are free to specify the dependency versions that are most appropriate for your project, either by leaving them unpinned or setting some constraints. However, to achieve Reproducible Builds it is better that you lock your dependencies, so that the decision to upgrade any of them is yours. Poetry does this using poetry.lock files that contain the exact versions of all your transitive dependencies (that is, all the dependencies of your dependencies).

The first time you run poetry install in your project directory Poetry will generate a new poetry.lock file with the versions available at that moment. You can then commit your poetry.lock to version control so that Read the Docs also uses these exact dependencies.
3.3.8 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:

```html
{% 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:

```css
/* 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.

3.3.9 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 mamba instead of conda

If you need conda packages to build your documentation, you can use mamba as a drop-in replacement to conda, which requires less memory and is noticeably faster.

Document Python modules API statically

If you are installing a lot of Python dependencies just to document your Python modules API using sphinx.ext.autodoc, you can give a try to sphinx-autoapi Sphinx’s extension instead which should produce the exact same output but running statically. This could drastically reduce the memory and bandwidth required to build your docs.

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.

3.3.10 Adding “Edit Source” links on your Sphinx theme

Read the Docs injects some extra variables in the Sphinx html_context that are used by our Sphinx theme to display “edit source” links at the top of all pages. You can use these variables in your own Sphinx theme as well.
More information can be found on Sphinx documentation.

GitHub

If you want to integrate GitHub, these are the required variables to put into your conf.py:

```python
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
}
```

They can be used like this:
If you want to integrate Bitbucket, these are the required variables to put into your `conf.py`:

```text
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
}
```

They can be used like this:

```text
{% 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 %}
```

If you want to integrate Gitlab, these are the required variables to put into your `conf.py`:

```text
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
}
```

They can be used like this:

```text
{% 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.
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 | Feature Flags
- **Multi-language documentation**: Translations and localization
- **Redirects**: Automatic redirects
- **Versions**: Automation rules
- **Topic specific guides**: How-to guides
- **Extending Read the Docs**: REST API

### 4.1 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:


### 4.1.1 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.
4.1.2 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/

4.1.3 Search

Search on the parent project will include results from its subprojects. If you search on the v1 version of the parent project, results from the v1 version of its subprojects will be included, or from the default version for subprojects that don’t have a v1 version.

This is currently the only way to share search results between projects, we do not yet support sharing search results between sibling subprojects or arbitrary projects.

4.2 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

4.2.1 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.

4.2.2 Effects

Links pointing to the root URL of the project 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 https://pip.readthedocs.io/ rather than redirecting to https://pip.readthedocs.io/en/latest/.

**Warning:** Documentation at /<language>/<default_version>/ will stop working. Remember to set Canonical URLs to tell search engines like Google what to index, and to create User-defined Redirects to avoid broken incoming links.
4.3 Feature Flags

Read the Docs offers some additional flag settings which are disabled by default for every project and can only be enabled by contacting us through our support form or reaching out to the administrator of your service.

4.3.1 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-renew 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

**LIST_PACKAGES_INSTALLED_ENV**: List packages installed in the environment ("pip list" or "conda list") on build’s output

**DONT_CREATE_INDEX**: Do not create index.md or README.rst if the project does not have one.

When Read the Docs detects that your project doesn’t have an index.md or README.rst, it auto-generate one for you with instructions about how to proceed.

In case you are using a static HTML page as index or an generated index from code, this behavior could be a problem. With this feature flag you can disable that.

4.4 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
4.4.1 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.

4.4.2 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:


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 and White Labeling 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 for Sphinx projects to understand the whole process for a documentation with multiples languages in the same repository and how to keep the translations updated on time.
4.5 User-defined Redirects

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

4.5.1 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 *Add* button.

Your redirects will be effective immediately.

**Note:** For the time being, redirects are only implemented in case of a *404 File Not Found* error.

4.5.2 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:

<table>
<thead>
<tr>
<th>Type: Prefix Redirect</th>
</tr>
</thead>
<tbody>
<tr>
<td>From URL: /dev/</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>Type: Page Redirect</th>
</tr>
</thead>
<tbody>
<tr>
<td>From URL: /example.html</td>
</tr>
<tr>
<td>To URL: /examples/intro.html</td>
</tr>
</tbody>
</table>

Note that the / at the start doesn’t count the /$lang/$version prefix (e.g. /en/latest), but just the user-controlled section of the URL. If you want to set directs only for some languages or some versions, you should use Exact Redirects with the fully-specified path.

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:

<table>
<thead>
<tr>
<th>Type: Exact Redirect</th>
</tr>
</thead>
<tbody>
<tr>
<td>From URL: /dev/install.html</td>
</tr>
<tr>
<td>To URL: /en/latest/installing-your-site.html</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>Type: Exact Redirect</th>
</tr>
</thead>
<tbody>
<tr>
<td>From URL: /en/2.0/$rest</td>
</tr>
<tr>
<td>To URL: /en/3.0/</td>
</tr>
</tbody>
</table>

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
Similarly, if you maintain several branches of your documentation (e.g. 3.0 and latest) and decide to move pages in latest but not the older branches, you can use Exact Redirects to do so.

**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.

### 4.6 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.

#### 4.6.1 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 prefix. You can reach this page by going to https://docs.readthedocs.io/page/automatic-redirects.html. This allows you to create links that are always up to date.

Another way to handle this is the latest version. You can set your latest version to a specific version and just always link to latest. You can read more about this in our versions page.

#### 4.6.2 Root URL

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

```markdown
<table>
<thead>
<tr>
<th>docs.readthedocs.io</th>
<th>docs.readthedocs.io/en/latest/</th>
</tr>
</thead>
</table>
```

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.

### 4.6. Automatic Redirects
rtfd.io

Links to rtfd.io 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.

You can reach these docs at https://docs.rtfd.io.

### 4.6.3 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:

<table>
<thead>
<tr>
<th>Redirect Pattern</th>
<th>Target URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td>/en/latest/</td>
</tr>
<tr>
<td>/en/</td>
<td>/en/latest/</td>
</tr>
<tr>
<td>/latest/</td>
<td>/en/latest/</td>
</tr>
</tbody>
</table>

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

### 4.7 Automation Rules

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

#### 4.7.1 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

#### 4.7.2 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.
4.7.3 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.

4.7.4 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.

4.7.5 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.
- **Hide version**: Hides the version. If the version is not active, activates it and builds the version. See **Version States**.
- **Make version public**: Sets the version’s privacy level to public. See **Privacy levels**.
- **Make version private**: Sets the version’s privacy level to private. See **Privacy levels**.
- **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.
- **Delete version**: When a branch or tag is deleted from your repository, Read the Docs will delete it **only if isn’t active**. This action allows you to delete active versions when a branch or tag is deleted from your repository.

**Note**: The default version isn’t deleted even if it matches a rule. You can use the **Set version as default** action to change the default version before deleting the current one.

**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 **Versioned Documentation**.

4.7.6 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).
4.7.7 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

Delete an active version when a branch is deleted

- Match: Any version
- Version type: Branch
- Action: Delete 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

4.8 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.

4.8.1 API v3

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.

**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
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.

Read the Docs Community

Read the Docs for Business

---

**Note:** On Read the Docs Community, you will find your access Token under your profile settings.
**Note:** On Read the Docs for Business, 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).

Read the Docs Community

Read the Docs for Business

**Tip:** You can browse the full API by accessing its root URL: [https://readthedocs.org/api/v3/](https://readthedocs.org/api/v3/)

**Tip:** You can browse the full API by accessing its root URL: [https://readthedocs.com/api/v3/](https://readthedocs.com/api/v3/)

**Note:** If you are using *Read the Docs for Business* take into account that you will need to replace [https://readthedocs.org/](https://readthedocs.org/) by [https://readthedocs.com/](https://readthedocs.com/) in all the URLs used in the following examples.

---

### Projects

### Projects list

**GET** /api/v3/projects/

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

**Example request:**

- Bash
- Python
Example response:

```json
{
    "count": 25,
    "next": "https://readthedocs.org/api/v3/projects/?limit=10&offset=10",
    "previous": null,
    "results": [{
        "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": {
            "home": "https://pip.pypa.io/"
        },
        "tags": [
            "distutils",
            "easy_install",
            "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

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

The results in response is an array of project data, which is same as GET /api/v3/projects/(string:project_slug)/.

Note:

Read the Docs for Business, also accepts

Query Parameters

• expand (string) – with organization and teams.

Project details

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

Example request:

Bash

```bash
```

Python

```python
import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/'
TOKEN = '<token>'
```

(continues on next page)
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
print(response.json())

Example response:

```json
{
    "id": 12345,
    "name": "Pip",
    "slug": "pip",
    "created": "2010-10-23T18:31:+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": {
        "home": "https://pip.pypa.io/"
    },
    "tags": [
        "distutils",
        "easy_install",
        "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/",
```

(continues on next page)
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.

---

**Note:**

*Read the Docs for Business*, also accepts

Query Parameters

- **expand** *(string)* – with `organization` and `teams`.

---

**Project create**

**POST /api/v3/projects/**

Import a project under authenticated user.

**Example request:**

**Bash**

```bash
$ curl \
   -X POST \
   -H "Content-Type: application/json" \
   -d @body.json
```

**Python**

```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'))
response = requests.post(URL, json=data, headers=HEADERS)
print(response.json())
```
The content of `body.json` is like,

```json
{
    "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

Note:

`Read the Docs for Business`, also accepts

Request JSON Object

- **organization** *(string)* – required organization’s slug under the project will be imported.
- **teams** *(string)* – optional teams’ slugs the project will belong to.

**Project update**

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

Update an existing project.

Example request:

Bash

```bash
$ curl \
   -X PATCH \
   -H "Content-Type: application/json" \
   -d @body.json
```

Python

```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,

```json
{
  "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",
  "analytics_disabled": false,
  "single_version": false,
  "external_builds_enabled": true,
}
```

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 Versioned Documentation 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

```bash
```

Python

```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:

```json
{
    "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
- **privacy_level** *(string)* – return versions with specific privacy level *(public or private)*
- **slug** *(string)* – return versions with matching slug
- **type** *(string)* – return versions with specific type *(branch or tag)*
- **verbose_name** *(string)* – return versions with matching version name

Version detail

**GET** /api/v3/projects/(string: project_slug)/versions/(string: version_slug)/

Retrieve details of a single version.

Example request:

**Bash**

```bash
```

**Python**

```python
import requests
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response:

```json
{
    "id": 71652437,
    "slug": "stable",
    "verbose_name": "stable",
}
```
"identifier": "3a6b3995c141c0888af6591a59240ba5db7d914",
"ref": "19.0.2",
"built": true,
"active": true,
"hidden": false,
"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": {
  "dashboard": {
  },
  "vcs": "https://github.com/pypa/pip/tree/19.0.2"
},
"_links": {
  "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
Python

$ curl \
-X PATCH \
   versions/0.23/ \
(continues on next page)
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,

```json
{
    "active": true,
    "hidden": false
}
```

**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/

Example request:

Bash

```bash
```

Python

```python
import requests
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,
        "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",
```

(continues on next page)
"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

```bash
```

Python

```python
import requests
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response:

```json
{
  "count": 15,
```
"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
    versions/latest/builds/

Python

import requests
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


Python

import requests
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


Python

import requests
URL = 'https://readthedocs.org/api/v3/projects/pip/subprojects/'

(continues on next page)
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/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

```bash
$ curl \
  -X POST \
  -H "Content-Type: application/json" \
  -d @body.json
```

Python

```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,

```json
{
    "child": "subproject-child-slug",
    "alias": "subproject-alias"
}
```
Note: child must be a project that you have access to. Or if you are using Read the Docs for Business, additionally the project must be under the same organization as the parent project.

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

Python

```bash
$ curl \
-X DELETE \
˓
→subprojects/subproject-alias/
```

```python
import requests
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/
Retrieves a list of all translations for a project.

Example request:

Bash


Python

```python
import requests
TOKEN = '<token>'
HEADERS = {'Authorization': f'token {TOKEN}'}
response = requests.get(URL, headers=HEADERS)
print(response.json())
```

Example response:

```json
{
    "count": 25,
    "previous": null,
    "results": [{
        "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": {
            "home": "https://pip.pypa.io"
        },
        "tags": [
    
```

(continues on next page)
The results in response is an array of project data, which is same as GET /api/v3/projects/(string:project_slug)/.

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

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/",
    "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


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/(str: project_slug)/redirects/

Create a redirect for this project.

Example request:

Bash

```bash
$ curl \
  -X POST \
  -H "Content-Type: application/json" \
  -d @body.json
```

Python

```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,

```json
{
  "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

```bash
$ curl \
-X PUT \
-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,

```json
{
  "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

Python
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 for example useful to define secrets in a safe way that can be used by your documentation to build properly. Environment variables can also be made public, allowing for them to be used in PR builds. See Environment Variables.

Environment Variable details

GET /api/v3/projects/\(str: \text{project\_slug}\)/environmentvariables/

int: \(\text{environmentvariable\_id}\)/ Retrieve details of a single environment variable for a project.

Example request

Bash

```bash
```

Python

```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

```json
{
    "_links": {
        "_self": "https://readthedocs.org/api/v3/projects/project/
        "environmentvariables/1/",
        "project": "https://readthedocs.org/api/v3/projects/project/"
    }
}
(continues on next page)```
Environment Variables listing

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

Example request
Bash


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/(str: project_slug)/environmentvariables/
Create an environment variable for this project.

Example request
Bash

Python

$ curl \n   -X POST \n   -H "Authorization: Token <token>" https://readthedocs.org/api/v3/projects/pip/environmentvariables/ \n(continues on next page)
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/id:environmentvariable_id/ Delete an environment variable for this project.

Example request:

Bash

```bash
$ curl \
   -X DELETE \
   "environmentvariables/1/
```

Python

```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

**Organizations**

**Note:** The `/api/v3/organizations/` endpoint is only available in *Read the Docs for Business* currently. We plan to have organizations on Read the Docs Community in a near future and we will add support for this endpoint at the same time.

**Organizations list**

**GET /api/v3/organizations/**

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

**Example request:**

Bash

```bash
```

Python

```python
import requests

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

**Example response:**

```json
{
    "count": 1,
    "next": null,
    "previous": null,
    "results": [
        {
            "_links": {
                "_self": "https://readthedocs.com/api/v3/organizations/pypa/",
                "projects": "https://readthedocs.com/api/v3/organizations/pypa/projects/
            },
            "created": "2019-02-22T21:54:52.768630Z",
            "description": "",
            "disabled": false,
            "email": "pypa@psf.org",
        }
    ]
}
```

(continues on next page)
Organization details

GET /api/v3/organizations/(string: organization_slug)/

Retrieve details of a single organization.

Example request:

Bash

```bash
```

Python

```python
import requests

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

Example response:

```json
{
   "_links": {
      "_self": "https://readthedocs.com/api/v3/organizations/pypa/",
      "projects": "https://readthedocs.com/api/v3/organizations/pypa/projects/"
   },
   "created": "2019-02-22T21:54:52.768630Z",
   "description": "",
   "disabled": false,
   "email": "pypa@psf.com",
   "modified": "2020-07-02T12:35:32.418423Z",
   "name": "Python Package Authority",
   "owners": [
      {
         "username": "dstufft"
      }
   ],
   "slug": "pypa",
```
"url": "https://github.com/pypa/
}

**Organization projects list**

**GET /api/v3/organizations/(string: organization_slug)/projects/**

Retrieve list of projects under an organization.

**Example request:**

Bash

```bash
```

Python

```python
import requests

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

**Example response:**

```json
{
    "count": 1,
    "next": null,
    "previous": null,
    "results": [
        {
            "_links": {
            },
            "default_branch": "master",
            "default_version": "latest",
            "environments": [
                {
                    "id": 1,
                    "name": "Linux",
                    "python_version": "3.6",
                    "python_any": false,
                    "python_version_tag": false,
                    "is_deprecated": false,
                }
            
```
Remote Organizations

Remote Organizations are the VCS organizations connected via GitHub, GitLab and BitBucket.

Remote Organization listing

GET /api/v3/remote/organizations/
Retrieve a list of all Remote Organizations for the authenticated user.

Example request:

Bash

```bash
```

Python

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

Example response:

```
{
    "count": 20,
    "next": "api/v3/remote/organizations/?limit=10&offset=10",
    "previous": null,
    "results": [
        {
            "created": "2019-04-29T10:00:00Z",
            "modified": "2019-04-29T12:00:00Z",
            "name": "Organization Name",
            "pk": 1,
            "slug": "organization",
            "url": "https://github.com/organization",
            "vcs_provider": "github"
        }
    ]
}
```

The results in response is an array of remote organizations data.

**Query Parameters**

- **name** *(string)* – return remote organizations with containing the name
- **vcs_provider** *(string)* – return remote organizations for specific vcs provider (github, gitlab or bitbucket)

**Request Headers**

- **Authorization** – token to authenticate.

**Remote Repositories**

Remote Repositories are the importable repositories connected via GitHub, GitLab and BitBucket.

**Remote Repository listing**

**GET** /api/v3/remote/repositories/

Retrieve a list of all Remote Repositories for the authenticated user.

**Example request:**

Bash

```
$ curl 
```

Python

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

Example response:

```json
{
   "count": 20,
   "next": "api/v3/remote/repositories/?expand=projects,remote_organization&limit=10&offset=10",
   "previous": null,
   "results": [
      {
         "remote_organization": {
            "created": "2019-04-29T10:00:00Z",
            "modified": "2019-04-29T12:00:00Z",
            "name": "Organization Name",
            "pk": 1,
            "slug": "organization",
            "url": "https://github.com/organization",
            "vcs_provider": "github"
         },
         "project": [{
            "id": 12345,
            "name": "project",
            "slug": "project",
            "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/organization/project",
               "type": "git"
            },
            "default_version": "stable",
            "default_branch": "master",
            "subproject_of": null,
            "translation_of": null,
            "urls": {
               "documentation": "http://project.readthedocs.io/en/stable/",
               "home": "https://readthedocs.org/projects/project/"
            }
         }
      }
   ]
}
```
The results in response is an array of remote repositories data.

**Query Parameters**

- **name** *(string)* – return remote repositories containing the name
- **vcs_provider** *(string)* – return remote repositories for specific vcs provider (github, gitlab or bitbucket)
- **organization** *(string)* – return remote repositories for specific remote organization (using remote organization slug)
- **expand** *(string)* – allows to add/expand some extra fields in the response. Allowed values are projects and remote_organization. Multiple fields can be passed separated by commas.

**Request Headers**
Embed

GET /api/v3/embed/

Retrieves HTML-formatted content from documentation page or section. Read *Embedding Content From Your Documentation* to know more about how to use this endpoint.

Example request:

```bash
```

Example response:

```json
{
    "url": "https://docs.readthedocs.io/en/latest/features.html#read-the-docs-features",
    "fragment": "read-the-docs-features",
    "content": 
        "<div class="section" id="read-the-docs-features">
            <h1>Read the Docs ...
        </
    "external": false
}
```

Response JSON Object

- **url** *(string)* – URL of the document.
- **fragment** *(string)* – fragment part of the URL used to query the page.
- **content** *(string)* – HTML content of the section.
- **external** *(string)* – whether or not the page is hosted on Read the Docs or externally.

Query Parameters

- **url** *(string)* – full URL of the document (with optional fragment) to fetch content from.
- **doctool** *(string)* – optional documentation tool key name used to generate the target documentation (currently, only sphinx is accepted)
- **doctoolversion** *(string)* – optional documentation tool version used to generate the target documentation (e.g. 4.2.0).

Note: Passing `?doctool=` and `?doctoolversion=` may improve the response, since the endpoint will know more about the exact structure of the HTML and can make better decisions.
Additional APIs

- *Server side search API.*

4.8.2 API v2

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>
<thead>
<tr>
<th><strong>Warning:</strong> API v2 is planned to be deprecated soon, though we have not yet set a time frame for deprecation yet. We will alert users with our plans when we do.</th>
</tr>
</thead>
<tbody>
<tr>
<td>For now, API v2 is still used by some legacy application operations still, but we highly recommend Read the Docs users use API v3 instead.</td>
</tr>
<tr>
<td>Some improvements in API v3 are:</td>
</tr>
<tr>
<td>• Token based authentication</td>
</tr>
<tr>
<td>• Easier to use URLs which no longer use numerical ids</td>
</tr>
<tr>
<td>• More common user actions are exposed through the API</td>
</tr>
<tr>
<td>• Improved error reporting</td>
</tr>
<tr>
<td>See its full documentation at API v3.</td>
</tr>
</tbody>
</table>

Authentication and authorization

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

Resources

Projects

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

As an example, this documentation is part of the Docs project which has documentation at https://docs.readthedocs.io. You can always view your Read the Docs projects in your project dashboard.

Project list

**GET /api/v2/project/**

Retrieve a list of all Read the Docs projects.

**Example request:**

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

**Example response:**
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.

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.

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

**Response JSON Object**

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

**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

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

**Response JSON Object**

4.8. Public API
**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": "3a6b3995c141c088af6591a59240ba5db7d9914",
  "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,
}
```

(continues on next page)
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 (e.g., 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.

**Embed**

**GET /api/v2/embed/**

Retrieve HTML-formatted content from documentation page or section.

**Example request:**

```bash
$ curl https://readthedocs.org/api/v2/embed/?project=docs&version=latest&doc=features&path=features.html
```
or

```bash
```

**Example response:**

```json
{
    "content": [
        "<div class="section" id="read-the-docs-features">
<h1>Read the Docs...</h1>
        
        "headers": [
            {
                "Read the Docs features": "#"
            },
            {
                "Automatic Documentation Deployment": "#automatic-documentation-deployment"
            },
            {
                "Custom Domains & White Labeling": "#custom-domains-white-labeling"
            },
            {
                "Versioned Documentation": "#versioned-documentation"
            },
            {
                "Downloadable Documentation": "#downloadable-documentation"
            },
            {
                "Full-Text Search": "#full-text-search"
            },
            {
                "Open Source and Customer Focused": "#open-source-and-customer-focused"
            }
        ],
        "url": "https://docs.readthedocs.io/en/latest/features",
        "meta": {
            "project": "docs",
            "version": "latest",
            "doc": "features",
            "section": "read the docs features"
        }
    }
}
```

(continues on next page)
Response JSON Object

- **content** *(string)* – HTML content of the section.
- **headers** *(object)* – section’s headers in the document.
- **url** *(string)* – URL of the document.
- **meta** *(object)* – meta data of the requested section.

Query Parameters

- **project** *(string)* – Read the Docs project’s slug.
- **doc** *(string)* – document to fetch content from.
- **version** *(string)* – *optional* Read the Docs version’s slug (default: latest).
- **section** *(string)* – *optional* section within the document to fetch.
- **path** *(string)* – *optional* full path to the document including extension.
- **url** *(string)* – full URL of the document (and section) to fetch content from.

**Note:** You can call this endpoint by sending at least **project** and **doc** or **url** attribute.

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.
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 | Development installation | Roadmap | Code of conduct

• Policies & Process: Security | DMCA takedown policy | Policy for abandoned projects | Release notes & changelog

• The people and philosophy behind Read the Docs: About Us | Team | Open source philosophy | Our story

• Financial and material support: Advertising | Sponsors

• Read the Docs for Business: Support and additional features

• Legal documents: Terms of service | Privacy policy | Data processing agreement

5.1 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.

5.1.1 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.

5.1.2 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.

You can read all of our Developer documentation to understand more the development of Read the Docs. When contributing code, then please follow the standard Contribution Guidelines set forth at contribution-guide.org.

5.1.3 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.

5.1.4 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.

5.1.5 Helping on translations

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

5.2 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.

5.2.1 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.
5.2.2 Development Installation

These are development setup and standards that are adhered to by the core development team while developing Read the Docs and related services. If you are a contributor to Read the Docs, it might be a good idea to follow these guidelines as well.

To follow these instructions you will need a Unix-like operating system, or Windows Subsystem for Linux (WSL). Other operating systems are not supported.

Note: We do not recommend to follow this guide to deploy an instance of Read the Docs for production usage. Take into account that this setup is only useful for developing purposes.

Set up your environment

1. install Docker following their installation guide.
2. clone the readthedocs.org repository:
   
   $ git clone --recursive-submodules https://github.com/readthedocs/readthedocs.org/
3. install the requirements from common submodule:
   
   $ pip install -r common/dockerfiles/requirements.txt
4. build the Docker image for the servers:
$ inv docker.build

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

5. pull down Docker images for the builders:

   $ inv docker.pull --only-latest

6. start all the containers:

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

7. add read permissions to the storage backend:

   • go to http://localhost:9000/ (MinIO S3 storage backend)
   • login as admin / password
   • click “…” next to the static bucket name and then “Edit Policy”
   • leave “prefix” empty and click “Add” to give “Read Only” access on the static bucket
   • click on the “+” icon on the bottom-right corner, then “Create bucket” with the name media, hit Enter on the keyboard, and repeat the operation above to give “Read Only” access to it

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

**Check that everything works**

1. go to http://community.dev.readthedocs.io and check that the appearance and style looks correct (otherwise the MinIO buckets might be misconfigured, see above)

2. login as admin / admin and verify that the project list appears

3. go to the “Read the Docs” project, click on the “Build version” button to build latest, and wait until it finishes

4. click on the “View docs” button to browse the documentation, and verify that it works

**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, 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).

   • --no-running spins up a new container and open a shell
   • --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++)

**Tip:** You can hit CTRL-p CTRL-p to detach it without stopping the running process.

**inv docker.test** Runs all the test suites inside the container.
- `--arguments` will pass arguments to Tox command (e.g. `--arguments "-e py38 -- -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.

**inv docker.buildassets** Build all the assets and “deploy” them to the storage.

**Adding a new Python dependency**

The Docker image for the servers is built with the requirements defined in the current checked out 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:

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

Once the PR that adds this dependency was merged, 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`:

```python
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.

Configuring connected accounts

These are optional steps to setup the connected accounts (GitHub, GitLab, and BitBucket) in your development environment. This will allow you to login to your local development instance using your GitHub, Bitbucket, or GitLab credentials and this makes the process of importing repositories easier.

However, because these services will not be able to connect back to your local development instance, incoming webhooks will not function correctly. For some services, the webhooks will fail to be added when the repository is imported. For others, the webhook will simply fail to connect when there are new commits to the repository.

- Configure the applications on GitHub, Bitbucket, and GitLab. For each of these, the callback URI is `http://community.dev.readthedocs.io/accounts/<provider>/login/callback/` where `<provider>` is one of `github`, `gitlab`, or `bitbucket_oauth2`. When setup, you will be given a “Client ID” (also called an “Application ID” or just “Key”) and a “Secret”.
- Take the “Client ID” and “Secret” for each service and enter it in your local Django admin at: `http://community.dev.readthedocs.io/admin/socialaccount/socialapp/`. Make sure to apply it to the “Site”.

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.

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.
5.2. Developer documentation

Fig. 1: Configuring an OAuth consumer for local development on Bitbucket
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.

