sphinxcontrib-bibtex Documentation

Release 2.1.0

Matthias C. M. Troffaes

Dec 28, 2020

CONTENTS

Release 2.1.0

Date Dec 28, 2020

CHAPTER

ONE

CONTENTS

1.1 Getting Started

1.1.1 Overview

Sphinx extensions for BibTeX style citations.

The bibtex extension allows BibTeX citations to be inserted into documentation generated by Sphinx, via a `bibliography` directive, and a `cite` role, which work similarly to LaTeX's `thebibliography` environment and `\cite` command.

For formatting, the extension relies on pybtex written by Andrey Golovizin. The extension is inspired by Matthew Brett's bibstuff.sphinxext.bibref and Weston Nielson's sphinx-natbib.

- Download: https://pypi.org/project/sphinxcontrib-bibtex/#files
- Documentation: https://sphinxcontrib-bibtex.readthedocs.io/en/latest/
- Development: https://github.com/mcmtroffaes/sphinxcontrib-bibtex/

1.1.2 Installation

Install the module with `pip install sphinxcontrib-bibtex`, or from source using `python setup.py install`. Then add:

```
extensions = ['sphinxcontrib.bibtex']
bibtex_bibfiles = ['refs.bib']
```

to your project's Sphinx configuration file `conf.py`.

1.1.3 Minimal Example

In your project's documentation, you can then write for instance:

```
See :cite:`1987:nelson` for an introduction to non-standard analysis.

.. bibliography::
```

where `refs.bib` would contain an entry:

```
@Book{1987:nelson,
  author = {Edward Nelson},
  title = {Radically Elementary Probability Theory},
  publisher = {Princeton University Press},
  year = {1987}
}
```

In the default style, this will get rendered as:

See [Nel87a] for an introduction to non-standard analysis.

Citations in sphinx are resolved globally across all documents. Typically, you have a single *bibliography* directive across your entire project which collects all citations. Advanced use cases with multiple *bibliography* directives across your project are also supported, but some care needs to be taken from your end to avoid duplicate citations.

In contrast, footnotes in sphinx are resolved locally per document. To achieve local bibliographies per document, you can use citations represented by footnotes as follows:

```
Non-standard analysis is lovely. :footcite:`1987:nelson`

.. footbibliography::
```

which will get rendered as:

Non-standard analysis is lovely.[1]

Typically, you have a single *footbibliography* directive at the bottom of each document that has *footcite* citations. Advanced use cases with multiple *footbibliography* directives per document are also supported. Since everything is local, there is no concern with duplicate citations when using footnotes.

1.2 Usage

1.2.1 Configuration

New in version 2.0.0.

To configure the extension, in your `conf.py` file, set `bibtex_bibfiles` to your list of bib files. For instance, a minimal configuration may look as follows:

```
extensions = ['sphinxcontrib.bibtex']
bibtex_bibfiles = ['refs.bib']
```

In bib files, LaTeX control characters are automatically converted to unicode characters (for instance, to convert `\'e` into é). Be sure to write `\%` when you intend to format a percent sign.

You can change the bibliography style, using the `bibtex_default_style` variable in your `conf.py`. If none is specified, the `alpha` style is used. Other supported styles are `plain`, `unsrt`, and `unsrtalpha`. You can also create your own style (see *Custom Formatting, Sorting, and Labelling*). For example:

```
bibtex_default_style = 'unsrt'
```

You can set the encoding of the bibliography files, using the `bibtex_encoding` variable in your `conf.py`. If no encoding is specified, `utf-8-sig` is assumed. For example:

[1] Edward Nelson. *Radically Elementary Probability Theory*. Princeton University Press, 1987.

```
bibtex_encoding = 'latin'
```

1.2.2 Roles and Directives

:cite:
Create a citation to a bibliographic entry. For example:

```
See :cite:`1987:nelson` for an introduction to non-standard analysis.
```

which would be equivalent to the following LaTeX code:

```
See \cite{1987:nelson} for an introduction to non-standard analysis.
```

Multiple comma-separated keys can be specified at once:

```
See :cite:`1987:nelson,2001:schechter`.
```

.. bibliography::
Create bibliography for all cited references. Citations in sphinx are resolved globally across all documents. Typically, you have a single bibliography directive across your entire project which collects all citations.

