
sphinx-highlights

Release 0.6.0

**Sphinx extension to display a selection of highlights
from a Python library.**

Dominic Davis-Foster

Mar 17, 2024

Contents

1	Installation	1
1.1	from PyPI	1
1.2	from Anaconda	1
1.3	from GitHub	1
2	Usage	3
2.1	.. api-highlights::	3
2.2	Customising the colours	4
3	Demo	5
3.1	Highlights	5
4	API Reference	7
4.1	SphinxHighlightsDirective	7
4.2	copy_assets	8
4.3	format_parameter	8
4.4	format_signature	8
4.5	get_random_sample	8
4.6	setup	8
5	Downloading source code	9
5.1	Building from source	10
	Python Module Index	11
	Index	13

Installation

1.1 from PyPI

```
$ python3 -m pip install sphinx-highlights --user
```

1.2 from Anaconda

First add the required channels

```
$ conda config --add channels https://conda.anaconda.org/conda-forge
$ conda config --add channels https://conda.anaconda.org/domdfcoding
```

Then install

```
$ conda install sphinx-highlights
```

1.3 from GitHub

```
$ python3 -m pip install git+https://github.com/sphinx-toolbox/sphinx-highlights@master --user
```

Enable `seed_intersphinx_mapping` by adding the following to the `extensions` variable in your `conf.py`:

```
extensions = [
    ...
    'seed_intersphinx_mapping',
]
```

For more information see

<https://www.sphinx-doc.org/en/master/usage/extensions#third-party-extensions>.

Usage

sphinx-highlights provides a single directive:

.. api-highlights::

Shows 4 random highlights of the library.

The objects to include in the highlights are given in the body of the directive. For example:

```
.. api-highlights::

    domdf_python_tools.stringlist.StringList
    domdf_python_tools.testing.check_file_regression
    domdf_python_tools.paths.PathPlus
    domdf_python_tools.iterative.groupfloats
```

More than four objects can be listed. A random selection of those will be chosen when the documentation is built.

:module: (string)

The parent module of all of these objects.

Allows the module name to be replaced with a dot (.). For example:

```
.. api-highlights::
    :module: domdf_python_tools

    .stringlist.StringList
```

:colours: (Comma- or space-separated list of strings.)

The colours to use for the panel headers. Choose from “blue”, “green”, “red”, or “orange”.
Default “blue”.

Changed in version 0.2.0: If more than four colours are provided four will be chosen at random.

:classes: (Comma- or space-separated list of strings.)

The classes to use for the panels.

Default `col-xl-6 col-lg-6 col-md-12 col-sm-12 col-xs-12 p-2`.

2.2 Customising the colours

By default the only colours available are:

- blue
- green
- red
- orange

Additional colours can be created by adding your own custom CSS to Sphinx:

```
div.sphinx-highlights div.highlight-purple div.card-header {  
    background-color: #B452CD;  
}
```

where `purple` is the name of the colour to use in the `colours` option.

- `purple`

See also:

<https://docs.readthedocs.io/en/stable/guides/adding-custom-css.html> for more information on adding custom CSS.

3.1 Highlights

- `iterative.groupfloats()`

```
groupfloats(  
    iterable: Iterable[float],  
    step: float = 1,  
    ) -> Iterable[Tuple[float, ...]]
```

Returns an iterator over the discrete ranges of values in `iterable`.

- `stringlist.StringList`

```
StringList(  
    iterable: Iterable[String] = (),  
    convert_indents: bool = False,  
    )
```

A list of strings that represent lines in a multiline string.

- `words.Plural`

```
Plural(singular: str, plural: str)
```

Represents a word as its singular and plural.

- `paths.PathPlus`

```
PathPlus(*args, **kwargs)
```

Subclass of `pathlib.Path` with additional methods and a default encoding of UTF-8.

See the online documentation at <https://sphinx-highlights.readthedocs.io> for an example with the HTML builder.