MinIO as Django storage backend  All static and media files are served using Minio—an emulator of S3, 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. All the documentation indexes are updated after a build is finished.

5.2.3 Designing Read the Docs

So you’re thinking of contributing some of your time and design skills to Read the Docs? That’s awesome. This document will lead you through a few features available to ease the process of working with Read the Doc’s CSS and static assets.

To start, you should follow the Development 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.
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


5.2.4 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.

Contributing through the Github UI

If you’re making small changes to the documentation, you can verify those changes through the documentation generated when you open a PR and can be accessed using the Github UI.

1. click the checkmark next to your commit and it will expand to have multiple options
2. click the “details” link next to the “docs/readthedocs.org:docs” item

3. navigate to the section of the documentation you worked on to verify your changes
Contributing from your local machine

If you’re making large changes to the documentation, you may want to verify those changes locally before pushing upstream.

1. clone the readthedocs.org repository:

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

2. create a virtual environment with Python 3.8 (preferably the latest release, 3.8.12 at the time of writing), activate it, and upgrade pip:

```
$ cd readthedocs.org
$ python3.8 -m venv .venv
$ source .venv/bin/activate
(.venv) $ python -m pip install -U pip
```

3. install documentation requirements

```
(.venv) $ pip install -r requirements/testing.txt
(.venv) $ pip install -r requirements/docs.txt
```

4. build the documents

```
(.venv) $ cd docs
(.venv) $ make livehtml
```

5. the documents will be available at http://127.0.0.1:4444/ and will rebuild each time you edit and save a file.

Guidelines

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.
5.2.5 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 to follow our guide to install a development Read the Docs instance first.

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 compile your changes and make them available in the application you need to run:

```bash
inv docker.buildassets
```

Once you are happy with your changes, make sure to check in both files under static and static-src, and commit those.

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.
5.2.6 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.


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:

```python
from django.utils.translation import pgettext

month = pgettext("text for the search button on the form", "Search")
```

Plurals

You have 1 new messages grate on discerning ears. Fortunately, gettext gives us a way to fix that in English and other locales, the ngettext() function:

```python
ngettext('singular sentence', 'plural sentence', count)
```

A more realistic example might be:

```python
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:

```html
<h1>{% trans "Heading" %}</h1>
```

Context can be added, too:

```html
<h1>{% trans "Heading" context "section name" %}</h1>
```

Comments for translators need to precede the internationalized text and must start with the Translators: keyword:

```html
#{ 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:
Welcome, {{ name }}!

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 %}alsoprovidespluralization. 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 ugettext() function in the wrong place, the string may end up in the wrong locale!

Here’s how you might localize a string in a view:

```python
from django.utils.translation import ugettext as _

def my_view(request):
    if request.user.is_superuser:
        msg = _(u'Oh hi, staff!')
    else:
        msg = _(u'You are not staff!')

Interpolation is done through normal Python string formatting:

msg = _(u'Oh, hi, {user}').format(user=request.user.username)

Context information can be supplied by using the pgettext() function:
```
msg = pgettext('the context', 'Search')

Translator comments are normal one-line Python comments:

```
# Translators: A message to users.
msg = _(u'Oh, hi there!')
```

If you need to use plurals, import the `ungettext()` function:

```python
from django.utils.translation import ungettext
n = len(results)
msg = ungettext('Found {0} result', 'Found {0} results', n).format(n)
```

### Lazily Translated Strings

You can use `ugettext()` or `ungettext()` only in views or functions called from views. If the function will be evaluated when the module is loaded, then the string may end up in English or the locale of the last request!

Examples include strings in module-level code, arguments to functions in class definitions, strings in functions called from outside the context of a view. To internationalize these strings, you need to use the `_lazy` versions of the above methods, `ugettext_lazy()` and `ungettext_lazy()`. The result doesn’t get translated until it is evaluated as a string, for example by being output or passed to `unicode()`:

```python
from django.utils.translation import ugettext_lazy as _

class UserProfileForm(forms.ModelForm):
    first_name = CharField(label=_('First name'), required=False)
    last_name = CharField(label=_('Last name'), required=False)
```

In case you want to provide context to a lazily-evaluated gettext string, you will need to use `pgettext_lazy()`.

### 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.

## 5.2. Developer documentation
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

---

### 5.2.7 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.

### 5.2.8 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 Elasticsearch 6.3.

**Local Development Configuration**

Elasticsearch is installed and run as part of the [development installation guide](https://readthedocs.org/docs/read-the-docs/setup/installation.html).

**Indexing into Elasticsearch**

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

```
$ inv docker.manage 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](https://django-elasticsearch-dsl.readthedocs.io/) 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.

Troubleshooting

If you get an error like:

RequestError(400, 'search_phase_execution_exception', 'failed to create query: ...

You can fix this by deleting the page index:

$ inv docker.manage 'search_index --delete'

Note: You’ll need to reindex the projects after this.

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 the `processed_json` property of `HTMLFile`. This property provides python dictionary with document data like `title`, `sections`, `path` etc.

5.2.9 Server Side Search Integration

Read the Docs provides server side search (SSS) in replace of the default search engine of your site. To accomplish this, Read the Docs parses the content directly from your HTML pages.

If you are the author of a theme or a static site generator you can read this document, and follow some conventions in order to improve the integration of SSS with your theme/site.

Indexing

The content of the page is parsed into sections, in general, the indexing process happens in three steps:

1. Identify the main content node.
2. Remove any irrelevant content from the main node.
3. Parse all sections inside the main node.

Read the Docs makes use of ARIA roles and other heuristics in order to process the content.

**Tip**: Following the ARIA conventions will also improve the accessibility of your site. See also [https://webaim.org/techniques/semanticstructure/](https://webaim.org/techniques/semanticstructure/).

---

---
Main content node

The main content node should have a main role (or a main tag), and there should only be one per page. This node is the one that contains all the page content. Example:

```html
<html>
  <head>
    ...
  </head>
  <body>
    <div>
      This content isn't processed
    </div>
    <div role="main">
      All content inside the main node is processed
    </div>
    <footer>
      This content isn't processed
    </footer>
  </body>
</html>
```

Irrelevant content

If you have content inside the main node that isn’t relevant to the page (like navigation items, menus, or search box), make sure to use the correct role or tag for it.

Roles to be ignored:
  • navigation
  • search

Tags to be ignored:
  • nav

Example:

```html
<div role="main">
  ...
  <nav role="navigation">
    ...
  </nav>
  ...
</div>
```
Sections

Sections are h tags, and sections of the same level should be neighbors. Additionally, sections should have an unique id attribute per page (this is used to link to the section). All content below the section, till the new section will be indexed as part of the section. Example:

```html
<div role="main">
   <h1 id="section-title">
      Section title
   </h1>
   <p>
      Content to be indexed
   </p>
   <ul>
      <li>This is also part of the section and will be indexed as well</li>
   </ul>

   <h2 id="2">
      This is the start of a new section
   </h2>
   <p>...
   </p>
   ...

   <h1 id="neighbor-section">
      This section is neighbor of "section-title"
   </h1>
   <p>...
   </p>
</div>
```

Sections can be inside till two nested tags (and have nested sections), and its immediate parent can contain the id attribute. Note that the section content still needs to be below the h tag. Example:

```html
<div role="main">
   <div class="section">
      <h1 id="section-title">
         Section title
      </h1>
      <p>
         Content to be indexed
      </p>
      <ul>
         <li>This is also part of the section</li>
      </ul>
   </div>

   <div class="section">
      <div id="nested-section">
         <h2>
            This is the start of a sub-section
         </h2>
      </div>
   </div>
</div>
```

(continues on next page)
Note: The title of the first section will be the title of the page, falling back to the title tag.

Other special nodes

• Anchors: If the title of your section contains an anchor, wrap it in a headerlink class, so it won’t be indexed as part of the title.

• Code blocks: If a code block contains line numbers, wrap them in a linenos or lineno class, so they won’t be indexed as part of the code.
Overriding the default search

Static sites usually have their own static index, and search results are retrieved via JavaScript. In order for Read the Docs to override the default search as expected, themes from the supported generators must follow these conventions.

Note: Read the Docs will fallback to the original search in case of an error or no results.

Sphinx

Sphinx’s basic theme provides the static/searchtools.js file, which initializes search with the Search.init() method. Read the Docs overrides the Search.query method and makes use of Search.output.append to add the results. A simplified example looks like this:

```javascript
var original_search = Search.query;

function search_override(query) {
    var results = fetch_results(query);
    if (results) {
        for (var i = 0; i < results.length; i += 1) {
            var result = process_result(results[i]);
            Search.output.append(result);
        }
    } else {
        original_search(query);
    }
}

Search.query = search_override;

$(document).ready(function() {
    Search.init();
});
```

Highlights from results will be in a span tag with the highlighted class (This is a <span class="highlighted">result</span>). If your theme works with the search from the basic theme, it will work with Read the Docs’ SSS.

MkDocs

Search on MkDocs is provided by the search plugin, which is included (and activated) by default in MkDocs. The js part of this plugin is included in the templates/search/main.js file, which subscribes to the keyup event of the #mkdocs-search-query element to call the doSearch function (available on MkDocs >= 1.x) on every key press.

Read the Docs overrides the initSearch and doSearch functions to subscribe to the keyup event of the #mkdocs-search-query element, and puts the results into the #mkdocs-search-results element. A simplified example looks like this:

```javascript
var original_search = doSearch;

function search_override() {
    var query = document.getElementById('mkdocs-search-query').value;
    // (continues on next page)
}
var search_results = document.getElementById('mkdocs-search-results');

var results = fetch_results(query);
if (results) {
    empty_results(search_results)
    for (var i = 0; i < results.length; i += 1) {
        var result = process_result(results[i]);
        append_result(result, search_results);
    }
} else {
    original_search();
}

var init_override = function () {
    var search_input = document.getElementById('mkdocs-search-query');
    search_input.addEventListener('keyup', doSearch);
};

window.doSearch = search_override;
window.initSearch = init_override;
initSearch();

Highlights from results will be in a mark tag (This is a <mark>result</mark>). If your theme works with the search plugin of MkDocs, and defines the #mkdocs-search-query and #mkdocs-search-results elements, it will work with Read the Docs’ SSS.

**Note:** Since the templates/search/main.js file is included after our custom search, it will subscribe to the keyup event too, triggering both functions when a key is pressed (but ours should have more precedence). This can be fixed by not including the search plugin (you won’t be able to fall back to the original search), or by creating a custom plugin to include our search at the end (this should be done by Read the Docs).

**Supporting more themes and static site generators**

Currently, Read the Docs supports building documentation from *Sphinx* and *MkDocs*. All themes that follow these conventions should work as expected. If you think other generators or other conventions should be supported, or content that should be ignored or have an especial treatment, or if you found an error with our indexing, let us know in our issue tracker.

**5.2.10 Interesting Settings**

**DOCKER_LIMITS**

Default: `{'memory': '1g', 'time': 600}`

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.

**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.

**RTD_INTERSPHINX_URL**

Default: https://readthedocs.org

This is the domain that is used to fetch the intersphinx inventory file. If not set explicitly this is the PRODUCTION_DOMAIN.

**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.

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 (e.g. 'path.to.MyBuildMediaStorage'). Your class should mixin readthedocs.builds.storage.BuildMediaStorageMixin.

ELASTICSEARCH_DSL

Default:

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

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

Default:

```python
{
    '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,
        }
    }
}
```

**ES_TASK_CHUNK_SIZE**

Default: 500

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.
5.2.11 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

To target a specific environment:

$ tox -e py38

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

**py38** Run our test suite using Python 3.8

**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 Circle CI on every push to our repo at GitHub. You can check out the current build status: https://app.circleci.com/pipelines/github/readthedocs/readthedocs.org

5.2.12 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.

**Warning:** These documents may not match the final implementation, or may be out of date.

**API v3 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 useless 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

Build Images

This document describes how Read the Docs uses the Docker Images and how they are named. Besides, it proposes a path forward about a new way to create, name and use our Docker build images to reduce its complexity and support installation of other languages (e.g. nodejs, rust, go) as extra requirements.

Introduction

We use Docker images to build user’s documentation. Each time a build is triggered, one of our VMs picks the task and go through different steps:

1. run some application code to spin up a Docker image into a container
2. execute git inside the container to clone the repository
3. analyze and parse files (.readthedocs.yml) from the repository outside the container
4. spin up a new Docker container based on the config file
5. create the environment and install docs’ dependencies inside the container
6. execute build commands inside the container
7. push the output generated by build commands to the storage

All those steps depends on specific commands versions: git, python, virtualenv, conda, etc. Currently, we are pinning only a few of them in our Docker images and that have caused issues when re-deploying these images with bugfixes: the images are not reproducible over time.

Note: We have been improving the reproducibility of our images by adding some tests cases. These are run inside the Docker image after it’s built and check that it contains the versions we expect.
To allow users to pin the image we ended up exposing three images: `stable`, `latest` and `testing`. With that naming, we were able to bugfix issues and add more features on each image without asking the users to change the image selected in their config file.

Then, when a completely different image appeared and after testing `testing` image enough, we discarded `stable`, old `latest` became the new `stable` and old `testing` became the new `latest`. This produced issues to people pinning their images to any of these names because after this change, *we changed all the images for all the users* and many build issues arised!

### Goals

- release completely new Docker images without forcing users to change their pinned image (*stable*, *latest*, *testing*)
- allow users to select language requirements instead of an image name
- use a base image with the dependencies that don’t change frequently (OS and base requirements)
- base image naming is tied to the OS version (e.g. Ubuntu LTS)
- allow us to add/update a Python version without affecting the base image
- reduce size on builder VM disks by sharing Docker image layers
- allow users to specify extra languages (e.g. nodejs, rust, go)
- de-motivate the usage of *stable*, *latest* and *testing*; and promote declaring language requirements instead
- new images won’t contain old/deprecated OS (eg. Ubuntu 18) and Python versions (eg. 3.5, miniconda2)
- install language requirements *at built time* using `asdf` and its plugins
- create local mirrors for all languages supported
- deleting a pre-built image won’t make builds to fail; only make them slower
- support only the latest Ubuntu LTS version and keep the previous one as long as it’s officially supported

### Non goals

- allow creation/usage of custom Docker images
- allow to execute arbitrary commands via hooks (eg. `pre_build`)
- automatically build & push *all* images on commit
- pre-built multiple images for all the languages combinations

### Pre-built build image structure

The new pre-built images will depend only on the Ubuntu OS. They will contain all the requirements to add extra languages support at built time via `asdf` command.

- `ubuntu20-base`
  - labels
  - environment variables
  - system dependencies
Instead of building all the Docker image per language versions combination, it will be easier to install all of them at build time using the same steps. Installing a language only adds a few seconds when binaries are provided. However, to reduce the time to install these languages as much as possible, a local mirror hosted on S3 for each language will be created.

It's important to note that Python does not provide binaries and compiling a version takes around ~2 minutes. However, the Python versions could be pre-compiled and expose their binaries via S3 to builders. Then, at build time, the builder will only download the binary and copy it in the correct path.

Note: Depending on the demand, Read the Docs may pre-build the most common combinations of languages used by users. For example, ubuntu20+python39+node14 or ubuntu20+python39+node14+rust1. However, this is seen as an optimization for the future and it's not required for this document.

Build steps

With this new approach, the steps followed by a builder will be:

1. run some application code to spin up the -base Docker image into a container
2. execute git inside the container to clone the repository
3. analyze and parse files (.readthedocs.yaml) from the repository outside the container
4. spin up a new Docker container based on the Ubuntu OS specified in the config file
5. install all language dependencies from the cache
6. create the environment and install docs’ dependencies inside the container
7. execute build commands inside the container
8. push the output generated by build commands to the storage

The main difference with the current approach are:

- the image to spin up is selected depending on the OS version
- all language dependencies are installed at build time
- languages not offering binaries are pre-compiled by Read the Docs and stored in the cache
- miniconda/mambaforge are now managed with the same management tool (e.g. asdf install python miniconda3-4.7.12)
Specifying extra languages requirements

Different users may have different requirements. People with specific language dependencies will be able to install them by using .readthedocs.yaml config file. Example:

```
build:
  os: ubuntu20
  languages:
    python: "3.9"   # supports "pypy3", "miniconda3" and "mambaforge"
    nodejs: "14"
    rust: "1.54"
    golang: "1.17"
```

Important highlights:

- do not treat Python language different from the others (will help us to support other non-Python doctools in the future)
- specifying build.languages.python: "3" will use Python version 3.x.y, and may differ between builds
- specifying build.languages.python: "3.9" will use Python version 3.9.y, and may differ between builds
- specifying build.languages.nodejs: "14" will use nodejs version 14.x.y, and may differ between builds
- if no full version is declared, it will try first latest available on our cache, and then the latest on asdf (it has to match the first part of the version declared)
- specifying minor language versions is not allowed (e.g. 3.7.11)
- not specifying build.os will make the config file parser to fail
- not specifying build.languages will make the config file parsing to fail (at least one is required)
- specifying only build.languages.nodejs and using Sphinx to build the docs, will make the build to fail (e.g. “Command not found”)
- build.image is incompatible with build.os or build.languages and will produce an error
- python.version is incompatible with build.os or build.languages and will produce an error
- Ubuntu 18 will still be available via stable and latest images, but not in new ones
- only a subset (not defined yet) of python, nodejs, rust and go versions on asdf are available to select

**Note:** We are moving away from users specifying a particular Docker image. With the new approach, users will specify the languages requirements they need, and Read the Docs will decide if it will use a pre-built image or will spin up the base one and install these languages on the fly.

However, build.image will be still available for backward compatibility with stable, latest and testing but won’t support the new build.languages config.

Note that knowing exactly what packages users are installing, could allow us to pre-build the most common combinations used images: ubuntu20+py39+node14.
Time required to install languages at build time

Testings using `time` command in ASG instances to install extra languages took these “real” times:

- **build-default**
  - python 3.9.6: 2m21.331s
  - mambaforge 4.10.1: 0m26.291s
  - miniconda3 4.7.12: 0m9.955s
  - nodejs 14.17.5: 0m5.603s
  - rust 1.54.0: 0m13.587s
  - golang 1.17: 1m30.428s

- **build-large**
  - python 3.9.6: 2m33.688s
  - mambaforge 4.10.1: 0m28.781s
  - miniconda3 4.7.12: 0m10.551s
  - nodejs 14.17.5: 0m6.136s
  - rust 1.54.0: 0m14.716s
  - golang 1.17: 1m36.470s

Note that the only one that required compilation was Python. All the others, spent 100% of its time downloading the binary. These download times are much better from EU with a home internet connection.

In the worst scenario: “none of the specified language version has a pre-built image”, the build will require ~5 minutes to install all the language requirements. By providing only pre-built images with the Python version (that’s the most time consuming), build times will only require ~2 minutes to install the others. However, requiring one version of each language is not a common case.

Cache language binaries on S3

`asdf` scripts can be altered to download the `.tar.gz` dist files from a different mirror than the official one. Read the Docs can make usage of this to create a mirror hosted locally on S3 to get faster download speeds. This will make a good improvement for languages that offer binaries: `nodejs`, `rust` and `go`:

- **nodejs** uses `NODEJS_ORG_MIRROR`: https://github.com/asdf-vm/asdf-nodejs/blob/master/lib/utils.sh#L5
- **rust** uses `RUSTUP_UPDATE_ROOT`: https://github.com/rust-lang/rustup/blob/master/rustup-init.sh#L23
- **go** has the URL hardcoded: https://github.com/kennyp/asdf-golang/blob/master/bin/download#L54

However, currently Python does not offer binaries and a different solution is needed. Python versions can be pre-compiled once and expose the output on the S3 for the builders to download and extract in the correct PATH.

**Tip:** Since we are building a special cache for pre-compiled Python versions, we could use the same method for all the other languages instead of creating a full mirror (many Gigabytes) This simple `bash script` download the language sources, compiles it and upload it to S3 without requiring a mirror. Note that it works in the same way for all the languages, not just for Python.
Questions

What Python versions will be pre-compiled and cached?

At start only a small subset of Python version will be pre-compiled:

- 2.7.x
- 3.7.x
- 3.8.x
- 3.9.x
- 3.10.x
- pypy3.x

How do we upgrade a Python version?

Python patch versions can be upgraded by re-compiling the new patch version and making it available in our cache. For example, if version 3.9.6 is the one available and 3.9.7 is released, after updating our cache:

- users specifying build.languages.python: "3.9" will get the 3.9.7 version
- users specifying buildlanguages.python: "3" will get the 3.9.7 version

As we will have control over these version, we can decide when to upgrade (if ever required) and we can roll back if the new pre-compiled version was built with a problem.

Note: Python versions may need to be re-compiled each time that the -base image is re-built. This is due that some underlying libraries that Python depend on may have changed.

Note: Installing always the latest version is harder to maintain. It will require building the newest version each time a new patch version is released. Because of that, Read the Docs will always be behind official releases. Besides, it will give projects different versions more often.

Exposing to the user the patch version would require to cache many different versions ourselves, and if the user selects one patched version that we don’t have cached by mistake, those builds will add extra build time.

How do we add a Python version?

Adding a new Python version requires:

- pre-compile the desired version for each Ubuntu OS version supported
- upload the compressed output to S3
- add the supported version to the config file validator
How do we remove an old Python version?

At some point, an old version of Python will be deprecated (e.g. 3.4) and will be removed. To achieve this, we can just remove the pre-compiled Python version from the cache.

However, unless it’s strictly needed for some specific reason, we shouldn’t require to remove support for a Python version as long as we support the Ubuntu OS version where this version was compiled for.

In any case, we will know which projects are using these versions because they are pinning these specific versions in the config file. We could show a message in the build output page and also send them an email with the EOL date for this image.

However, removing pre-compiled Python version that it’s being currently used by some users won’t make their builds to fail. Instead, that Python version will be compiled and installed at build time; adding a “penalization” time to those projects and motivating them to move forward to a newer version.

How do we upgrade system versions?

We usually don’t upgrade these dependencies unless we upgrade the Ubuntu version. So, they will be only upgraded when we go from Ubuntu 18.04 LTS to Ubuntu 20.04 LTS for example.

Examples of these versions are:

- doxygen
- git
- subversion
- pandoc
- swig
- latex

This case will introduce a new base image. Example, ubuntu22-base in 2022. Note that these images will be completely isolated from the rest and don’t require them to rebuild. This also allow us to start testing a newer Ubuntu version (e.g. 22.04 LTS) without breaking people’s builds, even before it’s officially released.

How do we add an extra requirement?

In case we need to add an extra requirement to the base image, we will need to rebuild all of them. The new image may have different package versions since there may be updates on the Ubuntu repositories. This conveys some risk here, but in general we shouldn’t require to add packages to the base images.

In case we need an extra requirement for all our images, I’d recommend to add it when creating a new base image.

If it’s strongly needed and we can’t wait for a new base image, we could install it at build time in a similar way as we do with build.apt_packages as a temporal workaround.
How do we create a mirror for each language?

A mirror can be created with `wget` together with `rclone`:

1. Download all the files from the official mirror:

   ```
   # https://stackoverflow.com/questions/29802579/create-private-mirror-of-http-nodejs-org-dist
   wget --mirror --convert-links --adjust-extension --page-requisites --no-parent --robots=off http://nodejs.org/dist
   ```

2. Upload all the files to S3:

   ```
   # https://rclone.org/s3/
   rclone sync -i nodejs.org s3:languages
   ```

Note: Downloading a copy of the official mirror took 15m and 52Gb.

How local development will work with the new approach?

Local development will require scripts to clone the official mirrors for each language and upload them to MinIO (S3). Besides, a script to define a set of Python version, pre-compile them and also upload them to S3.

This is already covered by this simple bash script and tested in this PR with a POC: https://github.com/readthedocs/readthedocs.org/pull/8453

Deprecation plan

After this design document gets implemented and tested, all our current images (stable, latest, testing) will be deprecated and their usage will be de-motivated. However, we could keep them on our builders to give users a good time to migrate their projects to the new ones.

We may want to keep only the latest Ubuntu LTS release available in production, with a special consideration for our current Ubuntu 18.04 LTS on stable, latest and testing because 100% of the projects depend on them currently. Once Ubuntu 22.04 LTS is released, we should communicate that Ubuntu 20.04 LTS is deprecated, and keep it available in our servers during the time that’s officially supported by Ubuntu during the “Maintenance updates” (see “Login term support and interim releases” in https://ubuntu.com/about/release-cycle). As an example, Ubuntu 22.04 LTS will be officially released on April 2022 and we will offer support for it until 2027.

Warning: Deleting -base images from the build servers will make project’s builds to fail. We want to keep supporting them as much as we can, but having a well-defined deprecation policy is a win.
Work required and rollout plan

The following steps are required to support the full proposal of this document.

1. allow users to install extras languages requirements via config file
   • update config file to support `build.os` and `build.languages` config
   • modify builder code to run `asdf install` for all supported languages

2. build a new base Docker image with new structure (`ubuntu20-base`)
   • build new image with Ubuntu 20.04 LTS and pre-installed `asdf` with all its plugins
   • do not install any language version on base image
   • deploy builders with new base image

At this point, we will have a full working setup. It will be opt-in by using the new configs `build.os` and `build.languages`. However, all languages will be installed at build time; which will “penalize” all projects because all of them will have to install Python.

After testing this for some time, we can continue with the following steps that provides a cache to optimize installation times:

1. create mirrors on S3 for all supported languages
2. create mirror for pre-compiled latest 3 Python versions, Python 2.7 and PyPy3

Conclusion

There is no need to differentiate the images by its state (stable, latest, testing) but by its main base differences: OS. The version of the OS will change many library versions, LaTeX dependencies, basic required commands like git and more, that doesn't seem to be useful to have the same OS version with different states.

Allowing users to install extra languages by using the Config File will cover most of the support requests we have had in the past. It also will allow us to know more about how our users are using the platform to make future decisions based on this data. Exposing users how we want them to use our platform will allow us to be able to maintain it longer, than giving the option to select a specific Docker image by name that we can't guarantee it will be frozen.

Finally, having the ability to deprecate and remove pre-built images from our builders over time, will reduce the maintenance work required from the the core team. We can always support all the languages versions by installing them at build time. The only required pre-built image for this are the OS `--base` images. In fact, even after decided to deprecate and removed a pre-built image from the builders, we can re-build it if we find that it’s affecting many projects and slowing down their builds too much, causing us problems.

Embed APIv3

The Embed API allows users to embed content from documentation pages in other sites. It has been treated as an experimental feature without public documentation or real applications, but recently it started to be used widely (mainly because we created the `hoverxref` Sphinx extension).

The main goal of this document is to design a new version of the Embed API to be more user friendly, make it more stable over time, support embedding content from pages not hosted at Read the Docs, and remove some quirksness that makes it hard to maintain and difficult to use.