The `all` flag forces all references to be included (equivalent to `\nocite{*}` in LaTeX). The `notcited` flag causes all references that were not cited to be included. The `cited` flag is recognized as well but is entirely optional. For example:

```
.. bibliography::
   :cited:
```

which would be roughly equivalent to the following LaTeX code:

```
\begin{thebibliography}{1}
  \bibitem{1987:nelson}
  Edward~Nelson
  \newblock {\em Radically Elementary Probability Theory}.
  \newblock Princeton University Press, 1987.
\end{thebibliography}
```

You can also override the default bibliography style:

```
.. bibliography::
   :style: unsrt
```

Warning: Sphinx will attempt to resolve references to the bibliography across all documents, so you must take care that no citation key is included more than once.

:footcite:
New in version 2.0.0.

Create a footnote reference to a bibliographic entry. For example:

```
See :footcite:`1987:nelson` for an introduction to non-standard analysis.
```

which would be equivalent to the following LaTeX code:

```
See \footcite{1987:nelson} for an introduction to non-standard analysis.
```

As with *cite*, multiple comma-separated keys can be specified at once:

```
See :footcite:`1987:nelson,2001:schechter`.
```

`.. footbibliography::`
New in version 2.0.0.

Create footnotes at this location for all references that are cited in the current document up to this point. Typically, you have a single footbibliography directive at the bottom of each document that has footcite citations.

If specified multiple times in the same document, footnotes are only created for references that do not yet have a footnote earlier in the document.

1.2.3 Advanced Features

Splitting Bibliographies Per Bib File

New in version 2.0.0.

If want multiple bibliographies each of which only contains references from specific bib files, you can specify the relevant bib files as an optional argument to the directive.

The next example shows how to split your citations between articles and books, assuming your articles are in `articles.bib` and your books are in `books1.bib` and `books2.bib`.

```
.. rubric:: Articles

.. bibliography:: articles.bib

.. rubric:: Books

.. bibliography:: books1.bib books2.bib
```

The bib files must be specified as a path that is relative to the containing document.

Bullet Lists and Enumerated Lists

New in version 0.2.4.

You can change the type of list used for rendering the bibliography. By default, a paragraph of standard citations is generated. However, instead, you can also generate a bullet list, or an enumerated list.

```
.. bibliography::
   :list: bullet
   :all:

.. bibliography::
   :list: enumerated
   :all:
```

Note that citations to these types of bibliography lists will not be resolved.

For enumerated lists, you can also specify the type (default is `arabic`), and the start of the sequence (default is `1`).

```
.. bibliography::
   :list: enumerated
   :enumtype: upperroman
   :start: 3
   :all:
```

The enumtype can be any of `arabic` (1, 2, 3, ...), `loweralpha` (a, b, c, ...), `upperalpha` (A, B, C, ...), `lowerroman` (i, ii, iii, ...), or `upperroman` (I, II, III, ...).

The start can be any positive integer (1, 2, 3, ...) or `continue` if you wish the enumeration to continue from the last *`bibliography`* directive. This is helpful if you split up your bibliography but still want to enumerate the entries continuously.

Label Prefixing

New in version 0.2.5.

If you have multiple bibliographies, and experience duplicate labels, use the `labelprefix` option.

```
.. rubric:: References

.. bibliography::
   :cited:
   :labelprefix: A

.. rubric:: Further reading

.. bibliography::
   :notcited:
   :labelprefix: B
```

Key Prefixing

New in version 0.3.3.

If you have multiple bibliographies, and you would like entries to be repeated in different documents, then use the `keyprefix` option.

For example, suppose you have two documents, and you would like to cite `boole1854` in both of these doucments, with the bibliography entries showing in both of the documents. In one document you could have:

```
See :cite:`a-boole1854`

.. bibliography::
   :labelprefix: A
   :keyprefix: a-
```

whilst in the other document you could have:

```
See :cite:`b-boole1854`

.. bibliography::
   :labelprefix: B
   :keyprefix: b-
```

The bibliographies will then both generate an entry for `boole1854`, with links and backlinks as expected.

See also:

Local Bibliographies

Filtering

New in version 0.2.7.