API Reference

Sphinx extension to display a selection of highlights from a Python library.

Classes:

<code>SphinxHighlightsDirective(name, arguments, ...)</code>	Provides the <i>api-highlights</i> directive.
--	---

Functions:

<code>copy_assets(app[, exception])</code>	Copy asset files to the output.
<code>format_parameter(param)</code>	Format an <code>inspect.Parameter</code> , for insertion into the highlight panel.
<code>format_signature(obj)</code>	Format the signature of the given object, for insertion into the highlight panel.
<code>get_random_sample(items)</code>	Returns four random elements from <code>items</code> .
<code>setup(app)</code>	Setup <i>sphinx_highlights</i> .

class `SphinxHighlightsDirective` (*name, arguments, options, content, lineno, content_offset, block_text, state, state_machine*)

Bases: `SphinxDirective`

Provides the *api-highlights* directive.

Methods:

<code>delimited_get(option, default)</code>	Returns the value of the option with the given name, splitting the input at commas, semicolons and spaces.
<code>run()</code>	Create the highlights node.
<code>run_generic()</code>	Generate generic reStructuredText output.
<code>run_html()</code>	Generate output for HTML builders.

delimited_get (*option, default*)

Returns the value of the option with the given name, splitting the input at commas, semicolons and spaces.

Parameters

- **option** (*str*) – The option name.
- **default** (*str*) – The default value, as a string separated by commas, spaces or semicolons.

Return type `Iterator[str]`

run()
Create the highlights node.
Return type `List[Node]`

run_generic()
Generate generic reStructuredText output.
Return type `List[Node]`

run_html()
Generate output for HTML builders.
Return type `List[Node]`

copy_assets(*app*, *exception=None*)
Copy asset files to the output.

Parameters

- **app** (`Sphinx`) – The Sphinx application.
- **exception** (`Optional[Exception]`) – Any exception which occurred and caused Sphinx to abort. Default `None`.

format_parameter(*param*)
Format an `inspect.Parameter`, for insertion into the highlight panel.

Parameters **param** (`Parameter`)

Return type `str`

Returns The reStructuredText string.

format_signature(*obj*)
Format the signature of the given object, for insertion into the highlight panel.

Parameters **obj** (`Union[type, FunctionType]`)

Return type `StringList`

Returns A list of reStructuredText lines.

get_random_sample(*items*)
Returns four random elements from *items*.

Parameters **items** (`Iterable[~T]`)

Return type `List[~T]`

setup(*app*)
Setup `sphinx_highlights`.

Parameters **app** (`Sphinx`) – The Sphinx application.

Return type `SphinxExtMetadata`

Downloading source code

The sphinx-highlights source code is available on GitHub, and can be accessed from the following URL:
<https://github.com/sphinx-toolbox/sphinx-highlights>

If you have git installed, you can clone the repository with the following command:

```
$ git clone https://github.com/sphinx-toolbox/sphinx-highlights
```

```
Cloning into 'sphinx-highlights'...
remote: Enumerating objects: 47, done.
remote: Counting objects: 100% (47/47), done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 173 (delta 16), reused 17 (delta 6), pack-reused 126
Receiving objects: 100% (173/173), 126.56 KiB | 678.00 KiB/s, done.
Resolving deltas: 100% (66/66), done.
```

Alternatively, the code can be downloaded in a ‘zip’ file by clicking:

Clone or download → Download Zip

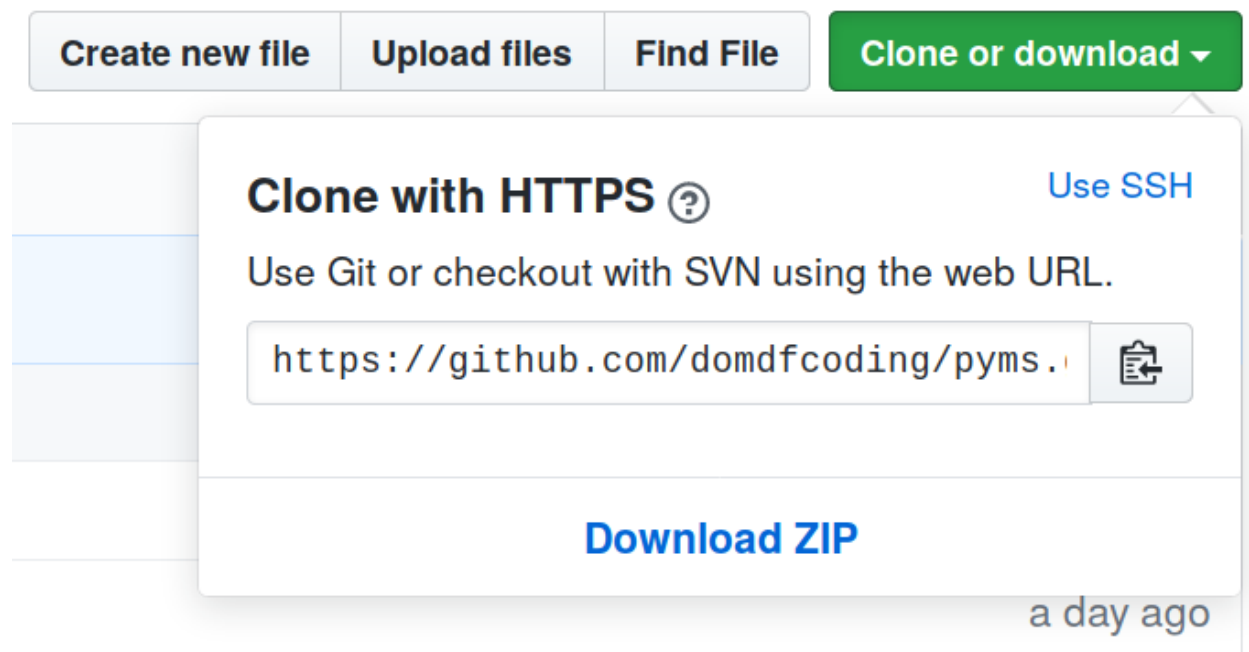


Fig. 1: Downloading a ‘zip’ file of the source code

5.1 Building from source

The recommended way to build `sphinx-highlights` is to use `tox`:

```
$ tox -e build
```

The source and wheel distributions will be in the directory `dist`.

If you wish, you may also use `pep517.build` or another **PEP 517**-compatible build tool.

Python Module Index

S

sphinx_highlights, [7](#)

Symbols

`:classes:` (*directive option*)
 `api-highlights` (*directive*), 3
`:colours:` (*directive option*)
 `api-highlights` (*directive*), 3
`:module:` (*directive option*)
 `api-highlights` (*directive*), 3

A

`api-highlights` (*directive*), 3
 `:classes:` (*directive option*), 3
 `:colours:` (*directive option*), 3
 `:module:` (*directive option*), 3

C

`copy_assets()` (*in module sphinx_highlights*), 8

D

`delimited_get()` (*SphinxHighlightsDirective method*), 7

F

`format_parameter()` (*in module sphinx_highlights*), 8
`format_signature()` (*in module sphinx_highlights*), 8

G

`get_random_sample()` (*in module sphinx_highlights*), 8

M

`module`
 `sphinx_highlights`, 7

P

Python Enhancement Proposals
 PEP 517, 10

R

`run()` (*SphinxHighlightsDirective method*), 7
`run_generic()` (*SphinxHighlightsDirective method*), 8

`run_html()` (*SphinxHighlightsDirective method*), 8

S

`setup()` (*in module sphinx_highlights*), 8
`sphinx_highlights`
 `module`, 7
`SphinxHighlightsDirective` (*class in sphinx_highlights*), 7