Note: This work is part of the CZI grant that Read the Docs received.
Current implementation

The current implementation of the API is partially documented in *Embedding Content From Your Documentation*. It has some known problems:

- There are different ways of querying the API: ?url= (generic) and ?doc= (relies on Sphinx’s specific concept)
- Doesn’t support MkDocs
- Lookups are slow (~500 ms)
- IDs returned aren’t well formed (like empty IDs "headers": ["title": "#"])
- The content is always an array of one element
- It tries different variations of the original ID
- It doesn’t return valid HTML for definition lists (dd tags without a dt tag)

Goals

We plan to add new features and define a contract that works the same for all HTML. This project has the following goals:

- Support embedding content from pages hosted outside Read the Docs
- Do not depend on Sphinx .json files
- Query and parse the .html file directly (from our storage or from an external request)
- Rewrite all links returned in the content to make them absolute
- Require a valid HTML id selector
- Accept only ?url= request GET argument to query the endpoint
- Support ?nwords= and ?nparagraphs= to return chunked content
- Handle special cases for particular doctools (e.g. Sphinx requires to return the .parent() element for dl)
- Make explicit the client is asking to handle the special cases (e.g. send ?doctool=sphinx&version=4.0.1&writer=html4)
- Delete HTML tags from the original document (for well-defined special cases)
• Add HTTP cache headers to cache responses
• Allow CORS from everywhere only for public projects

The contract

Return the HTML tag (and its children) with the id selector requested and replace all the relative links from its content making them absolute.

Note: Any other case outside this contract will be considered special and will be implemented only under ?doctool=, ?version= and ?writer= arguments.

If no id selector is sent to the request, the content of the first meaningful HTML tag (<main>, <div role="main"> or other well-defined standard tags) identifier found is returned.

Embed endpoints

This is the list of endpoints to be implemented in APIv3:

GET /api/v3/embed/
Returns the exact HTML content for a specific identifier (id). If no anchor identifier is specified the content of the first one returned.

Example request:


Example response:

```
{
    "project": "docs",
    "version": "latest",
    "language": "en",
    "path": "development/install.html",
    "title": "Development Installation",
    "url": "https://docs.readthedocs.io/en/latest/development/install.html#set-up-your-environment",
    "id": "set-up-your-environment",
    "content": "<div class="section" id="development-installation">
<h1>
 Development Installation<a class="headerlink" href="https://docs.readthedocs.io/en/stable/development/install.html#development-installation" title="Permalink to this headline">¶</a></h1>
 ...

```

Query Parameters

• (required) (url) – Full URL for the documentation page with optional anchor identifier.

GET /api/v3/embed/metadata/
Returns all the available metadata for an specific page.
Note: As it’s not trivial to get the title associated with a particular id and it’s not easy to get a nested list of identifiers, we may not implement this endpoint in initial version.

The endpoint as-is, is mainly useful to explore/discover what are the identifiers available for a particular page – which is handy in the development process of a new tool that consumes the API. Because of this, we don’t have too much traction to add it in the initial version.

Example request:

```
```

Example response:

```
{
   "identifiers": {
      "id": "set-up-your-environment",
      "url": "https://docs.readthedocs.io/en/latest/development/install.html#set-up-your-environment"
      "_links": {
         "embed": "https://docs.readthedocs.io/_/api/v3/embed/?url=https://docs.readthedocs.io/en/latest/development/install.html#set-up-your-environment"
      }
   },
   {
      "id": "check-that-everything-works",
      "url": "https://docs.readthedocs.io/en/latest/development/install.html#check-that-everything-works"
      "_links": {
         "embed": "https://docs.readthedocs.io/_/api/v3/embed/?url=https://docs.readthedocs.io/en/latest/development/install.html#check-that-everything-works"
      }
   }
}
```

Query Parameters

- **(required) (url)** – Full URL for the documentation page

Handle specific Sphinx cases

We are currently handling some special cases for Sphinx due how it writes the HTML output structure. In some cases, we look for the HTML tag with the identifier requested but we return the `.next()` HTML tag or the `.parent()` tag instead of the requested one.

Currently, we have identified that this happens for definition tags (`dl`, `dt`, `dd`) – but may be other cases we don’t know yet. Sphinx adds the `id=` attribute to the `dt` tag, which contains only the title of the definition, but as a user, we are expecting the description of it.

In the following example we will return the whole `dl` HTML tag instead of the HTML tag with the identifier `id="term-name"` as requested by the client, because otherwise the “Term definition for Term Name” content won’t be included and the response would be useless.
If the definition list (dl) has more than one definition it will return only the term requested. Considering the following example, with the request ?url=glossary.html#term-name

It will return the whole dl with only the dt and dd for id requested:

However, this assumptions may not apply to documentation pages built with a different doctool than Sphinx. For this reason, we need to communicate to the API that we want to handle this special cases in the backend. This will be done by appending a request GET argument to the Embed API endpoint: ?doctool=sphinx&version=4.0.1&writer=html4. In this case, the backend will known that has to deal with these special cases.

Note: This leaves the door open to be able to support more special cases (e.g. for other doctools) without breaking the actual behavior.

**Support for external documents**

When the ?url= argument passed belongs to a documentation page not hosted on Read the Docs, the endpoint will do an external request to download the HTML file, parse it and return the content for the identifier requested.

The whole logic should be the same, the only difference would be where the source HTML comes from.

Warning: We should be careful with the URL received from the user because those may be internal URLs and we could be leaking some data. Example: ?url=http://localhost/some-weird-endpoint or ?url=http://169.254.169.254/latest/meta-data/ (see https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instancetype-data-retrieval.html).

This is related to SSRF (https://en.wikipedia.org/wiki/Server-side_request_forgery). It doesn’t seem to be a huge problem, but something to consider.

Also, the endpoint may need to limit the requests per-external domain to avoid using our servers to take down another site.
Note: Due to the potential security issues mentioned, we will start with an allowed list of domains for common Sphinx docs projects. Projects like Django and Python, where `sphinx-hoverxref` users might commonly want to embed from. We aren't planning to allow arbitrary HTML from any website.

Handle project's domain changes

The proposed Embed APIv3 implementation only allows ?url= argument to embed content from that page. That URL can be:

- a URL for a project hosted under `<project-slug>.readthedocs.io`
- a URL for a project with a custom domain

In the first case, we can easily get the project's slug directly from the URL. However, in the second case we get the project's slug by querying our database for a `Domain` object with the full domain from the URL.

Now, consider that all the links in the documentation page that uses Embed APIv3 are pointing to `docs.example.com` and the author decides to change the domain to be `docs.newdomain.com`. At this point there are different possible scenarios:

- The user creates a new `Domain` object with `docs.newdomain.com` as domain's name. In this case, old links will keep working because we still have the old `Domain` object in our database and we can use it to get the project's slug.
- The user deletes the old `Domain` besides creating the new one. In this scenario, our query for a `Domain` with name `docs.example.com` to our database will fail. We will need to do a request to `docs.example.com` and check for a 3xx response status code and in that case, we can read the `Location:` HTTP header to find the new domain's name for the documentation. Once we have the new domain from the redirect response, we can query our database again to find out the project's slug.

Note: We will follow up to 5 redirects to find out the project's domain.

Embed APIv2 deprecation

The v2 is currently widely used by projects using the `sphinx-hoverxref` extension. Because of that, we need to keep supporting it as-is for a long time.

Next steps on this direction should be:

- Add a note in the documentation mentioning this endpoint is deprecated
- Promote the usage of the new Embed APIv3
- Migrate the `sphinx-hoverxref` extension to use the new endpoint

Once we have done them, we could check our NGINX logs to find out if there are people still using APIv2, contact them and let them know that they have some months to migrate since the endpoint is deprecated and will be removed.
Unanswered questions

- How do we distinguish between our APIv3 for resources (models in the database) from these “feature API endpoints”?

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. 2: 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 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 its 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

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

Proposed contents for new Sphinx guides

Note: This work is in progress, see discussion on this Sphinx issue and the pull requests linked at the end.

The two main objectives are:

- Contributing a good Sphinx tutorial for beginners. This should introduce the readers to all the various Sphinx major features in a pedagogical way, and be mostly focused on Markdown using MyST. We would try to find a place for it in the official Sphinx documentation.
- Write a new narrative tutorial for Read the Docs that complements the existing guides and offers a cohesive story of how to use the service.

Sphinx tutorial

Appendixes are optional, i.e. not required to follow the tutorial, but highly recommended.

1. The Sphinx way
   - Preliminary section giving an overview of what Sphinx is, how it works, how reStructuredText and Markdown/MyST are related to it, some terminology (toctree, builders), what can be done with it.

2. About this tutorial
   - A section explaining the approach of the tutorial, as well as how to download the result of each section for closer inspection or for skipping parts of it.

3. Getting started
   1. Creating our project
      - Present a fictitious goal for a documentation project
      - Create a blank README.md to introduce the most basic elements of Markdown (headings and paragraph text)
   2. Installing Sphinx and cookiecutter in a new development environment
      - Install Python (or miniforge)
      - Create a virtual environment (and/or conda environment)
      - Activate our virtual environment (it will always be the first step)
      - Install Sphinx inside the virtual environment
      - Check that sphinx-build --help works (yay!)
   3. Creating the documentation layout
• Apply our cookiecutter to create a minimal docs/ directory (similar to what sphinx-quickstart does, but with source and build separation by default, project release 0.1, English language, and a MyST index, if at all)\(^1\)

• Check that the correct files are created (yay!)

4. Appendix: Using version control

• Install git (we will not use it during the tutorial)
• Add a proper .gitignore file (copied from gitignore.io)
• Create the first commit for the project (yay!)

4. First steps to document our project using Sphinx

1. Converting our documentation to local HTML

• Create (or minimally tweak) index.md
• Build the HTML output using sphinx-build -b html doc docs/_build/html\(^2\)
• Navigate to docs/_build/html and launch an HTTP server (python -m http.server)
• Open http://localhost:8000 in a web browser, and see the HTML documentation (yay!)

2. Converting our documentation to other formats

• Build PseudoXML using make pseudoxml
• Build Text using make text
• See how the various formats change the output (yay!)

3. Appendix: Simplify documentation building by using Make\(^3\)

• Install Make (nothing is needed on Windows, make.bat is standalone)
• Add more content to index.md
• Build HTML doing cd doc && make html
• Observe that the HTML docs have changed (yay!)

4. Appendix: PDF without LaTeX using rinoh (beta)

5. Customizing Sphinx configuration

1. Changing the HTML theme

• Install https://pypi.org/project/furo/
• Change the html_theme in conf.py
• Rebuild the HTML documentation and observe that the theme has changed (yay!)

2. Changing the PDF appearance

• Add a latex_theme and set it to howto
• Rebuild make latexpdf
• Check that the appearance changed (yay!)

3. Enable an extension

\(^1\) Similar to https://github.com/sphinx-contrib/cookiecutter, but only for the docs/ directory? This way it can be less opinionated about everything else
\(^2\) At first I considered “make mode“, but the current maintainers don’t know much about its original intent (see my comment here and the discussion after it)
\(^3\) There have been attempts at creating a sphinx command, see this pull request
• Add a string to the extensions list in conf.py for sphinx.ext.duration
• Rebuild the HTML docs make html and notice that now the times are printed (yay!)

6. Writing narrative documentation with Sphinx

• First focus on index.md, diving more into Markdown and mentioning Semantic Line Breaks.
• Then add another .md file to teach how toctree works.
• Then continue introducing elements of the syntax to add pictures, cross-references, and the like.

7. Describing code in Sphinx

• Explain the Python domain as part of narrative documentation to interleave code with text, include doctests, and justify the usefulness of the next section.

8. Autogenerating documentation from code in Sphinx

9. Deploying a Sphinx project online

• A bit of background on the options: GitHub/GitLab Pages, custom server, Netlify, Read the Docs
• Make reference to Read the Docs tutorial

10. Appendix: Using Jupyter notebooks inside Sphinx

11. Appendix: Understanding the docutils document tree

12. Appendix: Where to go from here

• Refer the user to the Sphinx, reST and MyST references, prominent projects already using Sphinx, compilations of themes and extensions, the development documentation.

Read the Docs tutorial

1. The Read the Docs way

2. Getting started

   1. Preparing our project on GitHub
      • Fork a starter GitHub repository (something like our demo template, as a starting point that helps mimicking the sphinx-quickstart or cookiecutter step without having to checkout the code locally)

   2. Importing our project to Read the Docs
      • Sign up with GitHub on RTD
      • Import the project (don’t “Edit advanced project options”, we will do this later)
      • The project is created on RTD
      • Browse “builds”, open the build live logs, wait a couple of minutes, open the docs (yay!)

   3. Basic configuration changes
      • Add a description, homepage, and tags
      • Configure your email for build failure notification (until we turn them on by default)
      • Enable “build pull requests for this project” in the advanced settings
      • Edit a file from the GitHub UI as part of a new branch, and open a pull request
      • See the RTD check on the GitHub PR UI, wait a few minutes, open result (yay!)
3. Customizing the build process
   • Use `readthedocs.yaml` (rather than the web UI) to customize build formats, change build requirements and Python version, enable fail-on-warnings

4. Versioning documentation
   • Explain how to manage versions on RTD: create release branches, activate the corresponding version, browse them in the version selector, selectively build versions
   • Intermediate topics: hide versions, create Automation Rules

5. Getting insights from your projects
   • Move around the project, explore results in Traffic Analytics
   • Play around with server-side search, explore results in Search Analytics

6. Managing translations

7. Where to go from here
   • Reference our existing guides, prominent projects already using RTD, domain configuration, our support form, our contributing documentation

Possible new how-to Guides

Some ideas for extra guides on specific topics, still for beginners but more problem-oriented documents, covering a wide range of use cases:
   • How to turn a bunch of Markdown files into a Sphinx project
   • How to turn a bunch of Jupyter notebooks into a Sphinx project
   • How to localize an existing Sphinx project
   • How to customize the appearance of the HTML output of a Sphinx project
   • How to convert existing reStructuredText documentation to Markdown
   • How to use Doxygen autogenerated documentation inside a Sphinx project
   • How to keep a changelog of your project

Reference

All the references should be external: the Sphinx reference, the MyST and reST syntax specs, and so forth.

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 me 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 querysets 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. i.e: `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 Managers 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 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

Privacy Levels

This document describes how to handle and unify privacy levels on the community and commercial version of Read the Docs.

Current state

Currently, we have three privacy levels for projects and versions:

1. Public
2. Private
3. Protected (currently hidden)

These levels of privacy aren’t clear and bring confusion to our users. Also, the private level doesn’t make sense on the community site, since we only support public projects.

Places where we use the privacy levels are:

- On serving docs
- Footer
- Dashboard

Project level privacy

Project level privacy was meant to control the dashboard visibility.

This privacy level brings to confusion when users want to make a version public. We should remove all the project privacy levels.

For the community site the dashboard would be always visible, and for the commercial site, the dashboard would be always hidden.

The project privacy level is also used to serve the 404.html page, show robots.txt, and show sitemap.xml. The privacy level from versions should be used instead.

Some other ideas about keeping the privacy level is to dictate the default version level of new versions, but removing all other logic related to this privacy level. This can be (or is going to be) possible with automation rules, so we can just remove the field.
Version level privacy

Version level privacy is mainly used to restrict access to documentation. For public level, everyone can access to the documentation. For private level, only users that are maintainers or that belong to a team with access (for the commercial site) can access to the documentation.

The protected privacy level was meant to hide versions from listings and search. For the community site these versions are treated like public versions, and on the commercial site they are treated like private.

The protected privacy level is currently hidden. To keep the behavior of hiding versions from listings and search, a new field should be added to the Version model and forms: hidden (#5321). The privacy level (public or private) would be respected to determine access to the documentation.

For the community site, the privacy level would be public and can't be changed.

The default privacy level of new versions for the commercial site would be private (this is the DEFAULT_PRIVACY_LEVEL setting).

Footer

The footer is used to display not hidden versions that the current user has access to.

For the community site no changes are required on the footer.

For the commercial site we use the project level privacy to decide if show or not links to the project's dashboard: downloads, project home, and builds. Given that the project privacy level would be removed (and the dashboard is always under login), those links would never be shown, except for admin users (owners or from a team with admin access) since they are the only ones allowed to make changes on the project.

Overview

For the community site:

• The project’s dashboard is visible to all users.
• All versions are always public.
• The footer shows links to the project’s dashboard (build, downloads, home) to all users.
• Only versions with hidden = False are listed on the footer and appear on search results.
• If a project has a 404.html file on the default version, it’s served.
• If a project has a robots.txt file on the default version, it’s served.
• A sitemap.xml file is always served.

For the commercial site:

• The project’s dashboard is visible to only users that have read permission over the project.
• The footer shows links to the project’s dashboard (build, downloads, home) to only admin users.
• Only versions with hidden = False are listed on the footer and appear on search results.
• If a project has a 404.html file on the default version, it’s served if the user has permission over that version.
• If a project has a robots.txt file on the default version, it’s served if the user has permission over that version.
• A sitemap.xml file is served if the user has at least one public version. And it will only list public versions.
Migration

To differentiate between allowing or not privacy levels, we need to add a setting `RTD_ALLOW_PRIVACY_LEVELS` (False by default).

For the community and commercial site, we need to:

- Remove/change code that depends on the project’s privacy level. Use the global setting `RTD_ALLOW_PRIVACY_LEVELS` and default version’s privacy level instead.
  - Display robots.txt
  - Serve 404.html page
  - Display sitemap.xml
  - Querysets
- Remove `Project.privacy_level` field
- Migrate all protected versions to have the attribute `hidden = True` (data migration), and set their privacy level to public for the community site and private for the commercial site.
- Change all querysets used to list versions on the footer and on search to use the `hidden` attribute.
- Update docs

For the community site:

- Hide all privacy level related settings from the version form.
- Don’t expose privacy levels on API v3.
- Mark all versions as public.

For the commercial site:

- Always hide the dashboard
- Show links to the dashboard (downloads, builds, project home) on the footer only to admin users.

Upgrade path overview

Community site

The default privacy level for the community site is public for versions and the dashboard is always public.

Public project (community)

- Public version: Normal use case, no changes required.
- Protected version: Users didn’t want to list this version on the footer, but also not deactivate it. We can do a data migration of those versions to the new `hidden` setting and make them public.
- Private version: Users didn’t want to show this version to their users yet or they were testing something. This can be solved with the pull request builder feature and the `hidden` setting. We migrate those to public with the `hidden` setting. If we are worried about leaking anything from the version, we can email users before doing the change.
Protected project (community)

Protected projects are not listed publicly. Probably users were hosting a WIP project, or personal public project. A public project should work for them, as we are removing listing all projects publicly (except for search).

The migration path for versions of protected projects is the same as a public project.

Private project (community)

Probably these users want to use our enterprise solution instead. Or they were hosting a personal project.

The migration path for versions of private projects is the same as a public project.

If we are worried about leaking anything from the dashboard or build page, we can email users before doing the change.

Commercial site

The default privacy level for the commercial site is private for versions and the dashboard is show only to admin users.

Private project (commercial)

- Private version: Normal use case, not changes required.
- Protected version: Users didn’t want to list this version on the footer, but also not deactivate it. This can be solved by using the new hidden setting. We can do a data migration of those versions to the new hidden setting and make them private.
- Public version: User has private code, but want to make public their docs. No changes required.

Protected project (commercial)

I can’t think of a use case for protected projects, since they aren’t listed publicly on the commercial site.

The migration path for versions of protected projects is the same as a private project.

Public project (commercial)

Currently we show links back to project dashboard if the project is public, which probably users shouldn’t see. With the implementation of this design doc, public versions don’t have links to the project dashboard (except for admin users) and the dashboard is always under login.

- Private versions: Users under the organization can see links to the dashboard. Not changes required.
- Protected versions: Users under the organization can see links to the dashboard. We can do a data migration of those versions to the new hidden setting and make them private.
- Public versions: All users can see links to the dashboard. Probably they have an open source project, but they still want to manage access using the same teams of the organization. Not changes are required.

A breaking change here is: users outside the organization would not be able to see the dashboard of the project.
Refactor RemoteRepository object

This document describes the current usage of RemoteRepository objects and proposes a new normalized modeling.

Goals

- De-duplicate data stored in our database.
- Save only one RemoteRepository per GitHub repository.
- Use an intermediate table between RemoteRepository and User to store associated remote data for the specific user.
- Make this model usable from our SSO implementation (adding remote_id field in Remote objects).
- Use Post JSONField to store associated json remote data.
- Make Project connect directly to RemoteRepository without being linked to a specific User.
- Do not disconnect Project and RemoteRepository when a user delete/disconnects their account.

Non-Goals

- Keep RemoteRepository in sync with GitHub repositories.
- Delete RemoteRepository objects deleted from GitHub.
- Listen to GitHub events to detect full_name changes and update our objects.

Note: We may need/want some of these non-goals in the future. They are just outside the scope of this document.

Current Implementation

When a user connect their account to a social account, we create a

- allauth.socialaccount.models.SocialAccount * basic information (provider, last login, etc) * provider's specific data saved in a JSON under extra_data
- allauthsocialaccount.models.SocialToken * token to hit the API on behalf the user

We *don't create* any RemoteRepository at this point. They are created when the user jumps into “Import Project” page and hit the circled arrows. It triggers sync_remote_repositories task in background that updates or creates RemoteRepositories, but *it does not delete* them (after #7183 and #7310 got merged, they will be deleted). One RemoteRepository is created per repository the User has access to.

Note: In corporate, we are automatically syncing RemoteRepository and RemoteOganization at signup (foreground) and login (background) via a signal. We should eventually move these to community.
**Where RemoteRepository is used?**

- List of available repositories to import under “Import Project”
- Show a “+”, “External Arrow” or a “Lock” sign next to the element in the list * +: it’s available to be imported * External Arrow: the repository is already imported (see RemoteRepository.matches method) * Lock: user doesn’t have (admin) permissions to import this repository (uses RemoteRepository.private and RemoteRepository.admin)
- Avatar URL in the list of project available to import
- Update webhook when user clicks “Resync webhook” from the Admin > Integrations tab
- Send build status when building Pull Requests

**New Normalized Implementation**

The ManyToMany relation RemoteRepository.users will be changed to be ManyToMany(through='RemoteRelation') to add extra fields in the relation that are specific only for the User. Allows us to have only one RemoteRepository per GitHub repository with multiple relationships to User.

With this modeling, we can avoid the disconnection Project and RemoteRepository only by removing the RemoteRelation.

**Note:** All the points mentioned in the previous section may need to be adapted to use the new normalized modeling. However, it may be only field renaming or small query changes over new fields.

**Use this modeling for SSO**

We can get the list of Project where a user as access:

```python
admin_remoteRepositories = RemoteRepository.objects.filter(
    users__contains=request.user,
    users__remoterelation__admin=True, # False for read-only access
)
Project.objects.filter(remote_repository__in=admin_remoteRepositories)
```

**Rollout Plan**

Due the constraints we have in the RemoteRepository table and its size, we can’t just do the data migration at the same time of the deploy. Because of this we need to be more creative here and find a way to re-sync the data from VCS providers, while the site continue working.

To achieve this, we thought on following these steps:

1. modify all the Python code to use the new modeling in .org and .com (will help us to find out bugs locally in an easier way) 1. QA this locally with test data 1. enable Django signal to re-sync RemoteRepository on login async (we already have this in .com). New active users will have updated data immediately 1. spin up a new instance with the new refactored code 1. run migrations to create a new table for RemoteRepository 1. re-sync everything from VCS providers into the new table for 1-week or so 1. dump-n-load Project - RemoteRepository relations 1. create a migration to use the new table with synced data 1. deploy new code once the sync is finished
Allow Installation of System Packages

Currently we don’t allow executing arbitrary commands in the build process. The more common use case is to install extra dependencies.

- **Current status**
- **Security concerns**
- **Exposing apt install**
- **Using docker exec**
- **Config file**
- **Possible problems**
- **Other possible solutions**

**Current status**

There is a workaround when using Sphinx to run arbitrary commands, this is executing the commands inside the `conf.py` file. There isn’t a workaround for MkDocs, but this problem is more common in Sphinx, since users need to install some extra dependencies in order to use autodoc or build Jupyter Notebooks.

However, installation of some dependencies require root access, or are easier to install using `apt`. Most of the CI services allow to use `apt` or execute any command with `sudo`, so users are more familiar with that workflow.

Some users use Conda instead of pip to install dependencies in order to avoid these problems, but not all pip users are familiar with Conda, or want to migrate to Conda just to use Read the Docs.

**Security concerns**

Builds are run in a Docker container, but the app controlling that container lives in the same server. Allowing to execute arbitrary commands with super user privileges may introduce some security issues.

**Exposing apt install**

For the previous reasons we won’t allow to execute arbitrary commands with root (yet), but instead allow only to install extra packages using `apt`.

We would expose this through the config file. Users will provide a list of packages to install, and under the hook we would run:

- `apt update -y`
- `apt install -y {packages}`

These commands will be run before the Python setup step and after the clone step.
Note: All package names must be validated to avoid injection of extra options (like -v).

Using docker exec

Currently we use docker exec to execute commands in a running container. This command also allows to pass a user which is used to run the commands (#8058_). We can run the apt commands in our current containers using a super user momentarily.

Config file

The config file can add an additional mapping build.apt_packages to a list of packages to install.

```yaml
version: 2
build:
    apt_packages:
        - cmatrix
        - mysql-server
```

Note: Other names that were considered were:

- build.packages
- build.extra_packages
- build.system_packages

These were rejected to avoid confusion with existing keys, and to be explicit about the type of package.

Possible problems

- Some users may require to pass some additional flags or install from a ppa.
- Some packages may require some additional setup after installation.

Other possible solutions

- We can allow to run the containers as root doing something similar to what Travis does: They have one tool to convert the config file to a shell script (travis-build), and another that spins a docker container, executes that shell script and streams the logs back (travis-worker).
- A similar solution could be implemented using AWS Lambda.

This of course would require a large amount of work, but may be useful for the future.
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`:

```python
{
    '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'
                }],
            },
            '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,
                }],
            },
        }
    }
}
```

(continues on next page)
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 }}{{ 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 removed\(^1\))
- Project description (this feature is planned to be removed\(^2\))
- Single version
- Default branch
- Default version
- Domains
- Active versions
- Translations
- Subprojects
- Integrations
- Notifications
- Language
- Programming Language
- Project homepage
- Tags
- Analytics code

\(^1\) [https://github.com/readthedocs/readthedocs.org/issues/2663](https://github.com/readthedocs/readthedocs.org/issues/2663)
\(^2\) [https://github.com/readthedocs/readthedocs.org/issues/3689](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 1.2\(^3\) (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

\(^3\) https://yaml.org/spec/1.2/spec.html
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.

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

\(^4\) https://github.com/readthedocs/readthedocs.org/issues/3806
5.3 Roadmap

5.3.1 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.

5.3.2 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.

5.4 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.

5.4.1 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!
5.4.2 Getting Started

The *Development 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.

5.4.3 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.

5.4.4 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!

5.4.5 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:

- Spell Check (https://pypi.python.org/pypi/sphinxcontrib-spelling/)

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.