Whilst the `cited`, `all`, and `notcited` options will cover many use cases, sometimes more advanced selection of bibliographic entries is desired. For this purpose, you can use the `filter` option:

```
.. bibliography::
   :list: bullet
   :filter: author % "Einstein"
```

The string specified in the filter option must be a valid Python expression.

Note: The expression is parsed using `ast.parse()` and then evaluated using an `ast.NodeVisitor`, so it should be reasonably safe against malicious code.

The filter expression supports:

- The boolean operators `and`, `or`.
- The unary operator `not`.
- The comparison operators ==, <=, <, >=, and >.
- Regular expression matching using the `%` operator, where the left hand side is the string to be matched, and the right hand side is the regular expression. Matching is case insensitive. For example:

  ```
  .. bibliography::
     :list: bullet
     :filter: title % "relativity"
  ```

  would include all entries that have the word "relativity" in the title.

  Note: The implementation uses `re.search()`.

- Single and double quoted strings, such as `'hello'` or `"world"`.
- Set literals, such has `{"hello", "world"}`, as well as the set operators `&`, `|`, `in`, and `not in`.

  New in version 0.3.0.
- Various identifiers, such as:
  - `type` is the entry type, as a lower case string (i.e. `"inproceedings"`).
  - `key` is the entry key, as a lower case string (this is because keys are considered case insensitive).
  - `cited` evaluates to `True` if the entry was cited in the document, and to `False` otherwise.
  - `docname` evaluates to the name of the current document.

    New in version 0.3.0.

- `docnames` evaluates to a set of names from which the entry is cited.

  New in version 0.3.0.

- `True` and `False`.
- `author` is the entry string of authors in standard format (last, first), separated by "and".
- `editor` is similar to `author` but for editors.
- Any other (lower case) identifier evaluates to a string containing the value of the correspondingly named field, such as `title`, `publisher`, `year`, and so on. If the item is missing in the entry then it evaluates to the empty string. Here is an example of how one would typically write an expression to filter on an optional field:

```
.. bibliography::
   :list: bullet
   :filter: cited and year and (year <= "2003")
```

  which would include all cited entries that have a year that is less or equal than 2003; any entries that do not specify a year would be omitted.

Local Bibliographies

The easiest way to have a local bibliography per document is to use *`footcite`* along with *`footbibliography`*.

If you prefer to have regular citations instead of footnotes, both the `keyprefix` and `filter` options can be used to achieve local bibliographies with *`cite`* and *`bibliography`*.

The `filter` system for local bibliographies can only be used if no citation key is used in more than one document. This is not always satisfied. If you need to cite the same reference in multiple documents with references to multiple local bibliographies, use the `keyprefix` system; see *Key Prefixing*.

To create a bibliography that includes only citations that were cited in the current document, use the following filter:

```
.. bibliography::
   :filter: docname in docnames
```

More generally, you can create bibliographies for citations that were cited from specific documents only:

```
.. bibliography::
   :filter: {"doc1", "doc2"} & docnames
```

This bibliography will include all citations that were cited from `doc1.rst` or `doc2.rst`. Another hypothetical example:

```
.. bibliography::
   :filter: cited and ({"doc1", "doc2"} >= docnames)
```

This bibliography will include all citations that were cited in `doc1.rst` or `doc2.rst`, but nowhere else.

Custom Formatting, Sorting, and Labelling

pybtex provides a very powerful way to create and register new styles, using setuptools entry points, as documented here: https://docs.pybtex.org/api/plugins.html

Simply add the following code to your conf.py:

```
from pybtex.style.formatting.unsrt import Style as UnsrtStyle
from pybtex.style.template import toplevel # ... and anything else needed
from pybtex.plugin import register_plugin

class MyStyle(UnsrtStyle):

    def format_XXX(self, e):
        template = toplevel [
            # etc.
        ]
        return template.format_data(e)

register_plugin('pybtex.style.formatting', 'mystyle', MyStyle)
```

Now mystyle will be available to you as a formatting style:

```
bibtex_default_style = 'mystyle'
```

An minimal example is available here: https://github.com/mcmtroffaes/sphinxcontrib-bibtex/tree/develop/test/roots/test-custom_style

The formatting code uses a very intuitive template engine. The source code for unsrt provides many great examples: https://bitbucket.org/pybtex-devs/pybtex/src/master/pybtex/style/formatting/unsrt.py?at=master&fileviewer=file-view-default

