Configuration File V2

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

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

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

# Required
version: 2

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

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

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

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

Supported settings

Note

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

version

Required: `true`

Example:

```yaml
version: 2
```

Warning

If you don't provide the version, v1 will be used.

formats

Formats of the documentation to be built.

Type: `list`

Options: `htmlzip`, `pdf`, `epub`

Default: `[]`

Example:

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

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

Note

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

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

Note

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

python

Configuration of the Python environment to be used. Example:

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

python.version

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

Type: `number`

Default: `3`

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

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:

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

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

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

python.system_packages

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

- **Type:** `bool`
- **Default:** `false`

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

conda

Configuration for Conda support. Example:

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

conda.environment

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

- **Type:** `path`
- **Required:** `true`

build

Configuration for the documentation build process. Example:

```
build:
  image: latest

python:
  version: 3.7
```

build.image

The Docker image used for building the docs.

- **Type:** `string`
- **Options:** `stable`, `latest`
- **Default:** `latest`

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

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

sphinx

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

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

sphinx.builder

The builder type for the Sphinx documentation.

Type: `string`

Options: `html`, `htmldir`, `singlehtml`

Default: `html`

sphinx.configuration

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

Type: `path`

Default: `null`

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

sphinx.fail_on_warning

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

Type: `bool`

Default: `false`

mkdocs

Configuration for Mkdocs documentation. Example:

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

mkdocs.configuration

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

Type: `path`

Default: `null`

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

mkdocs.fail_on_warning

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

Type: `bool`

Default: `false`

submodules

VCS submodules configuration.

Note

Only Git is supported at the moment.

Note

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

Example:

```yaml
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
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: []`.

```yaml
submodules:
  exclude: all
```

submodules.recursive

Do a recursive clone of the submodules.

Type: `bool`

Default: `false`

Note

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

Schema

You can see the complete schema here.

Migrating from v1

Changes

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

Warning

Some values from the web interface are no longer respected, please see Migrating from the web interface if you have settings there.

New settings

- sphinx
- mkdocs
- submodules
- python.install

Migrating from the web interface

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

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

- `Name`
- `Repository URL`
- `Repository type`
- `Language`
- `Programming language`
- `Project homepage`
- `Tags`
- `Single version`

- `Default branch`
- `Default version`
- `Show versions warning`
- `Privacy level`
- `Analytics code`