**5.4.6 Projects from previous years**

- Improved Search And Search As You Type (2019)
- Building Docs For Pull Requests (2019)
- Search Improvement (2018)

**5.4.7 Thanks**

This page was heavily inspired by Mailman’s similar GSOC page. Thanks for the inspiration.
5.5 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 live chat, 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, color, 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.

5.6 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.

5.6.1 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.

5.6.2 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.
5.6.3 PGP key

You may use this PGP key to securely communicate with us and to verify signed messages you receive from us.

5.6.4 Security issue archive

Version 5.19.0

Version 5.19.0 fixes an issue that allowed a malicious user to fetch internal and private information from a logged user in readthedocs.org/readthedocs.com by creating a malicious site hosted on readthedocs.io/readthedocs-hosted.com or from any custom domain registered in the platform.

It would have required the attacker to get a logged in user to visit an attacker controlled web page, which could then have made GET API requests on behalf of the user. This vulnerability was found by our team as part of a routine security audit, and there is no indication it was exploited.

The issue was found by the Read the Docs team.

Version 5.14.0

Version 5.14.0 fixes an issue where that affected new code that removed multiple slashes in URL paths. 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](https://docs.readthedocs.io/en/stable/custom_domains.html)) but instead went to a different domain.

This issue was reported by Splunk after it was reported by a security audit.

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 desec.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.
5.7 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.

5.7.1 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, you 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

5.7.2 Request Archive

Currently, Read the Docs has not received any takedown requests.

5.8 Policy for Abandoned Projects

This policy describes the process by which a Read the Docs project name may be changed.

5.8.1 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.
5.8.2 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

5.8.3 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.

**5.8.4 Prior art**

The Python Package Index (PyPI) policy for claiming abandoned packages ([PEP-0541](https://www.python.org/dev/peps/pep-0541/)) heavily influenced this policy.

**5.9 Changelog**

**5.9.1 Version 6.3.0**

**Date** November 29, 2021

• @humitos: Tests: run tests with Python3.8 in CircleCI ([#8718](https://github.com/readthedocs/readthedocs.org/pull/8718))
• @stsewd: Test external serving for projects with `-- in slug` ([#8716](https://github.com/readthedocs/readthedocs.org/pull/8716))
• @humitos: requirements: add requests-oauthlib ([#8712](https://github.com/readthedocs/readthedocs.org/pull/8712))
• @astrojuanlu: Avoid future breakage of `setup.py` invokations ([#8711](https://github.com/readthedocs/readthedocs.org/pull/8711))
• @humitos: spam: fix admin filter ([#8707](https://github.com/readthedocs/readthedocs.org/pull/8707))
• @humitos: oauth: sync remote repositories fix ([#8706](https://github.com/readthedocs/readthedocs.org/pull/8706))
• @humitos: structlog: migrate application code to better logging ([#8705](https://github.com/readthedocs/readthedocs.org/pull/8705))
• @astrojuanlu: Add guide on Poetry ([#8702](https://github.com/readthedocs/readthedocs.org/pull/8702))
• @humitos: EmbedAPI: log success requests ([#8689](https://github.com/readthedocs/readthedocs.org/pull/8689))

**5.9.2 Version 6.2.1**

**Date** November 23, 2021

• @agjohnson: Fix issue with PR build hostname parsing ([#8700](https://github.com/readthedocs/readthedocs.org/pull/8700))
• @ericholscher: Fix sharing titles ([#8695](https://github.com/readthedocs/readthedocs.org/pull/8695))
• @stsewd: Downgrade celery and friends ([#8693](https://github.com/readthedocs/readthedocs.org/pull/8693))
• @stsewd: Downgrade pyyaml ([#8691](https://github.com/readthedocs/readthedocs.org/pull/8691))
• @stsewd: Downgrade redis ([#8690](https://github.com/readthedocs/readthedocs.org/pull/8690))
• @humitos: Spam: make admin filters easier to understand ([#8688](https://github.com/readthedocs/readthedocs.org/pull/8688))
• @astrojuanlu: Clarify how to pin the Sphinx version ([#8687](https://github.com/readthedocs/readthedocs.org/pull/8687))
• @stsewd: Webhooks: fix link to docs ([#8685](https://github.com/readthedocs/readthedocs.org/pull/8685))
• @stsewd: Docs: update docs about search on subprojects ([#8683](https://github.com/readthedocs/readthedocs.org/pull/8683))
• @stsewd: Update common ([#8681](https://github.com/readthedocs/readthedocs.org/pull/8681))
• @pyup-bot: pyup: Scheduled weekly dependency update for week 46 (#8680)

5.9.3 Version 6.2.0

Date November 16, 2021
• @rokroskar: docs: update faq to mention apt for dependencies (#8676)
• @stsewd: UserProfile: add time fields (#8675)
• @stsewd: Admin: don’t use update to ban users (#8674)
• @stsewd: UserProfile: add historical model (#8673)
• @astrojuanlu: Add entry on Jupyter Book to the FAQ (#8669)
• @stsewd: Proxito: add CDN-Cache-Control headers (#8667)
• @humitos: Spam: sort admin filters and show threshold (#8666)
• @humitos: Cleanup: remove old code (#8665)
• @humitos: Spam: check for spam rules after user is banned (#8664)
• @humitos: Spam: use 410 - Gone status code when blocked (#8661)
• @astrojuanlu: Upgrade readthedocs-sphinx-search (#8660)
• @agjohnson: Release 6.1.2 (#8657)
• @astrojuanlu: Update requirements pinning (#8655)
• @stsewd: Historical records: set the change reason explicitly on the instance (#8627)
• @stsewd: Support for generic webhooks (#8522)

5.9.4 Version 6.1.2

Date November 08, 2021
• @astrojuanlu: Update requirements pinning (#8655)
• @ericholscher: Fix GitHub permissions required (#8654)
• @stsewd: Organizations: allow to add owners by email (#8651)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 44 (#8645)
• @humitos: Spam: use thresholds to decide actions (#8632)
• @astrojuanlu: Document generic webhooks (#8609)
• @stsewd: Organizations: show audit logs (#8588)
5.9.5 Version 6.1.1

Date November 02, 2021

- @agjohnson: Drop beta from title of build config option (#8637)
- @astrojuanlu: Remove mentions to old Python version specification (#8635)
- @jugmac00: fix typos (#8630)
- @Arthur-Milchior: Correct an example (#8628)
- @davidfischer: Inherit theme template (#8626)
- @astrojuanlu: Clarify duration of extra DNS records (#8625)
- @astrojuanlu: Promote mamba more in the documentation, hide CONDA_USES_MAMBA (#8624)
- @davidfischer: Floating ad placement for docs.readthedocs.io (#8621)
- @stsewd: Audit: track downloads separately from page views (#8619)
- @humitos: Celery: quick hotfix to task (#8617)

5.9.6 Version 6.1.0

Date October 26, 2021

- @astrojuanlu: Clarify docs (#8608)
- @stsewd: Move more organization tests (#8606)
- @astrojuanlu: New Read the Docs tutorial, part III (and finale?) (#8605)
- @humitos: SSO: re-sync VCS accounts for SSO organization daily (#8601)
- @humitos: Django Action: re-sync SSO organization’s users (#8600)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 42 (#8598)
- @saadmk11: Don’t show the rebuild option on closed/merged Pull Request builds (#8590)
- @carltongibson: Adjust Django intersphinx link to stable version. (#8589)
- @humitos: Small fixes to asdf image upload script (#8587)
- @astrojuanlu: Documentation names cleanup (#8586)
- @stsewd: Fix docs (#8584)
- @adamtheturtle: Fix typo “interpreters” (#8583)
- @ericholscher: Small fixes to asdf image upload script (#8578)
- @stsewd: Move organization views (#8577)
- @humitos: EmbedAPIv3: docs for endpoint and guide updated (#8566)
- @stsewd: GitLab integration: escape path (#8525)
- @stsewd: Domain: allow to disable domain creation/update (#8020)
5.9.7 Version 6.0.0

**Date** October 13, 2021

This release includes the upgrade of some base dependencies:

- Python version from 3.6 to 3.8
- Ubuntu version from 18.04 LTS to 20.04 LTS

Starting from this release, all the Read the Docs code will be tested and QAed on these versions.

- @ericholscher: Release 5.25.1 (#8576)
- @deepto98: Moved authenticated_classes definitions from API classes to AuthenticatedClassesMixin (#8562)
- @humitos: Upgrade to Ubuntu 20.04 and Python 3.8 (#7421)

5.9.8 Version 5.25.1

**Date** October 11, 2021

- @astrojuanlu: Small fixes (#8564)
- @deepto98: Moved authenticated_classes definitions from API classes to AuthenticatedClassesMixin (#8562)
- @humitos: Build: update ca-certificates before cloning (#8559)
- @humitos: Build: support Python 3.10.0 stable release (#8558)
- @astrojuanlu: Document new build specification (#8547)
- @astrojuanlu: Add checkbox to subscribe new users to newsletter (#8546)

5.9.9 Version 5.25.0

**Date** October 05, 2021

- @humitos: Docs: comment about how to add a new tool/version for builders (#8548)
- @astrojuanlu: Add checkbox to subscribe new users to newsletter (#8546)
- @humitos: Build: missing updates from review (#8544)
- @humitos: EmbedAPI: allow CORS for /api/v3/embed/ (#8543)
- @humitos: Script tools cache: fix environment variables (#8541)
- @humitos: EmbedAPIv3: proxy URLs to be available under /_/ (#8540)
- @humitos: Docker: use the correct image tag (#8539)
- @humitos: Requirement: ping django-redis-cache to git tag (#8536)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 39 (#8531)
- @astrojuanlu: Promote and restructure guides (#8528)
- @stsewd: Analytics: allow to download all data (#8498)
- @stsewd: HistoricalRecords: add additional fields (ip and browser) (#8490)
5.9.10 Version 5.24.0

Date September 28, 2021

- @humitos: EmbedAPIv3: updates after QA with sphinx-hoverxref (#8521)
- @stsewd: Contact users: show progress (#8518)
- @stsewd: Rename audit retention days setting (#8517)
- @stsewd: Contact users: make notification sticky (#8516)
- @stsewd: Contact users: report usernames instead of PK (#8515)
- @stsewd: Audit: update admin (#8514)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 38 (#8510)
- @stsewd: New config for new docker build images (#8478)
- @stsewd: Audit: expose user security logs (#8477)
- @humitos: Build: use new Docker images from design document (#8453)
- @humitos: Embed APIv3: initial implementation (#8319)

5.9.11 Version 5.23.6

Date September 20, 2021

- @astrojuanlu: Change newsletter form (#8509)
- @stsewd: Contact users: Allow to pass additional context to each email (#8507)
- @astrojuanlu: Update onboarding (#8504)
- @astrojuanlu: List default installed dependencies (#8503)
- @astrojuanlu: Clarify that the development installation instructions are for Linux (#8494)
- @stsewd: Audit: attach organization (#8491)
- @astrojuanlu: Add virtual env instructions to local installation (#8488)
- @humitos: Requirement: update orjson (#8487)
- @humitos: Release 5.23.5 (#8486)
- @astrojuanlu: New Read the Docs tutorial, part II (#8473)

5.9.12 Version 5.23.5

Date September 14, 2021

- @humitos: Organization: only mark artifacts cleaned as False if they are True (#8481)
- @astrojuanlu: Fix link to version states documentation (#8475)
- @stsewd: OAuth models: increase avatar_url length (#8472)
- @pzhlkj6612: Docs: update the links to the dependency management content of setuptools docs (#8470)
- @stsewd: Permissions: avoid using project.users, use proper permissions instead (#8458)
- @humitos: Docker build images: update design doc (#8447)
• @astrojuanlu: New Read the Docs tutorial, part I (#8428)

5.9.13 Version 5.23.4

Date  September 07, 2021

• @pzhlkj6612: Docs: update the links to the dependency management content of setuptools docs (#8470)
• @nienn: Add custom team img styling (#8467)
• @nienn: Docs: Change “right-click” to “click” (#8465)
• @stsewd: Permissions: avoid using project.users, use proper permissions instead (#8458)
• @stsewd: Add templatetag to filter by admin projects (#8456)
• @stsewd: Support form: don’t allow to change the email (#8455)
• @stsewd: Search: show only results from the current role_name being filtered (#8454)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 35 (#8451)
• @stsewd: Update common (#8449)
• @stsewd: API v3 (subprojects): filter by correct owner/organization (#8446)
• @astrojuanlu: Rework Team page (#8441)
• @mforbes: Added note about how to use Anaconda Project. (#8436)
• @stsewd: Contact users: pass user and domain in the context (#8430)
• @astrojuanlu: New Read the Docs tutorial, part I (#8428)
• @stsewd: Footer: remove auth block (#8397)
• @stsewd: API: fix subprojects creation when organizations are enabled (#8393)
• @stsewd: QuerySets: remove unused overrides (#8299)
• @stsewd: QuerySets: filter permissions by organizations (#8298)

5.9.14 Version 5.23.3

Date  August 30, 2021

• @stsewd: Update common (#8449)
• @stsewd: Upgrade ES to 7.14.0 (#8448)
• @humitos: Docs: typo in tutorial (#8442)
• @astrojuanlu: Docs miscellaneous enhancements (#8440)
• @astrojuanlu: New Read the Docs tutorial, part I (#8428)
• @humitos: Track organization artifacts cleanup (#8418)
5.9.15 Version 5.23.2

Date August 24, 2021

- @astrojuanlu: Add MyST (Markdown) examples to “cross referencing with Sphinx” guide (#8437)
- @saadmk11: Added Search and Filters for RemoteRepository and RemoteOrganization admin list page (#8431)
- @agjohnson: Try out codeowners (#8429)
- @humitos: Proxito: do not log response header for each custom domain request (#8427)
- @cclauss: Fix undefined names (#8425)
- @stsewd: Allow cookies from cross site requests to avoid problems with iframes (#8422)
- @cclauss: Finish codespell – Concludes #8409 (#8421)
- @cclauss: codespell CHANGELOG.rst (#8420)
- @cclauss: codespell part 2 - Continues #8409 (#8419)
- @ericholscher: Don’t filter on large items in the auditing sidebar. (#8417)
- @astrojuanlu: Fix YAML extension (#8416)
- @ericholscher: Release 5.23.1 (#8415)
- @stsewd: Audit: attach project from the request if available (#8414)
- @stsewd: Docs: update logout button image (#8413)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 33 (#8411)
- @cclauss: Fix typos discovered by codespell in /docs (#8409)
- @stsewd: Support: update contact information via Front webhook (#8406)
- @stsewd: Docs: environment variables (#8390)
- @stsewd: Allow users to remove themselves from a project (#8384)
- @stsewd: Docs: document how to terminate a session (#8286)

5.9.16 Version 5.23.1

Date August 16, 2021

- @cclauss: Fix typos discovered by codespell in /docs (#8409)
- @stsewd: Audit: use analytics’ get_client_ip (#8404)
- @steko: Add documentation about webhooks for Gitea (#8402)
- @ericholscher: Add CSP header to the domain options (#8388)
- @stsewd: Cookies: set samesite: Lax by default (#8304)
- @stsewd: Docs: document how to terminate a session (#8286)
- @stsewd: Docs: update sharing (#8239)
5.9.17 Version 5.23.0

Date August 09, 2021

• @ericholscher: Only call analytics tracking of flyout when analytics are enabled (#8398)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 31 (#8385)
• @stsewd: Audit: track user events (#8379)
• @stsewd: Cookies: set samesite: Lax by default (#8304)
• @stsewd: Docs: update sharing (#8239)
• @DetectedStorm: Update LICENSE (#5125)

5.9.18 Version 5.22.0

Date August 02, 2021

• @pzhlkj6612: Docs: fix typo in versions.rst: -> need (#8383)
• @ericholscher: Remove clickjacking middleware for proxito (#8378)
• @stsewd: Config file: use string for python.version (#8372)
• @humitos: Add support for Python3.10 on testing Docker image (#8328)
• @stsewd: Analytics: don’t fail if the page was created in another request (#8310)

5.9.19 Version 5.21.0

Date July 27, 2021

• @stsewd: Fix migrations (#8373)
• @ericholscher: Build out the MyST section of the getting started (#8371)
• @stsewd: Fix tasks (#8370)
• @astrojuanlu: Update common (#8368)
• @astrojuanlu: Redirect users to appropriate support channels using template chooser (#8366)
• @humitos: Proxito: return user-defined HTTP headers on custom domains (#8360)
• @ericholscher: Release 5.20.3 (#8356)
• @stsewd: Track model changes with django-simple-history (#8355)
• @stsewd: SSO: move models (#8330)
5.9.20 Version 5.20.3

Date July 19, 2021

• @Nkarnaud: change vieweable to viewable on home page: issue#8346 (#8351)
• @stsewd: Builds: don’t record git rev-parse command (#8348)
• @stsewd: UI: allow to close notifications (#8345)
• @stsewd: Use email from DEFAULT_FROM_EMAIL to contact users (#8344)
• @stsewd: Downgrade taggit (#8342)
• @stsewd: Don’t mutate env vars outside the BuildEnv classes (#8340)
• @stsewd: Guides: how to import a private project using an ssh key (#8336)

5.9.21 Version 5.20.2

Date July 13, 2021

• @humitos: psycopg2: pin to a compatible version with Django 2.2 (#8335)
• @stsewd: Contact owners: use correct organization to filter (#8325)
• @humitos: Design doc: fix render api endpoints (#8320)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 27 (#8317)
• @mongolsteppe: Fixing minor error (#8313)
• @stsewd: Build: remove after_vcs signal (#8311)
• @The-Compiler: Add link to redirect docs (#8308)
• @ericholscher: Add docs about setting up permissions for GH apps & orgs (#8305)
• @stsewd: Schema: fix version type (#8303)
• @stsewd: Slugify: don’t generate slugs with trailing - (#8302)
• @ericholscher: Increase guide depth (#8300)
• @humitos: autoscaling: remove app autoscaling tasks (#8297)
• @humitos: PR build status: re-try up to 3 times if it fails for some reason (#8296)
• @SethFalco: feat: add json schema (#8294)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 26 (#8293)
• @stsewd: Organizations: validate that a correct slug is generated (#8292)
• @stsewd: Footer: remove unused context vars (#8285)
• @astrojuanlu: Add new guide about Jupyter in Sphinx (#8283)
• @humitos: oauth webhook: check the Project has a RemoteRepository (#8282)
• @stsewd: Allow to email users from a management command (#8243)
• @humitos: Design doc: Embed APIv3 (#8222)
• @astrojuanlu: Add proposal for new Sphinx and RTD tutorials (#8106)
• @stsewd: Allow to change the privacy level of external versions (#7825)
Read the Docs Documentation, Release 6.3.0

- @stsewd: Add tests for remove index files (#6381)

### 5.9.22 Version 5.20.1

**Date** June 28, 2021

- @stsewd: Organizations: validate that a correct slug is generated (#8292)
- @stsewd: Footer: remove unused context vars (#8285)
- @stsewd: Search: remove additional fields (#8284)
- @humitos: oauth webhook: check the Project has a RemoteRepository (#8282)
- @stsewd: Search: small improvements (#8276)
- @stsewd: Search: ask for confirmation when running reindex_elasticsearch (#8275)
- @saadmk11: Hit Elasticsearch only once for each search query through the APIv2 (#8228)
- @humitos: Design doc: Embed APIv3 (#8222)
- @stsewd: QuerySets: remove include_all (#8212)
- @astrojuanlu: Add proposal for new Sphinx and RTD tutorials (#8106)
- @stsewd: Add tests for remove index files (#6381)

### 5.9.23 Version 5.20.0

**Date** June 22, 2021

- @humitos: Migration: fix RemoteRepository - Project data migration (#8271)
- @ericholscher: Release 5.19.0 (#8266)
- @humitos: Sync RemoteRepository for external collaborators (#8265)
- @stsewd: Git: don’t expand envvars in Gitpython (#8263)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 24 (#8262)
- @stsewd: Builds: check for nonexistent object (#8261)
- @humitos: Make Project -> ForeignKey -> RemoteRepository (#8259)
- @agjohnson: Add basic security policy (#8254)
- @stsewd: Search: remove workaround for subprojects (#8211)
- @stsewd: Search: allow to filter by project slugs (#8149)

### 5.9.24 Version 5.19.0

**Warning:** This release contains a security fix to our CSRF settings: https://github.com/readthedocs/readthedocs.org/security/advisories/GHSA-3v5m-qmm9-3c6c

**Date** June 15, 2021

- @stsewd: Builds: check for nonexistent object (#8261)
• @ericholscher: Remove video from our Sphinx quickstart. (#8246)
• @ericholscher: Remove “Markdown” from Mkdocs title (#8245)
• @astrojuanlu: Make sustainability page more visible (#8244)
• @stsewd: Builds: move send_build_status to builds/tasks.py (#8241)
• @humitos: Add ability to rebuild a specific build (#8227)
• @ericholscher: Don’t do any CORS checking on Embed API requests (#8226)
• @stsewd: Footer: return well formed html (#8202)
• @agjohnson: Add project/build filters (#8142)
• @humitos: Sign Up: limit the providers allowed to sign up (#8062)
• @stsewd: Search: use multi-fields for Wildcard queries (#7613)
• @ericholscher: Add ability to rebuild a specific build (#6995)

5.9.25 Version 5.18.0

**Date** June 08, 2021

• @stsewd: Fix tests (#8240)
• @ericholscher: Backport manual indexes (#8235)
• @ericholscher: Clean up SSO docs (#8233)
• @stsewd: Cache get_project (#8231)
• @ericholscher: Don’t do any CORS checking on Embed API requests (#8226)
• @saadmk11: Optimize Index time database query (#8224)
• @stsewd: Proxito: check if attribute exists (#8220)
• @agjohnson: Update gitter channel name (#8217)
• @ericholscher: Remove IRC from our docs (#8216)
• @stsewd: QuerySets: filter by admin/team (#8214)
• @stsewd: QuerySets: remove unused detail parameter (#8213)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 21 (#8206)
• @stsewd: QuerySets: refactor _add_user_repos (#8182)
• @stsewd: Organizations: simplify querysets/managers (#8180)
• @akien-mga: Docs: Add some details to the User Defined Redirects (#7894)
• @stsewd: Search: refactor views (#8157)
• @stsewd: Search: little optimization when saving search queries (#8132)
• @akien-mga: Docs: Add some details to the User Defined Redirects (#7894)
• @agjohnson: Add APIv3 version edit URL (#7594)
• @saadmk11: Add List API Endpoint for RemoteRepository and RemoteOrganization (#7510)
5.9.26 Version 5.17.0

Date May 24, 2021

- @stsewd: Proxito: don’t require the middleware for proxied apis (#8203)
- @stsewd: Builds: prevent code injection in cwd (#8198)
- @ericholscher: Remove specific name from security page at user request (#8195)
- @humitos: Docker: remove volumes= argument when creating the container (#8194)
- @stsewd: Proxito: fix https and canonical redirects (#8193)
- @stsewd: API v2: allow listing when using private repos (#8192)
- @stsewd: Docker: set cwd explicitly (#8191)
- @stsewd: API v2: allow to filter more endpoints (#8189)
- @stsewd: Proxito: redirect to main project from subprojects (#8187)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 20 (#8186)
- @agjohnson: Add DPA to legal docs in documentation (#8130)

5.9.27 Version 5.16.0

Date May 18, 2021

- @stsewd: QuerySets: check for .is_superuser instead of has_perm (#8181)
- @humitos: Build: use is_active method to know if the build should be skipped (#8179)
- @humitos: APIv2: disable listing endpoints (#8178)
- @stsewd: Project: use IntegerField for remote_repository from project form. (#8176)
- @stsewd: Docs: remove some lies from cross referencing guide (#8173)
- @stsewd: Docs: add space to bash code (#8171)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 19 (#8170)
- @stsewd: Querysets: include organizations in is_active check (#8163)
- @stsewd: Querysets: remove private and for_project (#8158)
- @davidfischer: Disable FLOC by introducing permissions policy header (#8145)
- @stsewd: Build: allow to install packages with apt (#8065)

5.9.28 Version 5.15.0

Date May 10, 2021

- @stsewd: Ads: don’t load script if a project is marked as ad_free (#8164)
- @stsewd: Querysets: include organizations in is_active check (#8163)
- @stsewd: Querysets: simplify project querysets (#8154)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 18 (#8153)
- @stsewd: Search: default to search on default version of subprojects (#8148)
• @stsewd: Remove protected privacy level (#8146)
• @stsewd: Embed: fix paths that start with / (#8139)
• @humitos: Metrics: run metrics task every 30 minutes (#8138)
• @humitos: web-celery: add logging for OOM debug on suspicious tasks (#8131)
• @agjohnson: Fix a few style and grammar issues with SSO docs (#8109)
• @stsewd: Embed: don’t fail while querying sections with bad id (#8084)
• @stsewd: Design doc: allow to install packages using apt (#8060)

5.9.29 Version 5.14.3

Date April 26, 2021
• @humitos: Metrics: run metrics task every 30 minutes (#8138)
• @humitos: web-celery: add logging for OOM debug on suspicious tasks (#8131)
• @stsewd: Celery router: check all n last builds for Conda (#8129)
• @jonels-msft: Include aria-label in flyout search box (#8127)
• @humitos: Logging: use %s to format the variable (#8125)
• @stsewd: Build: improve list_packages_installed (#8122)
• @stsewd: BuildCommand: don’t leak stacktrace to the user (#8121)
• @stsewd: API (v2): use empty list in serializer’s exclude (#8120)
• @astrojuanlu: Miscellaneous doc improvements (#8118)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 16 (#8117)
• @agjohnson: Fix a few style and grammar issues with SSO docs (#8109)

5.9.30 Version 5.14.2

Date April 20, 2021
• @stsewd: OAuth: check if user exists (#8115)
• @stsewd: Sync versions: don’t fetch/return all versions (#8114)
• @astrojuanlu: Improve contributing docs, take 2 (#8113)
• @stsewd: ImportedFile: remove md5 field (#8111)
• @stsewd: Config file: improve docs and help text (#8110)
• @stsewd: Docs: add warning about design docs (#8104)
• @Harmon758: Docs: fix typo in config-file/v2.rst (#8102)
• @cocobennett: Improve documentation on contributing to documentation (#8082)
5.9.31 Version 5.14.1

Date April 13, 2021

- @stsewd: OAuth: protection against deleted objects (#8081)
- @cocobennett: Add page and page_size to server side api documentation (#8080)
- @stsewd: Version warning banner: inject on role="main" or main tag (#8079)
- @stsewd: OAuth: avoid undefined var (#8078)
- @stsewd: Conda: protect against None when appending core requirements (#8077)
- @humitos: SSO: add small paragraph mentioning how to enable it on commercial (#8063)
- @agjohnson: Add separate version create view and create view URL (#7595)

5.9.32 Version 5.14.0

Date April 06, 2021

This release includes a security update which was done in a private branch PR. See our security changelog for more details.