The above example only demonstrates a custom formatting style plugin. It is also possible to register custom author/editor naming plugins (using the pybtex.style.names group) labelling plugins (using the pybtex.style.labels group), and sorting plugins (using the pybtex.style.sorting group). A few minimal examples demonstrating how to create a custom label styles are available here:

- https://github.com/mcmtroffaes/sphinxcontrib-bibtex/tree/develop/test/roots/test-issue77
- https://github.com/mcmtroffaes/sphinxcontrib-bibtex/tree/develop/test/roots/test-custom_labels

Custom Bibliography Header

By default, the *bibliography* and *footbibliography* directives simply insert a paragraph. The bibtex_bibliography_header and bibtex_footbibliography_header configuration variables can be set to add a header to this. For example, in your conf.py you could have:

```
bibtex_bibliography_header = ".. rubric:: References"
bibtex_footbibliography_header = bibtex_bibliography_header
```

This adds a rubric title to every bibliography.

1.2.4 Known Issues and Workarounds

Encoding: Percent Signs

Be sure to write `\%` for percent signs at all times (unless your file contains a genuine comment), otherwise the parser will ignore the remainder of the line.

Duplicate Labels When Using `:style: plain`

With `:style: plain`, labels are numerical, restarting at `[1]` for each *bibliography* directive. Consequently, when inserting multiple *bibliography* directives with `:style: plain`, you are bound to get duplicate labels for entries. There are a few ways to work around this problem:

- Use a single bibliography directive for all your references.
- Use the `labelprefix` option, as documented above.
- Use a style that has non-numerical labelling, such as `:style: alpha`.

LaTeX Backend Fails with Citations In Figure Captions

Sphinx generates `\phantomsection` commands for references, however LaTeX does not support these in figure captions. You can work around this problem by adding the following code to your `conf.py`:

```
latex_elements = {
 'preamble': r'''
     % make phantomsection empty inside figures
     \usepackage{etoolbox}
     \AtBeginEnvironment{figure}{\renewcommand{\phantomsection}{}}
 '''
}
```

Mismatch Between Output of HTML/Text and LaTeX Backends

Sphinx's LaTeX writer currently collects all citations together, and puts them on a separate page, with a separate title, whereas the html and text writers puts citations at the location where they are defined. This issue will occur also if you use regular citations in Sphinx: it has nothing to do with sphinxcontrib-bibtex per se.

To get a closer match between the two outputs, you can tell Sphinx to generate a rubric title only for html or text outputs:

```
.. only:: html or text

   .. rubric:: References

.. bibliography::
```

This code could be placed in a `references.rst` file that you include at the end of your toctree.

Alternatively, to remove the bibliography section title from the LaTeX output, you can add the following to your LaTeX preamble:

```
\usepackage{etoolbox}
\patchcmd{\thebibliography}{\section*{\refname}}{}{}{}
```

1.3 Extension API

1.3.1 Sphinx Interface

sphinxcontrib.bibtex.**setup**(*app: sphinx.application.Sphinx*) → Dict[str, Any]

Set up the bibtex extension:

- register config values
- register directives
- register nodes
- register roles
- register transforms
- connect events to functions

sphinxcontrib.bibtex.**init_foot_bibliography_id**(*app: sphinx.application.Sphinx*, *docname: docutils.nodes.document*, *source: str*) → None

Initialize current footbibliography id for *docname*.

1.3.2 New Roles

class sphinxcontrib.bibtex.roles.**CiteRole**(*fix_parens: bool = False*, *lowercase: bool = False*, *nodeclass: Type[Element] = None*, *innernodeclass: Type[TextElement] = None*, *warn_dangling: bool = False*)

Bases: sphinx.roles.XRefRole

Class for processing the *cite* role.

result_nodes(*document*, *env*, *node*, *is_ref*)

Associate the pending_xref with the cite domain, and note the cited citation keys.

class sphinxcontrib.bibtex.foot_roles.**FootCiteRole**(*fix_parens: bool = False*, *lowercase: bool = False*, *nodeclass: Type[Element] = None*, *innernodeclass: Type[TextElement] = None*, *warn_dangling: bool = False*)

Bases: sphinx.roles.XRefRole

Class for processing the *footcite* role.

result_nodes(*document: docutils.nodes.document*, *env: sphinx.environment.BuildEnvironment*, *node: docutils.nodes.Element*, *is_ref: bool*) → Tuple[List[docutils.nodes.Node], List[docutils.nodes.system_message]]

Transform reference node into a footnote reference, and note that the reference was cited.