- @pyup-bot: pyup: Scheduled weekly dependency update for week 14 (#8071)
- @astrojuanlu: Clarify ad-free conditions (#8064)
- @humitos: SSO: add small paragraph mentioning how to enable it on commercial (#8063)
- @stsewd: Build environment: allow to run commands with a custom user (#8058)
- @humitos: Design document for new Docker images structure (#7566)

5.9.33 Version 5.13.0

Date March 30, 2021

- @stsewd: Test proxied embed API (#8051)
- @stsewd: Feature flag: remove EXTERNAL_BUILD (#8050)
- @stsewd: Sync versions: always use a task (#8049)
- @stsewd: Versions: don’t create versions in bulk (#8046)
- @stsewd: Embed: add cache tags (#8045)
- @ericholscher: Fix proxito slash redirect for leading slash (#8044)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 12 (#8038)
- @humitos: Docs: cleanup of old/deprecated documents (#7994)
- @flying-sheep: Add publicly visible env vars (#7891)
- @stsewd: Remove broadcast function (#7044)
5.9.34 Version 5.12.2

Date March 23, 2021

- @humitos: AWS homepage link (#8037)
- @hukkinj1: Fix a typo in the docs (#8035)
- @stsewd: Clean some feature flags (#8034)
- @ericholscher: Standardize footerjs code (#8032)
- @stsewd: Docs: remove pdf format in MkDocs example (#8030)
- @stsewd: Search: don’t leak data for projects with this feature disabled (#8029)
- @ericholscher: Canonicalize all proxito slashes (#8028)
- @ericholscher: Make pageviews analytics show top 25 pages (#8027)
- @ericholscher: Add CSV header data for search analytics (#8026)
- @stsewd: HTMLFile: make md5 field nullable (#8025)
- @humitos: Use RemoteRepository relation to match already imported projects (#8024)
- @stsewd: Badge: exclude duplicated builds (#8023)
- @stsewd: Intersphinx: declare user agent (#8022)
- @stsewd: Builds: restart build commands before a new build (#7999)
- @saadmk11: Remote Repository and Remote Organization Normalization (#7949)
- @stsewd: Build: don’t track changed files (#7874)

5.9.35 Version 5.12.1

Date March 16, 2021

- @pyup-bot: pyup: Scheduled weekly dependency update for week 11 (#8019)
- @stsewd: Embed: Allow to override embed view for proxied use (#8018)
- @humitos: RemoteRepository: Improvements to sync_vcs_data.py script (#8017)
- @humitos: Stripe checkout: handle events (#8016)
- @humitos: Remove contrib/ directory (#8014)
- @stsewd: Dockerfile: install lsb_release (#8010)
- @davidfischer: Fix AWS image so it looks sharp (#8009)
- @stsewd: Update common (#8008)
- @stsewd: Update Sphinx (#8007)
- @2Fake: fix small typo (#8005)
- @stsewd: Embed: validate query arguments (#8003)
- @humitos: Stripe Checkout: handle duplicated webhook (#8002)
- @saadmk11: Add __str__ to RemoteRepositoryRelation and RemoteOrganizationRelation and Use raw_id_fields in Admin (#8001)
- @saadmk11: Remove duplicate results from RemoteOrganization API (#8000)
• @humitos: Typo fixed on checkout.js (#7998)
• @ericholscher: Make SupportView login_required (#7997)
• @ericholscher: Release 5.12.0 (#7996)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 10 (#7995)
• @saadmk11: Remove json field from RemoteRepositoryRelation and RemoteOrganizationRelation model (#7993)
• @humitos: Use independent Docker image to build assets (#7992)
• @Pradhvan: Fixes typo in getting-started-with-sphinx: (#7991)
• @humitos: Allow donate app to use Stripe Checkout for one-time donations (#7983)
• @ericholscher: Add proxito healthcheck (#7948)
• @Pradhvan: Docs: Adds Myst to the getting started with sphinx (#7938)
• @humitos: Use Stripe Checkout for Gold Users (#7889)
• @stsewd: Docs: guide about reproducible builds (#7888)

5.9.36 Version 5.12.0

Date March 08, 2021
• @pyup-bot: pyup: Scheduled weekly dependency update for week 10 (#7995)
• @saadmk11: Remove json field from RemoteRepositoryRelation and RemoteOrganizationRelation model (#7993)
• @humitos: Use independent Docker image to build assets (#7992)
• @Pradhvan: Fixes typo in getting-started-with-sphinx: (#7991)
• @stsewd: Search: use doctype from indexed pages instead of the db (#7984)
• @humitos: Allow donate app to use Stripe Checkout for one-time donations (#7983)
• @humitos: Update development/standards guide (#7981)
• @stsewd: Docs: update expand_tabs to work with the latest version of sphinx-tabs (#7979)
• @ericholscher: Fix build routing (#7978)
• @stsewd: Builds: register tasks to delete inactive external versions (#7975)
• @stsewd: Embed: fix join (#7974)
• @stsewd: Embed: change proxied urls (#7973)
• @ericholscher: refactor footer, add jobs & status page (#7970)
• @stsewd: Sphinx domain: remove API (#7969)
• @humitos: Upgrade postgres-client to v12 in Docker image (#7967)
• @saadmk11: Add management command to Load Project and RemoteRepository Relationship from JSON file (#7966)
• @astrojuanlu: Update guide on conda support (#7965)
• @stsewd: Embed: add more tests (#7962)
• @humitos: Lower rank of development/install.html (#7960)
• @stsewd: Embed: refactor view (#7955)
• @stsewd: Search: make –queue required for management command (#7952)
• @ericholscher: Add proxito healthcheck (#7948)
• @Pradhvan: Docs: Adds Myst to the getting started with sphinx (#7938)
• @ericholscher: Add a support form to the website (#7929)
• @humitos: Use Stripe Checkout for Gold Users (#7889)
• @stsewd: Docs: guide about reproducible builds (#7888)
• @stsewd: Docs: update links from build images (#7886)
• @stsewd: Install latest mkdocs by default as we do with sphinx (#7869)
• @stsewd: Docs: document analytics (#7857)
• @stsewd: Remove some feature flags (#7846)
• @stsewd: Update requirements/deploy.txt (#7843)
• @humitos: Implementation of APIv3 (#4863)

5.9.37 Version 5.11.0

**Date**  March 02, 2021

• @saadmk11: Add management command to Load Project and RemoteRepository Relationship from JSON file (#7966)
• @humitos: Lower rank of development/install.html (#7960)
• @saadmk11: Add Management Command to Dump Project and RemoteRepository Relationship in JSON format (#7957)
• @davidfischer: Enable the cached template loader (#7953)
• @stsewd: Update common to master (#7951)
• @stsewd: Embed: refactor tests (#7947)
• @stsewd: Downgrade jedi (#7946)
• @FatGrizzly: Added warnings for previous gitbook users (#7945)
• @stsewd: Move embed app (#7943)
• @ericholscher: Change our sponsored hosting from Azure -> AWS. (#7940)
• @Pradhvan: Docs: Adds Myst to the getting started with sphinx (#7938)
• @ericholscher: Add a support form to the website (#7929)
• @stsewd: Drop six (#7890)
• @fabianmp: Allow to use a different url for intersphinx object file download (#7807)
5.9.38 Version 5.10.0

Date February 23, 2021

• @pyup-bot: pyup: Scheduled weekly dependency update for week 08 (#7941)
• @PawelBorkar: Update license (#7934)
• @humitos: Route external versions to the queue were default version was built (#7933)
• @humitos: Pin jedi dependency to avoid breaking ipython (#7932)
• @humitos: Use admin user for SLUMBER API on local environment (#7925)
• @stsewd: Search: add cache tags (#7922)
• @humitos: Use S3 from community (#7920)
• @stsewd: Use only one variant of the config file (#7918)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 07 (#7913)
• @humitos: Router PRs builds to last queue where a build was executed (#7912)
• @stsewd: Search: improve re-index management command (#7904)
• @stsewd: Search: link to main project in subproject results (#7880)
• @humitos: Upgrade Celery and friends to latest versions (#7786)

5.9.39 Version 5.9.0

Date February 16, 2021

Last Friday we migrated our site from Azure to AWS (read the blog post). This is the first release into our new AWS infra.

• @humitos: Router PRs builds to last queue where a build was executed (#7912)
• @humitos: Update common/submodule (#7910)
• @humitos: Upgrade Redis version to match production (#7909)
• @davidfischer: Make storage classes into module level vars (#7908)
• @csdev: fix typo (#7902)
• @humitos: Match Redis version from AWS producion (#7897)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 06 (#7896)
• @nedbat: Doc fix: two endpoints had ‘pip’ for the project_slug (#7895)
• @stsewd: Set storage for BuildCommand and BuildEnvironment as private (#7893)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 05 (#7887)
• @humitos: Add support for Python 3.9 on “testing” Docker image (#7885)
• @stsewd: Add version_changed signal (#7878)
• @stsewd: Search: don’t index permalinks (#7876)
• @stsewd: Update common (#7873)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 04 (#7867)
• @humitos: Log Stripe errors when trying to delete customer/subscription (#7853)
5.9.40 Version 5.8.5

Date January 18, 2021

• @pyup-bot: pyup: Scheduled weekly dependency update for week 03 (#7840)
• @humitos: Speed up concurrent builds by limited to 5 hours ago (#7839)
• @humitos: Match Redis version with production (#7838)
• @saadmk11: Add Option to Enable External Builds Through Project Update API (#7834)
• @stsewd: Docs: mention the version warning is for sphinx only (#7832)
• @agjohnson: Hide design docs from documentation (#7826)
• @stsewd: Update docs about preview from pull/merge requests (#7823)
• @humitos: Documentation for /organizations/ endpoint in commercial (#7800)
• @stsewd: Privacy Levels: migrate protected projects to private (#7608)
• @pawamoy: Don’t lose python/name tags values in mkdocs.yml (#7507)
• @stsewd: Install latest version of setuptools (#7290)
• @humitos: Implementation of APIv3 (#4863)
5.9.41 Version 5.8.4

Date January 12, 2021

• @pyup-bot: pyup: Scheduled weekly dependency update for week 02 (#7818)
• @stsewd: List SYNC_VERSIONS_USING_A_TASK flag in the admin (#7802)
• @ericholscher: Update build concurrency numbers for Business (#7794)
• @stsewd: Sphinx: use html_baseurl for setting the canonical URL (#7540)

5.9.42 Version 5.8.3

Date January 05, 2021

• @humitos: Change query on send_build_status task for compatibility with .com (#7797)
• @ericholscher: Update build concurrency numbers for Business (#7794)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 01 (#7793)
• @timgates42: docs: fix simple typo, -> translations (#7781)
• @ericholscher: Release 5.8.2 (#7776)
• @humitos: Use Python3.7 on conda base environment when using mamba (#7773)
• @stsewd: Remove domain verify signal and task (#7763)
• @stsewd: Import page: fix wizard form (#7702)
• @ericholscher: Migrate sync_versions from an API call to a task (#7548)
• @humitos: Design document for RemoteRepository DB normalization (#7169)

5.9.43 Version 5.8.2

Date December 21, 2020

• @humitos: Use Python3.7 on conda base environment when using mamba (#7773)
• @stsewd: Remove domain verify signal and task (#7763)
• @humitos: Register StopBuilder task to be executed by builders (#7759)
• @stsewd: Footer: remove absolute_uri (#7758)
• @stsewd: Search: use alias to link to search results of subprojects (#7757)
• @stsewd: Footer: remove jsonp call (#7756)
• @humitos: Register AutoscaleBuildersTask (#7755)
• @saadmk11: Set The Right Permissions on GitLab OAuth RemoteRepository (#7753)
• @stsewd: Use lru_cache for caching methods (#7751)
• @fabianmp: Allow to add additional binds to Docker build container (#7684)
5.9.44 Version 5.8.1

**Date** December 14, 2020

- @humitos: Register ShutdownBuilder task (#7749)
- @saadmk11: Use “path_with_namespace” for GitLab RemoteRepository full_name Field (#7746)
- @stsewd: Features: remove USE_NEW_PIP_RESOLVER (#7745)
- @stsewd: Version sync: exclude external versions when deleting (#7742)
- @stsewd: Search: limit number of sections and domains to 10K (#7741)
- @stsewd: Traffic analytics: don’t pass context if the feature isn’t enabled (#7740)
- @stsewd: Analytics: move page views to its own endpoint (#7739)
- @stsewd: FeatureQuerySet: make check for date inclusive (#7737)
- @stsewd: Typo: date -> data (#7736)
- @saadmk11: Use remote_id and vcs_provider Instead of full_name to Get RemoteRepository (#7734)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 49 (#7730)
- @saadmk11: Update parts of code that were using the old RemoteRepository model fields (#7728)
- @stsewd: Builds: don’t delete them when a version is deleted (#7679)
- @stsewd: Sync versions: create new versions in bulk (#7382)
- @humitos: Use mamba under a feature flag to create conda environments (#6815)

5.9.45 Version 5.8.0

**Date** December 08, 2020

- @stsewd: Update common (#7731)
- @stsewd: Bitbucket: mainbranch can be None (#7725)
- @stsewd: Search: use with_positions_offsets term vector for some fields (#7724)
- @stsewd: Search: filter only active and built versions from subprojects (#7723)
- @stsewd: Extra features: allow to display them conditionally (#7715)
- @humitos: Define pre/post_collectstatic signals and send them (#7701)
- @davidfischer: Support the new Google analytics gtag.js (#7691)
- @stsewd: HTMLFile: remove slug field (#7680)
- @stsewd: External versions: delete after 3 months of being merged/closed (#7678)
- @stsewd: Automation Rules: keep history of recent matches (#7658)
- @stsewd: Search: update to ES 7.x (#7582)
5.9.46 Version 5.7.0

Date December 01, 2020

- @davidfischer: Ensure there is space for sidebar ads (#7716)
- @humitos: Install six as core requirement for builds (#7710)
- @stsewd: Features: increase feature_id max_length (#7698)
- @ericholscher: Release 5.6.1 (#7695)
- @stsewd: Tests: mock trigger_build (#7681)
- @stsewd: Sync versions: use stable version instead of querying all versions (#7380)

5.9.47 Version 5.6.5

Date November 23, 2020

- @stsewd: Tests: mock trigger_build (#7681)
- @stsewd: Tests: mock update_docs_task to speed up tests (#7677)
- @stsewd: Versions: add timestamp fields (#7676)
- @stsewd: Tests: create an organization when running in .com (#7673)
- @davidfischer: Speed up the tag index page (#7671)
- @davidfischer: Fix for out of order script loading (#7670)
- @davidfischer: Set ad configuration values if using explicit placement (#7669)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 46 (#7668)
- @stsewd: Tests: mock trigger build to speed up tests (#7661)
- @stsewd: Remote repository: save and set default_branch (#7646)
- @stsewd: Search: exclude some fields from source results (#7640)
- @stsewd: Search: allow to search on different versions of subprojects (#7634)
- @stsewd: Search: refactor api view (#7633)
- @saadmk11: Add Initial Modeling with Through Model and Data Migration for RemoteRepository Model (#7536)
- @stsewd: ImportedFile: remove slug 1/2 (#7228)
- @humitos: Changes required for APIv3 in corporate (#6489)

5.9.48 Version 5.6.4

Date November 16, 2020

- @davidfischer: Fix for out of order script loading (#7670)
- @davidfischer: Set ad configuration values if using explicit placement (#7669)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 46 (#7668)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 45 (#7655)
- @stsewd: Automation rules: add delete version action (#7644)
5.9.49 Version 5.6.3

Date November 10, 2020

• @pyup-bot: pyup: Scheduled weekly dependency update for week 43 (#7602)

5.9.50 Version 5.6.2

Date November 03, 2020

• @humitos: Check only for override settings (part 2) (#7630)
• @humitos: Check only override settings (#7628)
• @davidfischer: Display sidebar ad when scrolled (#7621)
• @humitos: Reserve 1Gb for Application Memory (#7618)
• @humitos: Catch requests.exceptions.ReadTimeout when removing container (#7617)
• @humitos: Allow search and filter in Django Admin for Message model (#7615)
• @stsewd: CI: use badge from circleci (#7614)
• @stsewd: Search: respect feature flag in dashboard search (#7611)
• @ericholscher: Release 5.6.1 (#7604)
• @stsewd: CI: use circleci (#7603)

5.9.51 Version 5.6.1

Date October 26, 2020

• @agjohnson: Bump common to include docker task changes (#7597)
• @agjohnson: Default to sphinx theme 0.5.0 when defaulting to latest sphinx (#7596)
• @humitos: Use correct Cache-Tag (CDN) and X-RTD-Project header on subprojects (#7593)
• @davidfischer: Ads JS hotfix (#7586)
• @agjohnson: Add remoterepo query param (#7580)
• @agjohnson: Undeprecate APIv2 in docs (#7579)
• @agjohnson: Add settings and docker configuration for working with new theme (#7578)
• @stsewd: Search: exclude changelog from results (#7570)
• @stsewd: Search: better results for single terms (#7569)
• @stsewd: Search: refactor query objects (#7568)
• @humitos: Add our readthedocs_processor data to our notifications (#7565)
• @stsewd: Update ES to 6.8.12 (#7559)
Read the Docs Documentation, Release 6.3.0

- @stsewd: Builds: always install latest version of our sphinx extension (#7542)
- @stsewd: Bring back project privacy level (#7525)
- @ericholscher: Add future default true to Feature flags (#7524)
- @stsewd: Add feature flag to not install the latest version of pip (#7522)
- @davidfischer: No longer proxy RTD ads through RTD servers (#7506)
- @stsewd: Subprojects: fix form (#7491)
- @AvdN: correct inconsistent indentation of YAML (#7459)

5.9.52 Version 5.6.0

Date October 19, 2020

- @stsewd: Search: exclude changelog from results (#7570)
- @stsewd: Docs: show example of a requirements.txt file (#7563)
- @stsewd: Update ES to 6.8.12 (#7559)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 40 (#7537)
- @ericholscher: Add future default true to Feature flags (#7524)
- @davidfischer: No longer proxy RTD ads through RTD servers (#7506)
- @davidfischer: Allow projects to opt-out of analytics (#7175)

5.9.53 Version 5.5.3

Date October 13, 2020

- @ericholscher: Add a reference to the Import guide at the start of Getting started (#7547)
- @kuzmoyev: Include month-ago day to traffic data (#7545)
- @stsewd: Downloads: fix translation of a subproject (#7541)
- @stsewd: Domains: more robust form (#7523)
- @stsewd: Revert “Revert ES: update dependencies” (#7439)
- @stsewd: Search: remove old endpoint (#7414)

5.9.54 Version 5.5.2

Date October 06, 2020

- @stsewd: Domain: show created/modified date in admin (#7517)
- @stsewd: Tests: fix eslint (#7516)
- @ericholscher: Revert “New docker image for builders: 8.0” (#7514)
- @srijan-deepsOURCE: Fix some code quality issues (#7494)
5.9.55 Version 5.5.1

Date September 28, 2020

- @stsewd: Domain: fix form (#7502)
- @stsewd: Builders: little refactor (#7500)
- @ericholscher: Add proper div names on our ad placements (#7493)
- @saadmk11: APIv3 Version list slug filter added (#7372)
- @humitos: Use "-j auto" on sphinx-build command to build in parallel (#7128)

5.9.56 Version 5.5.0

Date September 22, 2020

- @stsewd: Don’t install pygments (#7490)
- @humitos: Limit concurrency per-organization (#7489)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 37 (#7482)
- @humitos: Use permissions (project and group) for RemoteRepository.admin on GitLab (#7479)

5.9.57 Version 5.4.3

Date September 15, 2020

- @stsewd: Domain: inherit from TimeStampedModel (#7476)
- @stsewd: Truncate output at the start for large commands (#7473)
- @stsewd: Add dependency explicitly (dateutil) (#7415)
- @stsewd: Domains: add ssl_status field (#7398)
- @stsewd: Search: allow ignoring files from indexing (#7308)
- @stsewd: Search: SSS integration guide (#7232)

5.9.58 Version 5.4.2

Date September 09, 2020

- @humitos: Show “Connected Services” form errors to the user (#7469)
- @humitos: Allow to extend OrganizationTeamBasicForm from -corporate (#7467)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 36 (#7465)
- @AvdN: correct invalid YAML (#7458)
- @stsewd: Remote repository: filter by account before deleting (#7454)
- @humitos: Truncate the beginning of the commands’ output (#7449)
- @davidfischer: Update links to advertising (#7443)
- @stsewd: Revert “Don’t retry on POST” (#7442)
- @stsewd: Organizations: move signals (#7441)
5.9.59 Version 5.4.1

Date September 01, 2020

- @stsewd: Docs: update readthedocs-sphinx-search ext (#7427)
- @humitos: Upgrade Django to 2.2.16 (#7426)
- @bmorrisson4: Fix typo in docs_guides/adding-custom-css.rst (#7424)
- @stsewd: Test: set privacy level explicitly (#7422)
- @stsewd: Pip: test new resolver (#7412)
- @stsewd: Update common (#7411)
- @stsewd: Release 5.4.0 (#7410)
- @stsewd: Docker: install requirements from local changes (#7409)
- @stsewd: ES: update dependencies (#7408)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 34 (#7406)
- @stsewd: API client: don’t retry on POST (#7383)
- @saadmk11: build_url added to all API v3 build endpoints (#7373)
- @stsewd: Guide: deprecating content (#7333)
- @humitos: Auto-join email users field for Team model (#7328)
- @stsewd: Guide: Cross-referencing with Sphinx (#7326)
- @humitos: Sync RemoteRepository and RemoteOrganization in all VCS providers (#7310)
- @stsewd: Page views: use origin URL instead of page name (#7293)

5.9.60 Version 5.4.0

Date August 25, 2020

- @stsewd: ES: match version used in production (#7407)
- @davidfischer: Advertising docs tweaks (#7400)
- @stsewd: Docs: update readthedocs-sphinx-search (#7399)
- @keewis: document installing into a environment with pinned dependencies (#7397)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 32 (#7377)
- @stsewd: Builds: store build commands in storage (#7356)
- @stsewd: Guide: Cross-referencing with Sphinx (#7326)
5.9.61 Version 5.3.0

**Date** August 18, 2020

- @humitos: Remove the comma added in logs that breaks grep parsing (#7393)
- @stsewd: GitLab webhook: don’t fail on invalid payload (#7391)
- @stsewd: Task router: improve logging (#7389)
- @stsewd: External providers: better logging for GitLab (#7385)
- @stsewd: Task router: small changes (#7379)
- @stsewd: Search: return relatives URLs (#7376)
- @stsewd: Sync versions: little optimization when deleting versions (#7367)
- @agjohnson: Add feature flag to just skip the sync version task entirely (#7366)
- @agjohnson: Convert zip to list for templates (#7359)
- @stsewd: Search: implement stable API (#7255)
- @stsewd: Search: improve parser (#7233)

5.9.62 Version 5.2.3

**Date** August 04, 2020

- @davidfischer: Add a middleware for referrer policy (#7346)
- @stsewd: Footer: don’t show the version warning for external version (#7340)
- @ericholscher: Lower rank for custom install docs. (#7339)
- @benjaoming: Argument list for “python -m virtualenv” without empty strings (#7330)
- @stsewd: Docs: fix some links (#7317)
- @stsewd: Docs: little improvements on getting start docs (#7316)
- @stsewd: Search: migrate null ranks to zero (#7274)
- @stsewd: Docs: make it more clear search on subprojects (#7272)

5.9.63 Version 5.2.2

**Date** July 29, 2020

- @agjohnson: Reduce robots.txt cache TTL (#7334)
- @davidfischer: Use the privacy embed for YouTube (#7320)
- @DougCal: re-worded text on top of “Import a Repository” (#7318)
- @stsewd: Docs: make it clear the config file options are per version (#7314)
- @humitos: Feature to disable auto-generated index.md/README.rst files (#7305)
- @stsewd: Sphinx: always exclude the build directory (#7303)
- @humitos: Enable SessionAuthentication on APIv3 endpoints (#7295)
- @humitos: Allow to extend TeamManager (#7294)
5.9.64 Version 5.2.1

Date July 14, 2020

• @davidfischer: Fix a case where “tags” is interpreted as a project slug (#7284)
• @stsewd: Dashboard: little optimization (#7281)
• @stsewd: Automation rules: privacy levels (#7278)
• @stsewd: Templates: optimize permissions check (#7277)
• @agjohnson: Fix versions (#7271)
• @stsewd: Tweak priority a little more (#7270)
• @stsewd: Don’t default on the migration (#7269)
• @saadmk11: Automation rule to make versions hidden added (#7265)
• @humitos: Add is_member template filter (#7264)
• @stsewd: Docs: set ranking for some pages (#7257)
• @stsewd: Sphinx: add –keep-going when fail_on_warning is true (#7251)
• @saadmk11: Don’t allow Domain name matching production domain to be created (#7244)
• @humitos: Documentation for Single Sign-On feature on commercial (#7212)

5.9.65 Version 5.2.0

Date July 07, 2020

• @saadmk11: Version docs Typo fix (#7266)
• @stsewd: CI: fix linter (#7261)
• @GioviQ: Update manage-translations.rst (#7260)
• @ericholscher: Add additional logging for sync_repository task (#7254)
• @stsewd: Search: custom search page ranking (#7237)
# 5.9.66 Version 5.1.5

**Date** July 01, 2020

- @choldgraf: cross-linking build limitations for pr builds (#7248)
- @humitos: Allow to extend Import Project page from corporate (#7234)
- @humitos: Make RemoteRepository.full_name db_index=True (#7231)
- @stsewd: Search: tweak fuzziness (#7225)
- @ericholscher: Re-add the rst filter that got removed (#7223)

# 5.9.67 Version 5.1.4

**Date** June 23, 2020

- @stsewd: Search: index from html files for mkdocs projects (#7208)
- @stsewd: Search: recursively parse sections (#7207)
- @stsewd: Search: more general parser for html (#7204)
- @humitos: Use total_memory to calculate “time” Docker limit (#7203)
- @davidfischer: Feature flag for using latest Sphinx (#7201)
- @ericholscher: Mention that we don’t index search in PR builds (#7199)
- @davidfischer: Add a feature flag to use latest RTD Sphinx ext (#7198)
- @ericholscher: Release 5.1.3 (#7197)
- @stsewd: Search: improve results for simple queries (#7194)
- @stsewd: Search: refactor json parser (#7184)
- @stsewd: Remove unused dep (#7147)
- @agjohnson: Use theme release 0.5.0rc1 for docs (#7037)
- @humitos: Skip promoting new stable if current stable is not machine=True (#6695)

# 5.9.68 Version 5.1.3

**Date** June 16, 2020

- @davidfischer: Fix the project migration conflict (#7196)
- @stsewd: Search: fix pagination (#7195)
- @ericholscher: Document the fact that PR builds are now enabled on .org (#7187)
- @stsewd: Project: make description shorter (#7186)
- @stsewd: Migrate private versions (#7181)
- @ericholscher: Update sharing examples (#7179)
- @davidfischer: Allow projects to opt-out of analytics (#7175)
- @stsewd: Docs: install readthedocs-sphinx-search from pypi (#7174)
- @humitos: Rename API endpoint call (#7173)
• @ericholscher: Reduce logging in proxito middleware so it isn’t in Sentry (#7172)
• @ericholscher: Release 5.1.2 (#7171)
• @humitos: Use CharField.choices for Build.status_code (#7166)
• @davidfischer: Store pageviews via signals, not tasks (#7106)
• @stsewd: Move organizations models (#6776)