1.3.3 New Nodes

class sphinxcontrib.bibtex.nodes.**bibliography**(*rawsource=''*, **children*, ***attributes*)
Node for representing a bibliography. Replaced by a list of citations by *BibliographyTransform*.

class sphinxcontrib.bibtex.foot_nodes.**footbibliography**(*rawsource=''*, **children*, ***attributes*)
Node for representing a bibliography. Replaced by a list of citations by *FootBibliographyTransform*.

1.3.4 New Directives

class sphinxcontrib.bibtex.directives.**BibliographyDirective**(*name*, *arguments*, *options*, *content*, *lineno*, *content_offset*, *block_text*, *state*, *state_machine*)
Class for processing the *bibliography* directive.

Parses the bibliography files, and produces a *bibliography* node.

See also:

Further processing of the resulting *bibliography* node is done by *BibliographyTransform*.

run()
Process .bib files, set file dependencies, and create a node that is to be transformed to the entries of the bibliography.

sphinxcontrib.bibtex.directives.**process_start_option**(*value*)
Process and validate the start option value of a *bibliography* directive. If *value* is `continue` then this function returns -1, otherwise *value* is converted into a positive integer.

class sphinxcontrib.bibtex.foot_directives.**FootBibliographyDirective**(*name*, *arguments*, *options*, *content*, *lineno*, *content_offset*, *block_text*, *state*, *state_machine*)
Class for processing the *footbibliography* directive.

Produces a *footbibliography* node.

See also:

Further processing of the resulting *footbibliography* node is done by *FootBibliographyTransform*.

run()
Set file dependencies, update footbib id, and create a node that is to be transformed to the entries of the bibliography.

1.3.5 New Transforms

class sphinxcontrib.bibtex.transforms.**BibliographyTransform**(*document*, *startnode=None*)

Bases: sphinx.transforms.post_transforms.SphinxPostTransform

A docutils transform to generate citation entries for bibliography nodes.

default_priority = 5

run(***kwargs*)

Transform each *bibliography* node into a list of citations.

sphinxcontrib.bibtex.transforms.**node_text_transform**(*node*, *transform*)

Apply transformation to all Text nodes within node.

sphinxcontrib.bibtex.transforms.**transform_url_command**(*textnode*)

Convert '\url{...}' into a proper docutils hyperlink.

class sphinxcontrib.bibtex.foot_transforms.**FootBibliographyTransform**(*document*, *startnode=None*)

Bases: sphinx.transforms.SphinxTransform

A docutils transform to generate footnotes for bibliography nodes.

default_priority = 10

Priority of the transform. See https://docutils.sourceforge.io/docs/ref/transforms.html

apply()

Transform each *footbibliography* node into a list of citations.

1.3.6 New Domains

Classes and methods to maintain any bibtex information that is stored outside the doctree.

class sphinxcontrib.bibtex.domain.**BibliographyKey**(*docname*, *id_*)

property docname

Alias for field number 0

property id_

Alias for field number 1

class sphinxcontrib.bibtex.domain.**BibliographyValue**(*line: int*, *bibfiles: List[str]*, *style: str*, *list_: str*, *enumtype: str*, *start: int*, *labelprefix: str*, *keyprefix: str*, *filter_: _ast.AST*, *citation_nodes: Dict[str, docutils.nodes.citation]*)

Contains information about a bibliography directive.

property bibfiles

List of bib files for this directive.

property citation_nodes

key -> citation node

property enumtype

The sequence type (for enumerated lists).

property filter_
Parsed filter expression.

property keyprefix
String prefix for citation keys.

property labelprefix
String prefix for pybtex generated labels.

property line
Line number of the directive in the document.

property list_
The list type.

property start
The start of the sequence (for enumerated lists).

property style
The pybtex style.

class sphinxcontrib.bibtex.domain.**Citation**(*citation_id: Optional[str]*, *bibliography_key: sphinxcontrib.bibtex.domain.BibliographyKey*, *key: str*, *label: str*, *formatted_entry: pybtex.style.FormattedEntry*)
Information about a citation.