5.9.69 Version 5.1.2

Date June 09, 2020
• @humitos: Use CharField.choices for Build.status_code (#7166)
• @humitos: Install argh for Docker environment (#7164)
• @ericholscher: Reindex search on the reindex queue (#7161)
• @stsewd: Project search: Show original description when there isn’t highlight (#7160)
• @stsewd: Search: highlight results from projects (#7158)
• @ericholscher: Fix custom URLConf redirects (#7155)
• @ericholscher: Allow blank=True for URLConf (#7153)
• @stsewd: Fix flaky test (#7148)
• @stsewd: Search: Make total_results not null (#7145)
• @stsewd: Project: make external_builds_enabled not null (#7144)
• @saadmk11: Do not Pre-populate username field for account delete (#7143)
• @davidfischer: Add feature flag to use the stock Sphinx builders (#7141)
• @ericholscher: Move changes_files to before search indexing (#7138)
• @stsewd: Proxito middleware: reset to original urlconf after request (#7137)
• @stsewd: Search: don’t index permalinks (#7134)
• @ericholscher: Revert “Merge pull request #7101 from readthedocs/show-last-total” (#7133)
• @ericholscher: Release 5.1.1 (#7129)
• @humitos: Use “-j auto” on sphinx-build command to build in parallel (#7128)
• @humitos: De-duplicate builds (#7123)
• @stsewd: Search: refactor API to not emulate a Django queryset (#7114)
• @davidfischer: Store pageviews via signals, not tasks (#7106)
• @stsewd: Search: don’t index line numbers from code blocks (#7104)
• @humitos: Document Embed APIv2 endpoint (#7095)
• @ericholscher: Add a project-level configuration for PR builds (#7090)
• @stsewd: Remove usage of project.privacy_level (#7013)
• @pyup-bot: pyup: Scheduled weekly dependency update for week 18 (#7012)
• @stsewd: Allow to enable server side search for MkDocs (#6986)
• @stsewd: Pass the NO_COLOR env var to builder (#6981)
• @humitos: Limit concurrency in translations (#6969)
• @ericholscher: Add ability for users to set their own URLConf (#6963)

5.9.70 Version 5.1.1

Date May 26, 2020
• @stsewd: Search: show total_results from last query (#7101)
• @humitos: Add a tip in EmbedAPI to use Sphinx reference in section (#7099)
• @ericholscher: Release 5.1.0 (#7098)
• @ericholscher: Add a setting for storing pageviews (#7097)
• @humitos: Document Embed APIv2 endpoint (#7095)
• @stsewd: Footer: Check for mkdocs doctype too (#7094)
• @ericholscher: Fix the unresolver not working properly with root paths (#7093)
• @ericholscher: Add a project-level configuration for PR builds (#7090)
• @santos22: Fix tests ahead of django-dynamic-fixure update (#7073)
• @ericholscher: Add ability for users to set their own URLConf (#6963)
• @dojutsu-user: Store Pageviews in DB (#6121)
• @humitos: GitLab Integration (#3327)

5.9.71 Version 5.1.0

Date May 19, 2020
This release includes one major new feature which is Pageview Analytics. This allows projects to see the pages in their docs that have been viewed in the past 30 days, giving them an idea of what pages to focus on when updating them. This release also has a few small search improvements, doc updates, and other bugfixes as well.
• @ericholscher: Add a setting for storing pageviews (#7097)
• @stsewd: Footer: Check for mkdocs doctype too (#7094)
• @ericholscher: Fix the unresolver not working properly with root paths (#7093)
• @stsewd: Privacy levels: migrate protected versions (#7092)
• @humitos: Guide for Embed API (#7089)
• @davidfischer: Document HSTS support (#7083)
• @stsewd: Search: record queries with 0 results (#7081)
• @stsewd: Search: track total results (#7080)
• @humitos: Proxy embed URL (#7079)
• @stsewd: Search: Little refactor (#7076)
• @davidfischer: Canonical/HTTPS redirect fix (#7075)
• @santos22: Fix tests ahead of django-dynamic-fixure update (#7073)
• @stsewd: Sphinx Search: don't skip indexing if one file fails (#7071)
Read the Docs Documentation, Release 6.3.0

- @stsewd: Search: generate full link from the server side (#7070)
- @ericholscher: Fix PR builds being marked built (#7069)
- @ericholscher: Add a page about choosing between .com/.org (#7068)
- @ericholscher: Release 5.0.0 (#7064)
- @stsewd: Search: Index more content from sphinx (#7063)
- @santos22: Hide unbuilt versions in footer flyout (#7056)
- @ericholscher: Docs: Refactor and simplify our docs (#7052)
- @stsewd: Search Document: remove unused class methods (#7035)
- @stsewd: Search: iterate over valid facets only (#7034)
- @stsewd: RTDFacetedSearch: pass filters in one way only (#7032)
- @dojutsu-user: Store Pageviews in DB (#6121)

5.9.72 Version 5.0.0

Date May 12, 2020

This release includes two large changes, one that is breaking and requires a major version upgrade:

- We have removed our deprecated doc serving code that used core/views, core/symlinks, and builds/syncers (#6535). All doc serving should now be done via proxito. In production this has been the case for over a month, we have now removed the deprecated code from the codebase.
- We did a large documentation refactor that should make things nicer to read and highlights more of our existing features. This is the first of a series of new documentation additions we have planned
- @ericholscher: Fix the caching of featured projects (#7054)
- @ericholscher: Docs: Refactor and simplify our docs (#7052)
- @stsewd: Mention using ssh URLs when using private submodules (#7046)
- @ericholscher: Show project slug in Version admin (#7042)
- @stsewd: List apiv3 first (#7041)
- @stsewd: Remove CELERY_ROUTER flag (#7040)
- @stsewd: Search: remove unused taxonomy field (#7033)
- @agjohnson: Use a high time limit for celery build task (#7029)
- @ericholscher: Clean up build admin to make list display match search (#7028)
- @stsewd: Task Router: check for None (#7027)
- @stsewd: Implement repo_exists for all VCS backends (#7025)
- @stsewd: Mkdocs: Index pages without anchors (#7024)
- @agjohnson: Move docker limits back to setting (#7023)
- @humitos: Fix typo (#7022)
- @stsewd: Fix linter (#7021)
- @ericholscher: Release 4.1.8 (#7020)
- @ericholscher: Cleanup unresolver logging (#7019)
• @stsewd: Document about next when using a secret link (#7015)
• @stsewd: Remove unused field project.version_privacy_level (#7011)
• @ericholscher: Add proxito headers to redirect responses (#7007)
• @stsewd: Make hidden field not null (#6996)
• @humitos: Show a list of packages installed on environment (#6992)
• @eric-wieser: Ensure invoked Sphinx matches importable one (#6965)
• @ericholscher: Add an unresolver similar to our resolver (#6944)
• @KengoTODA: Replace “PROJECT” with project object (#6878)
• @humitos: Remove code replaced by El Proxito and stateless servers (#6535)

5.9.73 Version 4.1.8

Date May 05, 2020

This release adds a few new features and bug fixes. The largest change is the addition of hidden versions, which allows docs to be built but not shown to users on the site. This will keep old links from breaking but not direct new users there. We’ve also expanded the CDN support to make sure we’re passing headers on 3xx and 4xx responses. This will allow us to expand the timeout on our CDN.

We’ve also updated and added a good amount of documentation in this release, and we’re starting a larger refactor of our docs to help users understand the platform better.

• @ericholscher: Cleanup unresolver logging (#7019)
• @ericholscher: Add CDN to the installed apps (#7014)
• @eric-wieser: Emit a better error if no feature flag is found (#7009)
• @ericholscher: Add proxito headers to redirect responses (#7007)
• @ericholscher: Add Priority 0 to Celery (#7006)
• @stsewd: Update conftest (#7002)
• @ericholscher: Start storing JSON data for PR builds (#7001)
• @yarikoptic: Add a note if build status is not being reported (#6999)
• @stsewd: Update common (#6997)
• @davidfischer: Exclusively handle proxito HSTS from the backend (#6994)
• @humitos: Mention concurrent builds limitation in “Build Process” (#6993)
• @humitos: Show a list of packages installed on environment (#6992)
• @humitos: Document SHARE_SPHINX_DOCTREE flag (#6991)
• @humitos: Contact us via email for Feature Flags (#6990)
• @santos22: Alter field url on webhook (#6988)
• @ericholscher: Log sync_repository_task when we run it (#6987)
• @ericholscher: Remove old SSL cert warning, since they now work. (#6985)
• @agjohnson: More fixes for automatic Docker limits (#6982)
• @davidfischer: Add details to our changelog for 4.1.7 (#6978)
5.9.74 Version 4.1.7

Date  April 28, 2020

As of this release, most documentation on Read the Docs Community is now behind Cloudflare’s CDN. It should be much faster for people further from US East. Please report any issues you experience with stale cached documentation (especially CSS/JS).

Another change in this release related to how custom domains are handled. Custom domains will now redirect HTTP -> HTTPS if the Domain’s “HTTPS” flag is set. Also, the subdomain URL (eg. <project>.readthedocs.io/...) should redirect to the custom domain if the Domain’s “canonical” flag is set. These flags are configurable in your project dashboard under Admin > Domains.

Many of the other changes related to improvements for our infrastructure to allow us to have autoscaling build and web servers. There were bug fixes for projects using versions tied to annotated git tags and custom user redirects will now send query parameters.

- @ericholscher: Reduce proxito logging (#6970)
- @humitos: Log build/sync tasks when triggered (#6967)
- @humitos: Stop builders gracefully on SIGTERM (#6960)
- @stsewd: Try to fix annotated tags (#6959)
- @stsewd: Include query params in 404 redirects (#6957)
- @ericholscher: Fix the trailing slash in our repo regexs (#6956)
- @davidfischer: Add canonical to the Domain listview in the admin (#6954)
- @davidfischer: Allow setting HSTS on a per domain basis (#6953)
- @humitos: Refactor how we handle GitHub webhook events (#6949)
- @humitos: Return 400 when importing an already existing project (#6948)
- @humitos: Return max_concurrent_builds in ProjectAdminSerializer (#6946)
- @tom-doerr: Update year (#6945)
- @humitos: Revert “Use requests.head to query storage.exists” (#6941)
• @ericholscher: Release 4.1.6 (#6940)
• @stsewd: Remove note about search analytics being beta (#6939)
• @stsewd: Add troubleshooting section for dev search docs (#6933)
• @davidfischer: Index date and ID together on builds (#6926)
• @davidfischer: CAA records are not only for users of Cloudflare DNS (#6925)
• @davidfischer: Docs on supporting root domains (#6923)
• @ericholscher: Add basic support for lower priority PR builds (#6921)
• @ericholscher: Change the dashboard search to default to searching files (#6920)
• @davidfischer: Canonicalize domains and redirect in proxito (#6905)
• @zdover23: Made syntactical improvements and fixed some vocabulary issues. (#6825)

5.9.75 Version 4.1.6

Date April 21, 2020
• @stsewd: Revert usage of watchman (#6934)
• @Mariatta: Fix typo: you -> your (#6931)
• @humitos: Do not override the domain of Azure Storage (#6928)
• @humitos: Per-project concurrency and check before triggering the build (#6927)
• @davidfischer: Remove note about underscore in domain (#6924)
• @stsewd: Pass INIT to azurite (#6918)
• @humitos: Use requests.head to query storage.exists (#6917)
• @stsewd: Bring back search highlight (#6914)
• @ericholscher: Improve logging around status setting on PR builds (#6912)
• @ericholscher: Add hoverxref to our docs (#6911)
• @stsewd: Safely join storage paths (#6910)
• @humitos: Release 4.1.5 (#6909)
• @ericholscher: Fix Cache-Tag header name (#6908)
• @stsewd: Handle paths with trailing / (#6906)
• @ericholscher: Include the project slug in the PR context (#6904)
• @ericholscher: Fix single version infinite redirect (#6900)
• @humitos: Load YAML files safely (#6897)
• @humitos: Use a custom Task Router to route tasks dynamically (#6849)
• @zdover23: Made syntactical improvements and fixed some vocabulary issues. (#6825)
• @humitos: Add CORS headers to Azurite (#6784)
• @stsewd: Force to use proxied API for footer and search (#6768)
• @ericholscher: Only output debug logging from RTD app (#6717)
• @ericholscher: Add ability to sort dashboard by modified date (#6680)
5.9.76 Version 4.1.5

Date April 15, 2020

• @ericholscher: Fix Cache-Tag header name (#6908)
• @stsewd: Handle paths with trailing / (#6906)
• @ericholscher: Fix single version infinite redirect (#6900)
• @ericholscher: Release 4.1.4 (#6899)
• @humitos: On Azure .exists blob timeout, log the exception and return False (#6895)
• @ericholscher: Fix URLs like /projects/subproject from 404ing when they don’t end with a slash (#6888)
• @ericholscher: Allocate docker limits based on server size. (#6879)

5.9.77 Version 4.1.4

Date April 14, 2020

• @humitos: On Azure .exists blob timeout, log the exception and return False (#6895)
• @ericholscher: Fix URLs like /projects/subproject from 404ing when they don’t end with a slash (#6888)
• @ericholscher: Add CloudFlare Cache tags support (#6887)
• @stsewd: Update requirements (#6885)
• @stsewd: Be explicit with PUBLIC_DOMAIN setting (#6881)
• @stsewd: Allow to override project detail view (#6880)
• @ericholscher: Allocate docker limits based on server size. (#6879)
• @ericholscher: Make the status name in CI configurable via setting (#6877)
• @ericholscher: Add 12 hour caching to our robots.txt serving (#6876)
• @humitos: Filter triggered builds when checking concurrency (#6875)
• @ericholscher: Fix issue with sphinx domain types with : in them: (#6874)
• @stsewd: Make dashboard faster for projects with a lot of subprojects (#6873)
• @ericholscher: Release 4.1.3 (#6872)
• @stsewd: Don’t do unnecessary queries when listing subprojects (#6869)
• @stsewd: Optimize resolve_path (#6867)
• @stsewd: Don’t do extra query if the project is a translation (#6865)
• @stsewd: Remove private argument from resolver (#6864)
• @stsewd: Support mkdocs html pages as doctype (#6846)
• @stsewd: Reduce queries to storage to serve 404 pages (#6845)
• @stsewd: Rework custom domains docs (#6844)
• @stsewd: Add checking the github oauth app in the troubleshooting page (#6827)
• @humitos: Return full path URL (including html) on /api/v2/docurl/ endpoint (#6082)

5.9.78 Version 4.1.3

Date April 07, 2020

• @stsewd: Don’t do unnecessary queries when listing subprojects (#6869)
• @stsewd: Don’t do extra query if the project is a translation (#6865)
• @stsewd: Remove private argument from resolver (#6864)
• @ericholscher: Make development docs a bit easier to find (#6861)
• @davidfischer: Add an advertising API timeout (#6856)
• @humitos: Add more exceptions as WARNING log level (#6851)
• @humitos: Limit concurrent builds (#6847)
• @humitos: Release 4.1.2 (#6840)
• @humitos: Report build status in a smarter way (#6839)
• @stsewd: Update messages-extends to latest version (#6838)
• @humitos: Do not save pip cache when using CACHED_ENVIRONMENT (#6820)
• @stsewd: Force to reinstall package (#6817)
• @ericholscher: Denormalize from_url_without_rest onto the redirects model (#6780)
• @davidfischer: Developer docs emphasize the Docker setup (#6682)
• @davidfischer: Document setting up connected accounts in dev (#6681)
• @humitos: Return full path URL (including html) on /api/v2/docurl/ endpoint (#6082)

5.9.79 Version 4.1.2

Date March 31, 2020

• @humitos: Report build status in a smarter way (#6839)
• @stsewd: Update messages-extends to latest version (#6838)
• @humitos: Allow receiving None for template_html when sending emails (#6834)
• @ericholscher: Fix silly issue with sync_callback (#6830)
• @ericholscher: Show the builder in the Build admin (#6826)
• @ericholscher: Properly call sync_callback when there aren’t any MULTIPLE_APP_SERVERS settings (#6823)
• @stsewd: Allow to override app from where to read templates (#6821)
• @humitos: Do not save pip cache when using CACHED_ENVIRONMENT (#6820)
• @stsewd: Allow to override ProfileDetail view (#6819)
• @ericholscher: Release 4.1.1 (#6818)
• @stsewd: Force to reinstall package (#6817)
• @humitos: Show uploading state (#6816)
• @humitos: Use watchman when calling runserver in local development (#6813)
5.9.80 Version 4.1.1

Date March 24, 2020
• @stsewd: Force to reinstall package (#6817)
• @humitos: Show uploading state (#6816)
• @stsewd: Respect order when serving 404 (version -> default_version) (#6805)
• @humitos: Use storage.open API correctly for tar files (build cached envs) (#6799)
• @humitos: Check 404 page once when slug and default_version is the same (#6796)
• @humitos: Do not reset the build start time when running build env (#6794)
• @humitos: Skip .cache directory for cached builds if it does not exist (#6791)
• @ericholscher: Remove GET args from the path passed via proxito header (#6790)
• @stsewd: Check for /index on pages’ slug (#6789)
• @ericholscher: Release 4.1.0 (#6788)
• @ericholscher: Revert “Add feature flag to just completely skip sync and symlink operations (#6689)” (#6781)

5.9.81 Version 4.1.0

Date March 17, 2020
• @ericholscher: Properly proxy the Proxito headers via nginx/sendfile (#6782)
• @ericholscher: Revert “Add feature flag to just completely skip sync and symlink operations (#6689)” (#6781)
• @humitos: Upgrade django-storages to support URLs with more http methods (#6771)
• @davidfischer: Use the hotfixed version of django-messages-extends (#6767)
• @ericholscher: Release 4.0.3 (#6766)
• @stsewd: Document usage or pytest marks (#6764)
• @humitos: Pull/Push cached environment using storage (#6763)
• @stsewd: Refactor search view to make use of permission_classes (#6761)
• @stsewd: Run proxito tests with proxito (#6714)
• @stsewd: Proxy footer api on docs’ domains (#6630)
5.9.82 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)

5.9.83 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)

5.9.84 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)
Read the Docs Documentation, Release 6.3.0

- @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)

5.9.85 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)

5.9.86 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) (#6554)
• @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)

5.9.87 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)

5.9.88 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)

5.9. Changelog
5.9.89 Version 3.11.4

Date January 28, 2020

- @humitos: Disable django debug toolbar in El Proxito (#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 Proxito (#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)
- @saadmk11: 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)
5.9.90 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)
- @dibyaaaaaax: 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)
- @Blackciphen101: 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 LocalBuildEnviroment (#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)

5.9.91 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)

5.9. Changelog
5.9.92 Version 3.11.1

- @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_NORELOAD 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 (#6409)
- @humitos: Use WORKDIR to cd into a directory in Dockerfile (#6407)
- @humitos: Serve non-html files from nginx (X-Accel-Redirect) (#6404)
- @humitos: Perform redirects at DB level (#6398)

5.9.93 Version 3.11.0

- @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)
• @saadmk11: Refactor Subproject validation to use it for Forms and API (#6285)
• @saadmk11: Refactor Gold Views (#6272)
• @stsewd: Add docs for automatin rules (#6072)

5.9.94 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)
• @saadmk11: Show only users projects in the APIv3 browsable 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)

5.9.95 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)
• @segevfiner: Remove a stray backtick from import-guide.rst (#6362)
• @stsewd: Don’t compare inactive or non build versions (#6361)
• @stsewd: Fix test (#6358)

5.9. Changelog 323
- @ericholscher: Change the default of proxied_api_host to api_host (#6355)
- @stsewd: Don’t 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)
- @saadmk11: 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 doccs version detail (api v3) (#6259)
- @stsewd: Merge #6176 to master (#6258)
- @humitos: Remove privacy_level field from AIPv3 (#6257)
- @saadmk11: Redirect /projects/ URL to /dashboard/ (#6255)
- @davidfischer: Allow project badges for private version (#6252)
- @stsewd: Add pub_date to project admin (#6244)
- @saadmk11: Allow only post requests for delete views (#6242)
- @Iamshankhadeep: Changing created to modified time (#6234)
- @ericholscher: Initial stub of proxito (#6226)
- @saadmk11: 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)
• @tamshankhadeep: Adding RTD prefix for docker only in setting.py and all other places where is needed (#6040)
• @stsewd: Design doc for organizations (#5958)

5.9.96 Version 3.8.0

**Date** October 09, 2019

• @stsewd: Update docs version detail (api v3) (#6259)
• @stsewd: Merge #6176 to master (#6258)
• @humitos: Remove privacy_level field from APIv3 (#6257)
• @saadmk11: 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)
• @saadmk11: 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)
• @saadmk11: Only Build Active Versions from Build List Page Form (#6205)
• @saadmk11: Make raw_config private (#6199)
• @tamshankhadeep: 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)

5.9.97 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)
• @saadmk11: 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)
• @saadmk11: Integration Re-sync Bug Fix (#6124)
• @stsewd: Don’t log BuildEnvironmentWarning as error (#6112)
• @dojutsu-user: Add Search Guide (#6101)
• @saadmk11: Add PR Builder guide to docs (#6093)
• @dojutsu-user: Record search queries smartly (#6088)
• @dojutsu-user: Remove ‘highlight’ URL param from search results (#6087)
5.9.98 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)

5.9.99 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)
- @saadmk11: GitLab Build Status Reporting for PR Builder (#6076)
- @davidfischer: Update ad details docs (#6074)
- @davidfischer: Gold makes projects ad-free again (#6073)
- @saadmk11: Auto Sync and Re-Sync for Manually Created Integrations (#6071)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 32 (#6067)
- @saadmk11: 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)
- @saadmk11: Update GitLab Webhook creating to enable merge request events (#6055)
- @ericholscher: Release 3.7.2 (#6054)

5.9. Changelog
5.9.100 Version 3.7.2

Date August 08, 2019

• @dojutsu-user: Update feature flags docs (#6053)
• @saadmk11: Add indelx.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)

5.9.101 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)
• @saadmk11: 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)
• @ericholscher: Update get_absolute_url for External Versions (#6020)
• @dojutsu-user: Search analytics (#6019)
• @saadmk11: 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)
• @saadmk11: 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 psycopg2 (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)
5.9.102 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)
• @saadmk11: Small Changes to PR Builder Code (#5948)
• @saadmk11: 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)
• @saadmk11: 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)

5.9.103 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)
• @saadmk11: 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)
• @saadmk11: 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)
• @saadmk11: 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)
• @saadmk11: Add check for external version in conf.py.tmpl for warning banner (#5900)
• @humitos: Update APIv3 documentation with latest changes (#5895)

5.9.104 Version 3.6.0

• @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)
• @saadmk11: 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)
• @saadmk11: Update build list and detail page UX (#5916)
• @humitos: These Project’s methods are not used (#5915)
• @saadmk11: Github Status reporting Test fix (#5914)

5.9. Changelog
• @humitos: APIv3 endpoint to manage Environment Variables (#5913)
• @humitos: Split APIv3 tests on different files (#5911)
• @saadmk11: 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)
• @saadmk11: 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)
• @saadmk11: 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 rtfd to readthedocs (#5871)
• @humitos: APIv3 refactor some fields (#5868)
• @saadmk11: 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)

5.9.105 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)
• @saadmk11: 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)
• @saadmk11: Add Build managers and Update Build Querysets. (#5779)
• @davidfischer: Project advertising page/form update (#5777)
• @davidfischer: Update docs around opt-out of ads (#5776)
• @saadmk11: Sitemap sort order priorities updated (#5724)
• @dojutsu-user: [Design Doc] In Doc Search UI (#5707)
• @saadmk11: 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)

5.9.106 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)

5.9.107 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)
• @saadmk11: 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)
• @wilv: 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 latexmx builder to ignore error codes (#5744)
• @ericholscher: Hide the Code API search in the UX for now. (#5743)
• @davidfischer: Add init.py under readthdocs/api (#5742)
• @dojutsu-user: Fix design docs missing from toctree (#5741)
• @ericholscher: Release 3.5.0 (#5740)
• @saadmk11: Pytest Timezone Warning Fixed (#5739)
• @humitos: Filter by projects with no banned users (#5733)
• @davidfischer: Fix the sidebar ad color (#5731)
• @saadmk11: 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)
5.9.108 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)
- @saadmk11: 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)
- @saadmk11: pylint fix for notifications, restapi and config (#5664)
- @saadmk11: pylint fix for readthedocs.search (#5663)
- @saadmk11: pylint fix for readthedocs.projects (#5662)
- @saadmk11: 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)
- @saadmk11: pylint fix for readthedocs.core (#5650)
- @pyup-bot: pyup: Scheduled weekly dependency update for week 17 (#5645)
- @saadmk11: Serve custom 404 pages from Django (#5644)
- @yarons: Typo fix (#5642)
- @saadmk11: 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)
• @saadmk11: Standardizing the use of settings directly (#5632)
• @saadmk11: 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)
• @saadmk11: Handle 401, 403 and 404 when setting up webhooks (#5589)
• @saadmk11: Unify usage of settings and remove the usage of getattr for settings (#5588)
• @saadmk11: Note about admin page in the docs (#5585)
• @humitos: Remove USE_SETUPTOOLS_LATEST feature flag (#5578)
• @saadmk11: 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)
• @saadmk11: “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)
• @saadmk11: 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)
5.9.109 Version 3.4.2

Date April 22, 2019

- @ericholscher: Add explicit egg version to unicode-slugify (#5612)
- @saadmk11: Update Environmental Variable character limit (#5597)
- @davidfischer: Add meta descriptions to top documentation (#5593)
- @stsewd: Ignore pytest-xdist from pyupdate (#5590)
- @saadmk11: 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)
- @saadmk11: Sitemap assumes that all versions are translated Fix. (#5535)
- @saadmk11: 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)
- @saadmk11: Don’t allow to create subprojects with same alias (#5404)
- @saadmk11: Improve project translation listing Design under admin tab (#5380)

5.9.110 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)
- @saadmk11: Sitemap assumes that all versions are translated Fix. (#5535)
- @saadmk11: 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)
• @saadmk11: Make /random/ path work (#5496)
• @humitos: Typo on conf.pytmpl (#5495)
• @rsrhrc: 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)
• @saadmk11: Dashboard screen performance fix (#5471)
• @saadmk11: 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)

5.9.111 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)
• @saadmk11: 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)
• @saadmk11: Remove pytest warnings (#5346)
• @davidfischer: Remove the v1 API (#5293)
• @stsewd: Remove doctype from resolver (#5230)
• @humitos: Implementation of APIv3 (#4863)