property bibliography_key
Key of its bibliography directive.

property citation_id
Unique id of this citation.

property formatted_entry
Entry as formatted by pybtex.

property key
Unique citation id used for referencing.

property label
Label (with brackets and label prefix).

class sphinxcontrib.bibtex.domain.**CitationRef**(*citation_ref_id: str*, *docname: str*, *line: int*, *keys: List[str]*)
Information about a citation reference.

property citation_ref_id
Unique id of this citation reference.

property docname
Document name.

property keys
Citation keys (including key prefix).

property line
Line number.

class sphinxcontrib.bibtex.domain.**BibtexDomain**(*env: sphinx.environment.BuildEnvironment*)
Global bibtex extension information cache.

property bibfiles
Map each bib filename to some information about the file (including the parsed data).

property bibliographies
Map storing information about each bibliography directive.

check_consistency() → None
Do consistency checks (**experimental**).

property citation_refs
Citation reference data.

property citations
Citation data.

clear_doc(*docname: str*) → None
Remove traces of a document in the domain-specific inventories.

get_all_cited_keys(*docnames*)
Yield all citation keys for given *docnames* in order, then ordered by citation order.

get_entries(*bibfiles: List[str]*) → Iterable[pybtex.database.Entry]
Return all bibliography entries from the bib files, unsorted (i.e. in order of appearance in the bib files.

get_filtered_entries(*bibliography_key:* sphinxcontrib.bibtex.domain.BibliographyKey) → Iterable[Tuple[str, pybtex.database.Entry]]
Return unsorted bibliography entries filtered by the filter expression.

get_formatted_entries(*bibliography_key:* sphinxcontrib.bibtex.domain.BibliographyKey, *docnames: List[str]*) → Iterable[pybtex.style.FormattedEntry]
Get sorted bibliography entries along with their pybtex labels, with additional sorting and formatting applied from the pybtex style.

get_sorted_entries(*bibliography_key:* sphinxcontrib.bibtex.domain.BibliographyKey, *docnames: List[str]*) → Iterable[Tuple[str, pybtex.database.Entry]]
Return filtered bibliography entries sorted by citation order.

merge_domaindata(*docnames: List[str]*, *otherdata: Dict*) → None
Merge in data regarding *docnames* from a different domaindata inventory (coming from a subprocess in parallel builds).

resolve_xref(*env: sphinx.environment.BuildEnvironment*, *fromdocname: str*, *builder: sphinx.builders.Builder*, *typ: str*, *target: str*, *node: sphinx.addnodes.pending_xref*, *contnode: docutils.nodes.Element*) → docutils.nodes.Element
Replace node by list of citation references (one for each key).

1.3.7 Bib Files

Classes and methods to work with bib files.

class sphinxcontrib.bibtex.bibfile.**BibFile**(*mtime: float*, *data: pybtex.database.BibliographyData*)
Contains information about a parsed bib file.

property data
parsed data from pybtex

property mtime
modification time of bib file when last parsed

sphinxcontrib.bibtex.bibfile.**normpath_filename**(*env: sphinx.environment.BuildEnvironment*, *filename: str*) → str
Return normalised path to *filename* for the given environment *env*.

sphinxcontrib.bibtex.bibfile.**parse_bibfile**(*bibfilename: str*, *encoding: str*) → pybtex.database.BibliographyData

Parse *bibfilename* with given *encoding*, and return parsed data.

sphinxcontrib.bibtex.bibfile.**process_bibfile**(*bibfiles: Dict[str, sphinxcontrib.bibtex.bibfile.BibFile]*, *bibfilename: str*, *encoding: str*) → None

Check if *bibfiles* is still up to date. If not, parse *bibfilename* and store parsed data in *bibfiles*.

sphinxcontrib.bibtex.bibfile.**get_bibliography_entry**(*bibfiles: Dict[str, sphinxcontrib.bibtex.bibfile.BibFile]*, *key: str*) → Optional[pybtex.database.Entry]

Return bibliography entry from *bibfiles* for the given *key*.