5.9.112 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)

5.9.113 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)
• @saadmk11: 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)
• @saadmk11: #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)

5.9. Changelog
Read the Docs Documentation, Release 6.3.0

- @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)

5.9.114 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)

5.9.115 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)
- @discdiver: 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)

5.9.116 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)

5.9.117 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 mergeable (#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)

5.9. Changelog
5.9.118 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)
5.9.119 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)

5.9.120 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)
• @discdiver: 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)

5.9.121 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)

5.9.122 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)
5.9.123 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)
- @chirathr: Webhook notification URL size validation check (#3680)
5.9.124 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)

5.9.125 Version 2.8.0

Date October 30, 2018

Major change is an upgrade to Django 1.11.
- @humitos: Cleanup old code (remove old_div) (#4817)
- @humitos: Remove unnecessary migration (#4806)
- @humitos: Feature flag to make readthedocs theme default on MkDocs docs (#4802)
- @stsewd: Add codecov badge (#4799)
- @humitos: Pin missing dependency for the MkDocs guide compatibility (#4798)
- @ericholscher: Release 2.7.2 (#4796)
- @humitos: Do not log as error a webhook with an invalid branch name (#4779)
- @ericholscher: Run travis on release branches (#4763)
- @ericholscher: Remove Eric & Anthony from ADMINS & MANAGERS settings (#4762)
- @stsewd: Don’t use RequestsContext (#4759)
- @davidfischer: Django 1.11 upgrade (#4750)
- @stsewd: Dropdown to select Advanced Settings (#4710)
- @stsewd: Remove hardcoded constant from config module (#4704)
- @stsewd: Update tastypie (#4325)
5.9.126 Version 2.7.2

* Date October 23, 2018

- **@stsewd**: Update to Django 1.10 (#4319)
- **@stsewd**: Upgrade logs from debug on middleware (#4769)
- **@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**: 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)
5.9.127 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)
5.9.128 Version 2.7.0

Date September 29, 2018

Reverted, do not use

5.9.129 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)

5.9.130 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)

5.9.131 Version 2.6.4

Date August 29, 2018
• @stsewd: Update tests failing on master (#4575)
• @davidfischer: Add a flag to disable docsearch (#4570)
• @stsewd: Fix nested syntax in docs (#4567)
• @stsewd: Fix incorrect reraise (#4566)
• @davidfischer: Add a note about specifying the version of build tools (#4562)
• @davidfischer: Serve badges directly from local filesystem (#4561)
• @humitos: Build JSON artifacts in HTML builder (#4554)
• @humitos: Route task to proper queue (#4553)
• @humitos: Sanitize BuildCommand.output by removing NULL characters (#4552)
• @davidfischer: Fix changelog for 2.6.3 (#4548)
• @ericholscher: Remove hiredis (#4542)
• @davidfischer: Use the STATIC_URL for static files to avoid redirection (#4522)
• @stsewd: Update docs about build process (#4515)
• @StefanoChiodino: Allow for period as a prefix and yaml extension for config file (#4512)
• @AumitLeon: Update information on mkdocs build process (#4508)
• @humitos: Fix Exact Redirect to work properly when using $rest keyword (#4501)
• @humitos: Mark some BuildEnvironmentError exceptions as Warning and do not log them (#4495)
• @xrmx: projects: don’t explode trying to update UpdateDocsTaskStep state (#4485)
• @humitos: Note with the developer flow to update our app translations (#4481)
• @humitos: Add trimmed to all multilines blocktrans tags (#4480)
• @humitos: Example and note with usage of trimmed option in blocktrans (#4479)
• @humitos: Update Transifex resources for our documentation (#4478)
• @humitos: Documentation for Manage Translations (#4470)
• @stsewd: Port https://github.com/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)

5.9.132 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: 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)

5.9.133 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)

5.9.134 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)

5.9.135 Version 2.6.0

Date July 16, 2018
• Adds initial support for HTTPS on custom domains
• @stsewd: Revert “projects: serve badge with same protocol as site” (#4353)
• @davidfischer: Do not overwrite sphinx context variables feature (#4349)
• @stsewd: Calrify docs about how rtd select the stable version (#4348)
• @davidfischer: Remove the UID from the GA measurement protocol (#4347)
• @stsewd: Fix error in command (#4345)
• @davidfischer: Improvements for the build/version admin (#4344)
• @safwanrahman: [Fix #4265] Porting frontend docsearch to work with new API (#4340)
• @ktreyer: 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)

5.9.136 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)
5.9.137 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)
- @huminos: Downgrade docker to 3.1.3 because of timeouts in EXEC call (#4241)
- @stsewd: Move parser tests from rtd-build repo (#4225)
- @huminos: Handle revoked oauth permissions by the user (#4074)
- @huminos: Allow to hook the initial build from outside (#4033)

5.9.138 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)
- @huminos: Upgrade celery to avoid AttributeError:asynclcelery (#4185)
- @stsewd: Prepare code for custo mkdocs.yaml location (#4184)
- @agjohnson: Updates to our triage guidelines (#4180)
- @davidfischer: Server side analytics (#4131)
5.9.139 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)

5.9.140 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 CHANGELGOG (#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)
5.9.141 Version 2.3.14

Date May 30, 2018

- @ericholscher: Remove CSS override that doesn’t exist. (#4165)
- @davidfischer: Include a DMCA request template (#4164)
- @davidfischer: No CSRF cookie for docs pages (#4153)
- @davidfischer: Small footer rework (#4150)
- @stsewd: Fix prospector dependencies (#4149)
- @ericholscher: Remove deploy directory which is unused. (#4147)
- @stsewd: Use autosectionlabel extension (#4146)
- @davidfischer: Add Intercom to the privacy policy (#4145)
- @humitos: Minimum refactor to decide_if_cors (#4143)
- @stsewd: Ignore migrations from coverage report (#4141)
- @stsewd: 5xx status in old webhooks (#4139)
- @davidfischer: Fix with Lato Bold font (#4138)
- @davidfischer: Release 2.3.13 (#4137)
- @davidfischer: Build static assets (#4136)
- @xrmx: oauth/services: correct error handling in paginate (#4134)
- @xrmx: oauth/services: don’t abuse log.exception (#4133)
- @cedk: Use quiet mode to retrieve branches from mercurial (#4114)
- @humitos: Add has_valid_clone and has_valid_webhook to ProjectAdminSerializer (#4107)
- @stsewd: Put the rtd extension to the beginning of the list (#4054)
- @stsewd: Use gitpython for tags (#4052)
- @davidfischer: Do Not Track support (#4046)
- @stsewd: Set urlconf to None after changing SUBDOMAIN setting (#4032)
- @humitos: Fix /404/ testing page (#3976)
- @xrmx: Fix some tests with postgres (#3958)
- @xrmx: Fixup DJANGO_SETTINGS_SKIP_LOCAL in tests (#3899)
- @xrmx: templates: mark a few more strings for translations (#3869)
- @ze: Make search bar in dashboard have a more clear message. (#3844)
- @varunotelli: Pointed users to Python3.6 (#3817)
- @stsewd: [RDY] Fix tests for environment (#3764)
- @ajatprabha: Ticket #3694: rename owners to maintainers (#3703)
- @SanketDG: Refactor to replace old logging to avoid mangling (#3677)
- @stsewd: Add rstcheck to CI (#3624)
- @techtonik: Update Git on prod (#3615)
- @stsewd: Allow to hide version warning (#3595)
• @cclauss: Modernize Python 2 code to get ready for Python 3 (#3514)
• @stsewd: Consistent version format (#3504)

5.9.142 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)

5.9.143 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)
5.9.144 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)

5.9.145 Version 2.3.10

Date April 24, 2018

• @humitos: Downgrade docker to 3.1.3 (#4003)

5.9.146 Version 2.3.9

Date April 20, 2018

• @agjohnson: Fix recursion problem more generally (#3989)
5.9.147 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)

5.9.148 Version 2.3.7

Date April 19, 2018

• @humitos: Fix server_error_500 path on single version (#3975)
• @davidfischer: Fix bookmark app lint failures (#3969)
• @humitos: Use latest setuptools (39.0.1) by default on build process (#3967)
• @ericholscher: Fix exact redirects. (#3965)
• @humitos: Make resolve_domain work when a project is subproject of itself (#3962)
• @humitos: Remove django-celery-beat and use the default scheduler (#3959)
• @xrmx: Fix some tests with postgres (#3958)
• @davidfischer: Add advertising details docs (#3955)
• @humitos: Use pur to upgrade python packages (#3953)
• @ze: Make adjustments to Projects page (#3948)
• @davidfischer: Small change to Chinese language names (#3947)
• @agjohnson: Don’t share state in build task (#3946)
• @davidfischer: Fixed footer ad width fix (#3944)
• @humitos: Allow extend Translation and Subproject form logic from corporate (#3937)
• @humitos: Resync valid webhook for project manually imported (#3935)
• @humitos: Resync webhooks from Admin (#3933)
• @humitos: Fix attribute order call (#3930)
• @humitos: Mention RTD in the Project URL of the issue template (#3928)
• @davidfischer: Correctly report mkdocs theme name (#3920)
• @xrmx: Fixup DJANGO_SETTINGS_SKIP_LOCAL in tests (#3899)
• @davidfischer: Show an adblock admonition in the dev console (#3894)
• @stsewd: Fix view docs link (#3882)
• @xrmx: templates: mark a few more strings for translations (#3869)
• @ze: Update quickstart from README (#3847)
• @vidartf: Fix page redirect preview (#3811)
• @stsewd: [RDY] Fix requirements file lookup (#3800)
• @aasis21: Documentation for build notifications using webhooks. (#3671)
• @mashrikt: [#2967] Scheduled tasks for cleaning up messages (#3604)
• @stsewd: Show URLs for exact redirect (#3593)
• @marcelstoer: Doc builder template should check for mkdocs_page_input_path before using it (#3536)
• @Code0x58: Document creation of slumber user (#3461)

5.9.149 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)

5.9.150 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)
5.9.151 Version 2.3.4

- Release for static assets

5.9.152 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 across 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)

5.9.153 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.

5.9.154 Version 2.3.1

- @humitos: Update instance and model when record_as_success (#3831)
- @agjohnson: Bump docker -> 3.1.3 (#3828)
- @Doug-AWS: Pip install note for Windows (#3827)
- @himanshutejwani12: Update index.rst (#3824)
- @ericholscher: Reorder GSOC projects, and note priority order (#3823)
- @agjohnson: Autolint cleanup for #3821 (#3822)
- @agjohnson: Add temporary method for skipping submodule checkout (#3821)
- @stsewd: Pin astroid to fix linter issue on travis (#3816)
- @varunotelli: Update install.rst dropped the Python 2.7 only part (#3814)
- @xrmx: Update machine field when activating a version from project_version_detail (#3797)
- @humitos: Allow members of “Admin” Team to wipe version envs (#3791)
• @ericholscher: Add sustainability api to CORS (#3782)
• @durwasa-chakraborty: Fixed a grammatical error (#3780)
• @humitos: Trying to solve the end line character for a font file (#3776)
• @stsewd: Fix tox env for coverage (#3772)
• @bansalnitish: Added eslint rules (#3768)
• @davidfischer: Use sustainability api for advertising (#3747)
• @davidfischer: Add a sustainability API (#3672)
• @humitos: Upgrade django-pagination to a “maintained” fork (#3666)
• @humitos: Project updated when subproject modified (#3649)
• @davidfischer: Anonymize IP addresses for Google Analytics (#3626)
• @humitos: Improve “Sharing” docs (#3472)
• @humitos: Upgrade docker-py to its latest version (docker==3.1.1) (#3243)
• @humitos: Upgrade all packages using pur tool (#2916)
• @rixx: Fix page redirect preview (#2711)

5.9.155 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)
• @rajuja373: #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)
**5.9.156 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)
- @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)

**5.9.157 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)

5.9.158 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)

5.9.159 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)

5.9.160 Version 2.1.4

• @davidfischer: Add programming language to API/READTHEDOCSDATA (#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 “resetting” misspelling. (#3487)
• @ericholscher: Add David to dev team listing (#3485)
• @ericholscher: Check to make sure changes exist in BitBucket pushes (#3480)
• @ericholscher: Use semvar for readthedocs-build to make bumping easier (#3475)
• @davidfischer: Add programming languages (#3471)
• @humitos: Remove TEMPLATE_LOADERS since it’s the default (#3469)
• @Code0x58: Minor virtualenv upgrade (#3463)
• @humitos: Remove invite only message (#3456)
• @maxirus: Adding to Install Docs (#3455)
• @stsewd: Fix a little typo (#3448)
• @stsewd: Better autogenerated index file (#3447)
• @stsewd: Better help text for privacy level (#3444)
• @msyriac: Broken link URL changed fixes #3442 (#3443)
• @ericholscher: Fix git (#3441)
• @ericholscher: Properly slugify the alias on Project Relationships. (#3440)
• @stsewd: Don’t show “build ideas” to unprivileged users (#3439)
• @Blendify: Docs: Point Theme docs to new website (#3438)
• @humitos: Do not use double quotes on git command with –format option (#3437)
• @ericholscher: Hack in a fix for missing version slug deploy that went out a while back (#3433)
• @humitos: Check versions used to create the venv and auto-wipe (#3432)
• @ericholscher: Upgrade psycopg2 (#3429)
• @humitos: Fix “Edit in Github” link (#3427)
• @ericholscher: Add celery theme to supported ad options (#3425)
• @humitos: Link to version detail page from build detail page (#3418)
• @humitos: Move wipe button to version detail page (#3417)
• @humitos: Show/Hide “See paid advertising” checkbox depending on USE_PROMOS (#3412)
• @benjaoming: Strip well-known version component origin/ from remote version (#3377)
• @humitos: Remove warnings from code (#3372)
• @ericholscher: Add docker image from the YAML config integration (#3339)
• @humitos: Show proper error to user when conf.py is not found (#3326)
• @humitos: Simple task to finish inactive builds (#3312)
• @techtonik: Fix Edit links if version is referenced by annotated tag (#3302)
• @Riyuzakii: changed <strong> from html to css (#2699)

5.9.161 Version 2.1.3

• @ericholscher: Upgrade psycopg2 (#3429)
• @humitos: Fix “Edit in Github” link (#3427)
• @ericholscher: Add celery theme to supported ad options (#3425)
• @ericholscher: Only build travis push builds on master. (#3421)
• @ericholscher: Add concept of dashboard analytics code (#3420)
• @humitos: Use default avatar for User/Orgs in OAuth services (#3419)

5.9. Changelog
• @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)
• @acraze: 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)
• @jonkarr: 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 <strong> from html to css (#2699)
• @tjanez: Support specifying ‘python setup.py build_sphinx’ as an alternative build command (#1857)
• @bdarnell: Broken edit links (#1637)

5.9.162 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)

5.9.163 Version 2.1.1

Release information missing

5.9.164 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)
Read the Docs Documentation, Release 6.3.0

• @ericholscher: Suggest wiping your environment to folks with bad build outcomes. (#3347)
• @huminos: GitLab Integration (#3327)
• @jimfulton: Draft policy for claiming existing project names. (#3314)
• @agjohnson: More logic changes to error reporting, cleanup (#3310)
• @safwanrahaman: [Fix #3182] Better user deletion (#3214)
• @ericholscher: Better User deletion (#3182)
• @RichardLitt: Add Needed: replication label (#3138)
• @josejrobles: Replaced usage of deprecated function get_fields_with_model with new ... (#3052)
• @ericholscher: Don’t delete the subprojects directory on sync of superproject (#3042)
• @andrew: Pass query string when redirecting, fixes #2595 (#3001)
• @saily: Add GitLab repo sync and webhook support (#1870)
• @destroyerofbuilds: Setup GitLab Web Hook on Project Import (#1443)
• @takotuesday: Add GitLab Provider from django-allauth (#1441)

5.9.165 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)
• @huminos: Do not split before first argument (#3333)
• @huminos: Update docs for pre-commit (auto linting) (#3332)
• @huminos: Take preference of tags over branches when selecting the stable version (#3331)
• @huminos: Add prospector as a pre-commit hook (#3328)
• @andrewgodwin: “stable” appearing to track future release branches (#3268)
• @huminos: 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)
5.9.166 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: https://docs.celeryproject.org/en/3.1/whatsnew-3.1.html
- Updated tastypie from 0.11.1 to current master (commit 1e1aff3dd4dcd21669e9c68bd7681253b286b856) as 0.11.x is not compatible with Django 1.8. No surprises expected but we should ask for a proper release, see release notes: https://github.com/django-tastypie/django-tastypie/blob/master/docs/release_notes/v0.12.0.rst
- Updated django-oauth from 0.16.1 to 0.21.0. No surprises expected, see release notes in the docs and finer grained in the repo
- Updated django-guardian from 1.2.0 to 1.3.0 to gain Django 1.8 support. No surprises expected, see release notes: https://github.com/lukaszb/django-guardian/blob/devel/CHANGES
- Using django-formtools instead of removed django.contrib.formtools now. Based on the Django release notes, these modules are the same except of the package name.
- Updated pytest-django from 2.6.2 to 2.8.0. No tests required, but running the testsuite :smile:
- Updated psycopg2 from 2.4 to 2.4.6 as 2.4.5 is required by Django 1.8. No trouble expected as Django is the layer between us and psycopg2. Also it’s only a minor version upgrade. Release notes: http://initd.org/psycopg/docs/news.html#what-s-new-in-psycopg-2-4-6
- Added django.setup() to conf.py to load django properly for doc builds.
- Added migrations for all apps with models in the readthedocs/ directory

**Deployment Notes**

After you have updated the code and installed the new dependencies, you need to run these commands on the server:

```
python manage.py migrate contenttypes
python manage.py migrate projects 0002 --fake
python manage.py migrate --fake-initial
```

Locally I had trouble in a test environment that pip did not update to the specified commit of tastypie. It might be required to use pip install -U -r requirements/deploy.txt during deployment.
Development Update Notes

The readthedocs developers need to execute these commands when switching to this branch (or when this got merged into master):

- **Before updating** please make sure that all migrations are applied:
  
  ```
  python manage.py syncdb
  python manage.py migrate
  ```

- Update the codebase: `git pull`
- You need to update the requirements with `pip install -r requirements.txt`
- Now you need to fake the initial migrations:
  
  ```
  python manage.py migrate contenttypes
  python manage.py migrate projects 0002 --fake
  python manage.py migrate --fake-initial
  ```

5.10 About Read the Docs

Read the Docs is a C Corporation registered in Oregon. Our bootstrapped company is owned and fully controlled by the founders, and fully funded by our customers and advertisers. This allows us to **focus 100% on our users**.

We have two main sources of revenue:

- Read the Docs for Business - where we provide a valuable paid service to companies.
- Read the Docs Community - where we provide a free service to the open source community, funded via Ethical Ads.

We believe that having both paying customers and ethical advertising is the best way to create a sustainable platform for our users. We have built something that we expect to last a long time, and we are able to make decisions based only on the best interest of our community and customers.

All of the source code for Read the Docs is open source. You are welcome to **contribute** the features you want or run your own instance. We should note that we generally only support our hosted versions as a matter of our philosophy.

We owe a great deal to the open source community that we are a part of, so we provide free ads via our community ads program. This allows us to give back to the communities and projects that we support and depend on.

We are proud about the way we manage our company and products, and are glad to have you on board with us in this great documentation journey.

5.11 Read the Docs Team

readthedocs.org is the largest open source documentation hosting service. Today we:

- Serve over **55 million pages** of documentation a month
- Serve over **40 TB** of documentation a month
- Host over **80,000 open source projects** and support over **100,000 users**

Read the Docs is provided as a free service to the open source community, and we hope to maintain a reliable and stable hosting platform for years to come.
5.11.1 Staff

The members of the Staff work full time on the service, and we are also honored to have several external contributors. We mainly fund our operations through advertising and corporate-hosted documentation with Read the Docs for Business, and we are supported by a number of generous sponsors.

<table>
<thead>
<tr>
<th>Eric Holscher</th>
<th>Anthony Johnson</th>
</tr>
</thead>
<tbody>
<tr>
<td>All teams</td>
<td>All teams</td>
</tr>
<tr>
<td>Manuel Kaufmann</td>
<td>Santos Gallegos</td>
</tr>
<tr>
<td>Backend, Operations, Support</td>
<td>Backend, Operations, Support</td>
</tr>
<tr>
<td>Juan Luis Cano</td>
<td>Ana Costa</td>
</tr>
<tr>
<td>Advocacy, Support</td>
<td>Frontend</td>
</tr>
</tbody>
</table>

Teams

- The **Backend Team** folks develop the Django code that powers the backend of the project.
- The members of the **Frontend Team** care about UX, CSS, HTML, and JavaScript, and they maintain the project UI as well as the Sphinx theme.
- As part of operating the site, members of the **Operations Team** maintain a 24/7 on-call rotation. This means that folks have to be available and have their phone in service.
- The members of the **Advocacy Team** spread the word about all the work we do, and seek to understand the users priorities and feedback.
- The **Support Team** helps our thousands of users using the service, addressing tasks like resetting passwords, enable experimental features, or troubleshooting build errors.

**Note:** Please don’t email us personally for support on Read the Docs. You can [use our support form](#) for any issues you may have.
5.11.2 Major Contributors

The code that powers the Read the Docs platform, as well as many other related projects in our GitHub organization, are open source, and therefore anybody can contribute.

Our platform code has over a hundred contributors, which makes us extremely proud and thankful. In addition, several contributors have performed ongoing maintenance on several subprojects over the years:

- Aaron Carlisle for our Sphinx theme.
- Ashley Whetter for our autoapi Sphinx extension.

We know that we’re missing a large number of people who have contributed in major ways to our various projects. Please let us know if you feel that you should be on this list, and aren’t!

5.12 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.

5.12.1 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

5.12.2 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
5.12.3 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.

5.13 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 it’s 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.

5.14 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.
5.14.1 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.

**Are you a marketer?**

We built a whole business around privacy-focused advertising. If you’re trying to reach developers, we have a network of hand-approved sites (including Read the Docs) where your ads are shown.

**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.
- After proving our ad model as a way to fund open source and building our ad serving infrastructure, we launched the EthicalAds network to help other projects be sustainable.

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.

5.14.2 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 (e.g. 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.

**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.

**Ad serving infrastructure**

Our entire ad server is open source, so you can inspect how we’re doing things. We believe strongly in open source, and we practice what we preach.
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.
- Project maintainers can completely disable analytics on their own projects. Follow the steps in Disabling Google Analytics on your project.

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.
5.14.3 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 when logged in by becoming a Gold member or a Supporter. Gold members may also completely remove advertising for all visitors to their projects. 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.
5.15 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.

5.15.1 Current sponsors

- **AWS** - They cover all of our hosting expenses every month. This is a pretty large sum of money, averaging around $5,000/mo.
- **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.
- **Chan Zuckerberg Initiative** - Through their “Essential Open Source Software for Science” programme, they fund our ongoing efforts to improve scientific documentation and make Read the Docs a better service for scientific projects.
- **You?** (Email us at hello@readthedocs.org for more info)

5.15.2 Past sponsors

- Microsoft Azure
- Python Software Foundation
- Revsys
- Mozilla Web Dev
- Django Software Foundation
- Lab305
- Twilio
- Rackspace
- Mozilla

5.15.3 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.
5.16 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.

5.16.1 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*.

---

**Warning:** Owners, Members and Teams behave differently if you are using [SSO with VCS provider (GitHub, Bitbucket or GitLab)](https://docs.readthedocs.io/en/latest/organizations.html#ssos)
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.

5.16.2 Single Sign-On

**Note:** This feature only exists on Read the Docs for Business.

Single sign-on is supported on Read the Docs for Business for all users. SSO (single sign-on) will allow you to grant permissions to your organization’s projects in an easy way.

Currently, we support two different types of single sign-on:

- Authentication and authorization are managed by the identity provider (e.g. GitHub, Bitbucket or GitLab)
- Authentication (only) is managed by the identity provider (e.g. an active Google Workspace account with a verified email address)

Users can log out by using the Log Out link in the RTD flyout menu.

- **SSO with VCS provider (GitHub, Bitbucket or GitLab)**
  - Grant access to read the documentation
  - Grant access to administrate a project
  - Grant access to import a project
  - Revoke access to a project

- **SSO with Google Workspace**
  - Grant access to read a project
  - Grant access to administer a project
  - Grant access to users to import a project
  - Revoke user’s access to a project
  - Revoke user’s access to all the projects
SSO with VCS provider (GitHub, Bitbucket or GitLab)

Using an identity provider that supports authentication and authorization allows you to manage who has access to projects on Read the Docs, directly from the provider itself. If a user needs access to your documentation project on Read the Docs, that user just needs to be granted permissions in the VCS repository associated with the project.

You can enable this feature in your organization by going to your organization’s detail page > Settings > Authorization and selecting GitHub, GitLab or Bitbucket as provider.

Note the users created under Read the Docs must have their GitHub, Bitbucket or GitLab account connected in order to make SSO work. You can read more about granting permissions on GitHub.

**Warning:** Once you enable this option, your existing Read the Docs teams will not be used.

Grant access to read the documentation

By granting read (or more) permissions to a user in the VCS repository you are giving access to read the documentation of the associated project on Read the Docs to that user.

Grant access to administrate a project

By granting write permission to a user in the VCS repository you are giving access to read the documentation and to be an administrator of the associated project on Read the Docs to that user.

Grant access to import a project

When SSO with a VCS provider is enabled, only owners of the Read the Docs organization can import projects. Adding users as owners of your organization will give them permissions to import projects.

Note that to be able to import a project, that user must have admin permissions in the VCS repository associated.

Revoke access to a project

If a user should not have access anymore to a project, for any reason, a VCS repository’s admin (e.g. user with Admin role on GitHub for that specific repository) can revoke access to the VCS repository and this will be automatically reflected in Read the Docs.

The same process is followed in case you need to remove admin access, but still want that user to have access to read the documentation. Instead of revoking access completely, just need lower down permissions to read only.

SSO with Google Workspace

**Note:** Google Workspace was formerly called G Suite

Using your company’s Google email address (e.g. employee@company.com) allows you to manage authentication for your organization’s members. As this identity provider does not provide authorization over each repositories/projects per user, permissions are managed by the internal Read the Docs’s teams authorization system.
By default, users that sign up with a Google account do not have any permissions over any project. However, you can define which teams users matching your company’s domain email address will auto-join when they sign up. Read the following sections to learn how to grant read and admin access.

You can enable this feature in your organization by going to your organization’s detail page > Settings > Authorization and selecting Google as provider and specifying your Google Workspace domain in the Domain field.

**Grant access to read a project**

You can add a user under a read-only team to grant **read** permissions to all the projects under that team. This can be done under your organization’s detail page > Teams > Read Only > Invite Member.

To avoid this repetitive task for each employee of your company, the owner of the Read the Docs organization can mark one or many teams for users matching the company’s domain email to join these teams automatically when they sign up.

For example, you can create a team with the projects that all employees of your company should have access to and mark it as Auto join users with an organization’s email address to this team. Then all users that sign up with their employee@company.com email will automatically join this team and have **read** access to those projects.

**Grant access to administer a project**

You can add a user under an admin team to grant **admin** permissions to all the projects under that team. This can be done under your organization’s detail page > Teams > Admins > Invite Member.

**Grant access to users to import a project**

Making the user member of any admin team under your organization (as mentioned in the previous section), they will be granted access to import a project.

Note that to be able to import a project, that user must have **admin** permissions in the GitHub, Bitbucket or GitLab repository associated, and their social account connected with Read the Docs.

**Revoke user’s access to a project**

To revoke access to a project for a particular user, you should remove that user from the team that contains that project. This can be done under your organization’s detail page > Teams > Read Only and click Remove next to the user you want to revoke access.

**Revoke user’s access to all the projects**

By disabling the Google Workspace account with email employee@company.com, you revoke access to all the projects that user had access and disable login on Read the Docs completely for that user.
5.16.3 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 specific projects or versions of a project inside your organization. Additionally, you can use a HTTP Authorization Header. This is useful to have access from a script.

**Enabling Sharing**

- Go into your project’s *Admin* page and click on *Sharing*.
- Click on *New Share*
- Select access type (secret link, password, or HTTP header token), add an expiration date and a *Description* so you remember who you’re sharing it with.
- Check *Allow access to all versions?* if you want to grant access to all versions, or uncheck that option and select the specific versions you want grant access to.
- Click *Save*.
- 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.

Users can log out by using the *Log Out* link in the RTD flyout menu.

**Sharing Methods**

**Secret Link**

Once the person you send the link to clicks the link, they will have access to the documentation while their browser window is open.

If you want to link to a specific page, you can do this by passing the `next` query parameter in the URL. For example, https://mydocs.readthedocs-hosted.com/_/sharing/xxxxxxxxx?next=/en/latest/page.html.

**Tip:** This is useful for sharing access to an entire set of documentation for a user. You can embed these links in an internal wiki, for example, and all your employees will be able to browse the docs without a login.
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.

Tip: This is useful for when you have documentation you want users to bookmark. They can enter a URL directly and enter the password when prompted.

HTTP Authorization Header

Tip: This approach is useful for automated scripts. It only allows access to a page when the header is present, so it doesn’t allow browsing docs inside of a browser.

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:'

5.16.4 Project Privacy Level

Note: This feature only exists on Read the Docs for Business.

By default, only users that belong to your organization can see the dashboard of your project and its builds. If you want users outside your organization and anonymous users to be able to see the dashboard of your project, and the build output of public versions you can set the privacy level of your project to Public.

- Go the Admin tab of your project.
- Click on Advanced Settings.
- Change to Privacy level to Public.

Note: To control access to the documentation itself, see Privacy levels.
5.17 Legal Documents and Policies

Here is some of the fine print used by Read the Docs Community and Read the Docs for Business:

5.17.1 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
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 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.
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.

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.

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.
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.

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.

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.

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.

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.
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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

5.17.2 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.

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!
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.

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.

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.

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 MailerLite. 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 MailerLite. 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.
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.
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.

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.

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.

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.
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.

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.

Contacting Read the Docs

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

5.17.3 Security Policy

Read the Docs adheres to the following security policies and procedures with regards to development, operations, and managing infrastructure. You can also find information on how we handle specific user data in our Privacy Policy.

Our engineering team monitors several sources for security threats and responds accordingly to security threats and notifications.

- We monitor 3rd party software included in our application and in our infrastructure for security notifications. Any relevant security patches are applied and released immediately.
- We monitor our infrastructure providers for signs of attacks or abuse and will respond accordingly to threats.
Infrastructure

Read the Docs infrastructure is hosted on Amazon Web Services (AWS). We also use Cloudflare services to mitigate attacks and abuse.

See also:
- AWS security policies
- Cloudflare privacy and security policies

Data and data center

All user data is stored in the USA in multi-tenant datastores in Amazon Web Services data centers. Physical access to these data centers is secured with a variety of controls to prevent unauthorized access.

Application

Encryption in transit All documentation, application dashboard, and API access is transmitted using SSL encryption. We do not support unencrypted requests, even for public project documentation hosting.

Temporary repository storage We do not store or cache user repository data, temporary storage is used for every project build on Read the Docs.

Authentication Read the Docs supports SSO with GitHub, GitLab, Bitbucket, and Google Workspaces (formerly G Suite).

Payment security We do not store or process any payment details. All payment information is stored with our payment provider, Stripe – a PCI-certified level 1 payment provider.

Engineering and Operational Practices

Immutable infrastructure We don’t make live changes to production code or infrastructure. All changes to our application and our infrastructure go through the same code review process before being applied and released.

Continuous integration We are constantly testing changes to our application code and operational changes to our infrastructure.

Incident response Our engineering team is on a rotating on-call schedule to respond to security or availability incidents.

5.17.4 Data Processing Agreement

Note: This agreement can be included with any subscription on Read the Docs for Business. Contact us at privacy@readthedocs.com to include this in your subscription agreement.

This Data Processing Agreement (“DPA”) is an addendum to the Master Services Agreement (“Agreement”) between Read the Docs, Inc., along with our affiliates and subsidiaries (collectively, “Read the Docs,” “us,” or “we”) and the organization subscribing to our Services (“Organization”). This DPA takes effect on the date Organization signs up for Services, and governs the collection, processing, or receipt of Personal Data by Read the Docs on behalf of the Organization in the course of providing the Services. Terms not defined herein shall have the meaning as set forth in the Agreement. If you have questions or would like to receive a signed copy of this DPA, please contact us at privacy@readthedocs.com.
1. Definitions

a. “Applicable Laws” means all laws, rules, regulations, and orders applicable to the subject matter herein, including without limitation Data Protection Laws.

b. “California Personal Information” means Personal Data that is subject to the protection of the CCPA.

c. “CCPA” means California Civil Code Sec. 1798.100 et seq. (also known as the California Consumer Privacy Act of 2018).

d. “Consumer”, “Business”, “Sell”, and “Service Provider” shall have the meanings given to them in the CCPA.

e. “Controller”, “Data Subject”, “Processing”, and “Processor” shall have the meanings given to them in the General Data Protection Regulation (Regulation (EU) 2016/679 of the European Parliament and of the Council together with any subordinate legislation or regulation implementing the General Data Protection Regulation) or “GDPR.”

f. “Controller-to-Processor SCCs” means the Standard Contractual Clauses (Processors) in the Annex to the European Commission Decision of February 5, 2010, as may be amended or replaced from time to time by the European Commission.

g. “Organization Data” means all Personal Data, including without limitation California Personal Information and European Personal Data, Processed by Read the Docs on behalf of Organization pursuant to the Agreement.

h. “Data Protection Laws” means all applicable worldwide legislation relating to data protection and privacy that apply to the respective Party in its role of Processing Personal Data in question under the Agreement, including without limitation European Data Protection Laws and the CCPA; in each case as amended, superseded, or replaced from time to time.

i. “Data Subject” means the Consumer or other individual to whom Personal Data relates.

j. “European Data” means Personal Data that is subject to the protection of European Data Protection Laws.

k. “European Data Protection Laws” means data protection laws applicable in Europe, including: (i) Regulation 2016/679 of the European Parliament and of the Council on the protection of natural persons with regard to the processing of personal data and on the free movement of such data (GDPR); (ii) Directive 2002/58/EC concerning the processing of personal data and the protection of privacy in the electronic communications sector; and (iii) applicable national implementations of (i) and (ii); or (iii) in respect of the United Kingdom, any applicable national legislation that replaces or converts in domestic law the GDPR or any other law relating to data and privacy as a consequence of the United Kingdom leaving the European Union; and (iv) Swiss Federal Data Protection Act on 19 June 1992 and its Ordinance; in each case, as may be amended, superseded or replaced.

l. “Instructions” means the written, documented instructions issued by Organization to Read the Docs, and directing Read the Docs to perform a specific or general action regarding Personal Data for the purpose of providing the Services to Organization. The Parties agree that the Agreement (including this DPA), together with Organization’s use of the Services in accordance with the Agreement, constitute Organization’s complete and final Instructions to Read the Docs in relation to the Processing of Organization Data, and additional Instructions outside the scope of the Instructions shall require prior written agreement between Read the Docs and Organization.

m. “Personal Data” means any information relating to an identified or identifiable individual where such information is contained within Organization Data and is protected similarly as personal data, personal information, or personally identifiable information under applicable Data Protection Laws.

n. “Personal Data Breach” means a breach of security leading to the accidental or unlawful destruction, loss, alteration, unauthorized disclosure of, or access to, Personal Data transmitted, stored, or otherwise Processed by Read the Docs and/or its Sub-Processors in connection with the provision of the Services. Personal Data Breach does not include unsuccessful attempts or activities that do not compromise the security of Personal Data, including unsuccessful log-in attempts, pings, port scans, denial of service attacks, and other network attacks on firewalls or networked systems.
o. “Sub-Processor” means any entity that provides processing services to Read the Docs in furtherance of Read the Docs’s processing of Organization Data.


The nature, purpose, and subject matter of Read the Docs’s data processing activities performed as part of the Services are set out in the Agreement. The Organization Data that may be processed may relate to Data Subjects, such as the Organization’s users, employees, and individual users of Read the Docs’s website or other Services (each a “User”). Categories of Personal Data Processed may include identifiers, internet activity, education or employment-related information, commercial information, and any other Personal Data that may be processed pursuant to the Agreement.

3. Duration

The term of this DPA shall follow the term of the Agreement. Read the Docs will Process Personal Data for the duration of the Agreement, unless otherwise agreed in writing.

4. Processing of Organization Data

Read the Docs shall process Organization Data only for the purposes described in the Agreement (including this DPA) or as otherwise agreed within the scope of Organization’s lawful Instructions, except where and to the extent otherwise required by Applicable Law. If Read the Docs is collecting Personal Data from Users on behalf of Organization, Read the Docs shall follow Organization’s Instructions regarding such Personal Data collection. Read the Docs shall inform Organization without delay if, in Read the Docs’s opinion, an Instruction violates applicable Data Protection Laws and, where necessary, cease all Processing until Organization issues new Instructions with which Read the Docs is able to comply. If this provision is invoked, Read the Docs will not be liable to Organization under the Agreement for any failure to perform the Services until such time as Organization issues new lawful Instructions.

5. Confidentiality

Read the Docs shall ensure that any personnel who Read the Docs authorizes to Process Organization Data on its behalf is subject to appropriate confidentiality obligations (whether a contractual or statutory duty) with respect to that Organization Data. Additionally, Read the Docs shall take reasonable steps to ensure that (i) persons employed by Read the Docs and (ii) other persons engaged to perform on Read the Docs’s behalf comply with the terms of the Agreement.

6. Organization Responsibilities

Within the scope of the Agreement (including this DPA) and in Organization’s use of the Services, Organization shall take sole responsibility for: (i) the accuracy, quality, and legality of Organization Data and the means by which Organization acquired Personal Data; (ii) complying with all necessary transparency and lawfulness requirements under applicable Data Protection Laws for the collection and use of the Personal Data, including obtaining any necessary consents and authorizations; (iii) ensuring Organization has the right to transfer, or provide access to, the Personal Data to Read the Docs for Processing in accordance with the terms of the Agreement (including this DPA); (iv) ensuring that Organization’s Instructions to Read the Docs regarding the Processing of Organization Data comply with Applicable Laws; and (v) complying with all Applicable Laws (including Data Protection Laws) applicable to Organization’s use of the Services, including without limitation Applicable Laws relating to Organization’s Processing of Personal Data, providing notice and obtaining consents, and the Instructions it issues to Read the Docs. Organization shall inform Read the Docs without undue delay if it is not able to comply with this section or applicable Data Protection Laws. For the avoidance of doubt, Read the Docs is not responsible for compliance with any Data Protection Laws applicable to Organization or Organization’s industry that are not generally applicable to Read the Docs.
7. Sub-Processors

Organization agrees that Read the Docs may engage Sub-Processors to Process Organization Data. Where Read the Docs engages Sub-Processors, Read the Docs will impose data protection terms on the Sub-Processors that provide at least the same level of protection for Personal Data as those in this DPA, to the extent applicable to the nature of the services provided by such Sub-Processors. Read the Docs will remain responsible for each Sub-Processor’s compliance with the obligations of this DPA and for any acts or omissions of such Sub-Processor that cause Read the Docs to breach any of its obligations under this DPA. Read the Docs will maintain a current list of the Sub-processors engaged to Process Organization Data (“Sub-Processor List”), which Read the Docs shall make available to Organization upon written request.

Sub-Processor List

Effective April 16, 2021
Last updated April 16, 2021

Read the Docs for Business uses services from the following sub-processors to provide documentation hosting services. This document supplements our Data Processing Agreement and may be separately updated on a periodic basis. A sub-processor is a third party data processor who has or potentially will have access to or will process personal data.

See also:
Previous versions of this document, as well as the change history to this document, are available on GitHub

Infrastructure

Amazon Web Services, Inc. Cloud infrastructure provider.

Services

Elasticsearch B.V. Hosted ElasticSearch services for documentation search. Search indexes do not include user data.

Sendgrid, Inc. Provides email delivery to dashboard and admin users for site notifications and other generated messages. The body of notification emails can include user information, including email address.

Google Analytics Website analytics for dashboard and documentation sites.

Stripe Inc. Subscription payment provider. Data collected can include user data necessary to process payment transactions, however this data is not processed directly by Read the Docs.

Monitoring

New Relic Application performance analytics. Data collected can include user data and visitor data used within application code.

Sentry Error analytics service used to log and track application errors. Error reports can include arguments passed to application code, which can include user and visitor data.
Support

Intercom, Inc. Dashboard user reporting and analytics. Data collected can include user information like email address and IP address. This is only used on our application dashboard and is not used on documentation sites.

FrontApp, Inc. Customer email support service. Can have access to user data, including user email and IP address, and stores communications related to user data.

See also:
Read the Docs Sub-Processor List for an up-to-date list of the sub-processors we use for hosting services.

8. Security

Taking into account the state of the art, the costs of implementation and the nature, scope, context and purposes of Processing as well as the risk of varying likelihood and severity for the rights and freedoms of natural persons, Read the Docs shall, in relation to the Organization Data, maintain appropriate technical and organizational security measures designed to protect against unauthorized or accidental access, loss, alteration, disclosure or destruction of Organization Data. In assessing the appropriate level of security, Read the Docs shall take specifically into account the risks that are presented by Processing, in particular from a Personal Data Breach. Upon request, Read the Docs shall provide Organization with a summary of Read the Docs’s security policies applicable to the Services.

9. Data Transfers

Organization acknowledges and agrees that Read the Docs may access and Process Personal Data on a global basis as necessary to provide the Services in accordance with the Agreement, and in particular that Personal Data will be transferred to and Processed by Read the Docs in the United States and to other jurisdictions where Read the Docs’s Sub-Processors have operations.

10. Personal Data Breaches

If Read the Docs becomes aware of any Personal Data Breach involving Organization Data, Read the Docs will promptly, and in no case more than five calendar days after becoming aware, notify Organization in writing of the Personal Data Breach. Following such notification, to the extent required by applicable Data Protection Laws, Read the Docs will: (a) provide Organization with timely information relating to such Personal Data Breach as it becomes known or is reasonably requested by Organization; and (b) upon Organization’s request, provide Organization with commercially reasonable assistance as necessary to enable Organization to notify authorities and/or affected Data Subjects. Each Party shall be solely responsible for all costs, damages, and liabilities incurred as the result of a Personal Data Breach of the Party’s own information system and shall, at the other Party’s request and cost, provide the other Party with reasonable assistance to investigate, respond to, and mitigate the effects of a Breach of the other Party’s information system.

11. Data Subject Requests

As part of the Services, Read the Docs provides Organization and with certain controls by which the Organization may access, correct, delete, or restrict Organization Data, which Organization may use to assist it in connection with its obligations under Data Protection Laws, including its obligations relating to responding to requests from Data Subjects to exercise their rights under applicable Data Protection Laws (“Data Subject Requests”). To the extent that Organization is unable to independently address a Data Subject Request through the Services, then upon Organization’s written request Read the Docs shall provide reasonable assistance to Organization to respond to any Data Subject Requests or requests from data protection authorities relating to the Processing of Organization Data under the Agreement. Organization shall reimburse Read the Docs for the commercially reasonable costs arising from this assistance. If a Data
Subject Request or other communication regarding the Processing of Organization Data under the Agreement is made directly to Read the Docs, Read the Docs will promptly inform Organization and will advise the Data Subject to submit their request to Organization. Organization shall be solely responsible for responding substantively to any such Data Subject Requests or communications involving Personal Data.

12. Data Protection Impact Assessment and Prior Consultation

To the extent Read the Docs is required under Data Protection Law, Read the Docs shall (at Organization’s expense) provide reasonably requested information regarding Read the Docs’s processing of Organization Data under the Agreement to enable Organization to carry out data protection impact assessments or prior consultations with data protection authorities as required by law.

13. Deletion or Return of Personal Data

Upon termination or expiration of the Agreement, Read the Docs will delete or return all Organization Data Processed pursuant to this DPA in accordance with Organization’s reasonable Instructions. The requirements of this section shall not apply to the extent that Read the Docs is required by Applicable Law to retain some or all of the Organization Data, or to Organization Data Read the Docs has archived on back-up systems, which data Read the Docs shall securely isolate and protect from any further Processing and delete in accordance with Read the Docs’s deletion practices.

14. Demonstration of Compliance

Upon Organization’s written request, Read the Docs shall make available to Organization (on a confidential basis) all information reasonably necessary, and allow for and contribute to audits, to demonstrate Read the Docs’s compliance with this DPA, provided Organization shall not exercise this right more than once per year. Organization shall take all reasonable measures to limit any impact on Read the Docs by combining several information and/or audit requests carried out on behalf of Organization in one single audit.

15. European Data

This Section 15 applies only with respect to Processing of European Data by Read the Docs.

   a. Roles of the Parties. When Processing European Data under the Agreement, the Parties acknowledge and agree that Organization is the Controller and Read the Docs is the Processor.

   b. Sub-Processors. In addition to the provisions of Section 7, Read the Docs will notify Organization of any changes to Sub-processors engaged to Process European Data by updating the Sub-Processor List and posting the changes for Organization’s review. Organization may object to the engagement of a new Sub-Processor on reasonable grounds relating to the protection of Personal Data within 30 days after posting the updated Sub-Processor List. If Organization so objects, the Parties will discuss Organization’s concerns in good faith with a view to achieving a commercially reasonable resolution. If no such resolution can be reached, Read the Docs will, at its sole discretion, either not appoint the new Sub-Processor, or permit Organization to suspend or terminate the Agreement without liability to either party (but without prejudice to any fees incurred by Organization prior to suspension or termination).

   c. Data Transfers. In addition to Section 9, for transfers of European Personal Data to Read the Docs for processing by Read the Docs in a jurisdiction other than a jurisdiction in the EU, the EEA, or the European Commission-approved countries providing “adequate” data protection, Read the Docs agrees it will (i) use the form of the Controller-to-Processor SCCs or (ii) provide at least the same level of privacy protection for European Personal Data as required under the U.S.-EU and U.S.-Swiss Privacy Shield frameworks, as applicable. If such data transfers rely on Controller-to-Processor SCCs to enable the lawful transfer of European Personal Data, as set forth in the preceding sentence, the Parties agree that Data Subjects for whom Read the Docs Processes European Personal Data are third-party beneficiaries under the Controller-to-Processor SCCs. If Read the Docs is unable
or becomes unable to comply with these requirements, then (a) Read the Docs shall notify Organization of such inability and (b) any movement of European Personal Data to a non-EU country requires the prior written consent of Organization.

d. **Data Protection Impact Assessments and Consultation with Supervisory Authorities.** To the extent that the required information is reasonably available to Read the Docs, and Organization does not otherwise have access to the required information, Read the Docs will provide reasonable assistance to Organization with any data protection impact assessments, and prior consultations with supervisory authorities or other competent data privacy authorities to the extent required by European Data Protection Laws.

### 16. California Personal Information

This Section 16 applies only with respect to Processing of California Personal Information by Read the Docs in Read the Docs’s capacity as a Service Provider.

a. **Roles of the Parties.** When Processing California Personal Information in accordance with Organization’s Instructions, the Parties acknowledge and agree that Organization is a Business and Read the Docs is the Service Provider for the purposes of the CCPA. Additionally, for the purposes of interpreting this DPA with respect to Processing of California Personal Information, the term “Controller” is replaced with “Business” and “Processor” is replaced with “Service Provider” wherever those terms appear in Sections 2 through 14 and Section 17 of this DPA.

b. **Responsibilities.** The Parties agree that Read the Docs will process Users’ California Personal Information as a Service Provider strictly for the business purpose of performing the Services under the Agreement and as set forth in Read the Docs’s Privacy Policy. The Parties agree that Read the Docs shall not (i) “sell” or “share” Users’ California Personal Information (as those terms are defined in the CCPA); (ii) retain, use, or disclose Users’ California Personal Information for a commercial purpose other than for such business purpose or as otherwise permitted by the CCPA; or (iii) retain, use, or disclose Users’ California Personal Information outside of the direct business relationship between Organization and Read the Docs.

c. **Certification.** Read the Docs hereby certifies that it understands and will comply with the restrictions of Section 16(b).

d. **No CCPA Sale.** The Parties agree that Organization does not sell California Personal Information to Read the Docs because, as a Service Provider, Read the Docs may only use California Personal Information for the purposes of providing the Services to Organization.

### 17. General

Organization represents that it is authorized to, and hereby agrees to, enter into and be bound by this DPA for and on behalf of itself and each of its affiliates and subsidiaries, thereby establishing a separate DPA between Read the Docs and Organization and each of Organization’s affiliates and subsidiaries subject to the Agreement, as applicable. The relationship between Parties is that of independent contractors, and nothing herein shall be interpreted to constitute the Parties as partners, joint venturers, principal-agent, or otherwise participants in a common undertaking, or, except as expressly provided herein, allow either Party to create or assume any obligation on behalf of the other for any purpose whatsoever. The limitations of liability set forth in the Agreement shall apply to Read the Docs’s liability arising out of or relating to this DPA and the Standard Contractual Clauses (where applicable), taken in the aggregate along with the Agreement and any other agreement between the Parties. In case of any conflict or inconsistency with the terms of the Agreement, this DPA shall take precedence over the terms of the Agreement to the extent of such conflict or inconsistency. If any individual provisions of this DPA are determined to be invalid or unenforceable, the validity and enforceability of the other provisions of this DPA shall not be affected. We periodically update this Agreement. If you are a current Organization, you will be informed of any modification by email, alert on the Organization dashboard or portal or by other means.
Read the Docs Terms of Service  The terms of service for using Read the Docs Community and Read the Docs for Business. You may instead have a master services agreement for your subscription if you have a custom or enterprise contract.

Privacy Policy  Our policy on collecting, storing, and protecting user and visitor data.

Security Policy  Our policies around application and infrastructure security.

Data Processing Agreement  For GDPR and CCPA compliance, we provide a data privacy agreement for Read the Docs for Business customers.
<table>
<thead>
<tr>
<th>Route Path</th>
<th>Method</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>/api/v2/build/</td>
<td>GET</td>
<td>189</td>
</tr>
<tr>
<td>/api/v2/build/(int:id)/</td>
<td>GET</td>
<td>189</td>
</tr>
<tr>
<td>/api/v2/embed/</td>
<td>GET</td>
<td>191</td>
</tr>
<tr>
<td>/api/v2/project/</td>
<td>GET</td>
<td>185</td>
</tr>
<tr>
<td>/api/v2/project/(int:id)/</td>
<td>GET</td>
<td>186</td>
</tr>
<tr>
<td>/api/v2/project/(int:id)/active_versions/</td>
<td>GET</td>
<td>187</td>
</tr>
<tr>
<td>/api/v2/search/</td>
<td>GET</td>
<td>61</td>
</tr>
<tr>
<td>/api/v2/version/</td>
<td>GET</td>
<td>187</td>
</tr>
<tr>
<td>/api/v2/version/(int:id)/</td>
<td>GET</td>
<td>188</td>
</tr>
<tr>
<td>/api/v3/embed/</td>
<td>GET</td>
<td>238</td>
</tr>
<tr>
<td>/api/v3/embed/metadata/</td>
<td>GET</td>
<td>238</td>
</tr>
<tr>
<td>/api/v3/organizations/(string:organization_slug)/</td>
<td>GET</td>
<td>177</td>
</tr>
<tr>
<td>/api/v3/organizations/(string:organization_slug)/projects/</td>
<td>GET</td>
<td>178</td>
</tr>
<tr>
<td>/api/v3/organizations/(string:organization_slug)/projects/</td>
<td>GET</td>
<td>179</td>
</tr>
<tr>
<td>/api/v3/projects/</td>
<td>GET</td>
<td>153</td>
</tr>
<tr>
<td>/api/v3/projects/(str:project_slug)/builds/</td>
<td>GET</td>
<td>164</td>
</tr>
<tr>
<td>/api/v3/projects/(str:project_slug)/builds/(int:build_id)/</td>
<td>GET</td>
<td>162</td>
</tr>
<tr>
<td>/api/v3/projects/(str:project_slug)/environmentvariables/</td>
<td>GET</td>
<td>175</td>
</tr>
<tr>
<td>/api/v3/projects/(str:project_slug)/environmentvariables/(int:environmentvariable_id)/</td>
<td>GET</td>
<td>174</td>
</tr>
<tr>
<td>/api/v3/projects/(str:project_slug)/redirects/</td>
<td>GET</td>
<td>171</td>
</tr>
<tr>
<td>/api/v3/projects/(str:project_slug)/redirects/(int:redirect_id)/</td>
<td>GET</td>
<td>170</td>
</tr>
<tr>
<td>/api/v3/projects/(str:project_slug)/subprojects/</td>
<td>GET</td>
<td>166</td>
</tr>
<tr>
<td>/api/v3/projects/(str:project_slug)/subprojects/(str:alias_slug)/</td>
<td>GET</td>
<td>166</td>
</tr>
<tr>
<td>/api/v3/projects/(str:project_slug)/translations/</td>
<td>GET</td>
<td>169</td>
</tr>
<tr>
<td>/api/v3/projects/(string:project_slug)/</td>
<td>GET</td>
<td>155</td>
</tr>
<tr>
<td>/api/v3/projects/(string:project_slug)/versions/</td>
<td>GET</td>
<td>159</td>
</tr>
<tr>
<td>/api/v3/projects/(string:project_slug)/versions/(string:version_slug)/</td>
<td>GET</td>
<td>160</td>
</tr>
</tbody>
</table>

HTTP ROUTING TABLE
D
dashboard, 26

E
environment variable
   READTHEDOCS, 83

F
flyout menu, 26

P
profile page, 26
project home, 26
project page, 26
Python Enhancement Proposals
   PEP 440, 149
   PEP 517, 136

R
READTHEDOCS, 83
root URL, 26