1.4 Changes

1.4.1 2.1.0 (28 December 2020)

- The extension no longer relies on the `bibtex.json` method. Instead, the extension now postpones identifying all citation cross-references to Sphinx's consistency check phase. The actual citation references and bibliography citations are then generated in the resolve phase using post-transforms. As a result, `bibtex.json` is no longer needed and thus Sphinx no longer needs to run twice as in the past if the file did not exist (closes issues #214 and #215). *Thanks to everyone who chimed in on this, especially everyone who made helpful suggestions to find better implementation approaches, and everyone who helped with testing.*
- Citations with multiple keys will now reside in the same bracket (closes issue #94).
- Consistent use of doctutils note_explicit_target to set ids, to ensure no clashing ids.
- Improved and robustified test suite, using regular expressions to verify generated html.
- The test suite now includes a patched version of the awesome but abandoned sphinx-natbib extension, to help comparing and testing implementations and features. The long term intention is to fully support sphinx-natbib style citations.
- **BACKWARD INCOMPATIBLE** The API has been refactored to accommodate the new design. Refer to the API documentation for details.

1.4.2 2.0.0 (12 December 2020)

- There is a new `footcite` role and a new `footbibliography` directive, to allow easy and simple local (per document) bibliographies through footnotes. See issues #184 and #185.
- Parallel builds are now finally supported. See issues #80, #96, and #164, as well as pull request #210.
- **BACKWARD INCOMPATIBLE** To enable parallel builds, a new mandatory config setting `bibtex_bibfiles` has been added. This setting specifies all bib files used throughout the project, relative to the source folder.
- **BACKWARD INCOMPATIBLE** The encoding of bib files has been moved to an optional config setting `bibtex_encoding`. The `:encoding:` option is no longer supported.
- Headers for `bibliography` and `footbibliography` directives can be configured via the `bibtex_bibliography_header` and `bibtex_footbibliography_header` config setting.
- The `bibliography` directive no longer requires the bib files to be specified as an argument. However, if you do, citations will be constrained to those bib files.

- Support newlines/whitespace around cite keys when multiple keys are specified. Thanks to dizcza for help with testing. See issue #205 and pull request #206.
- Improve citation ordering code (reported by ukos-git, see issue #182).
- The unresolved citations across documents issue has been resolved. The extension stores all citation information in a `bibtex.json` file. If it does not exist, the file will be created on your first sphinx build, and you will have to rerun the build to make use of it. The file is automatically kept up to date, with a warning whenever you need to rerun the build. Thanks to dizcza for help with testing. See issues #197 and #204. Also see pull request #208.
- Migrate test suite to pytest, using sphinx's testing fixtures.
- **BACKWARD INCOMPATIBLE** The API has been refactored. Some functions have moved to different modules. Refer to the API documentation for details.
- Drop Python 3.5 support.
- Add Python 3.9 support.

1.4.3 1.0.0 (20 September 2019)

- Drop Python 2.7 and 3.4 support (as upstream sphinx has dropped support for these as well).
- Add Python 3.8 support (contributed by hroncok).
- Update for Sphinx 2.x, and drop Sphinx 1.x support (as there is too much difference between the two versions).
- Non-bibtex citations will now no longer issue warnings (fix contributed by chrisjsewell).
- Switch to codecov for coverage reporting.

1.4.4 0.4.2 (7 January 2018)

- Drop Python 3.3 support, add Python 3.7 support.
- Work around issue with sphinx-testing on Fedora (reported by jamesjer in issue #157, fix contributed by mitya57 in pull request #158).

1.4.5 0.4.1 (28 November 2018)

- Disable tinkerer test due to upstream bug.
- Remove crossref test due to changed upstream behaviour in pybtex.
- Fix latex test to match new upstream code generation.
- Fix documentation of encoding option (contributed by Kai Mühlbauer).
- Migrate to sphinx.util.logging in favour of old deprecated logging method.

1.4.6 0.4.0 (19 April 2018)

- Remove latexcodec and curly bracket strip functionality, as this is now supported by pybtex natively (see issue #127, reported by erosennin).
- Fix tests failures with Sphinx 1.7 (see pull request #136, reported and fixed by mitya57).

1.4.7 0.3.6 (25 September 2017)

- Real fix for issue #111 (again reported by jamesjer).
- Fix test regressions due to latest Sphinx updates (see issues #115, #120, #121, and #122, reported by ndarmage and ghisvail).
- Fix test regressions on ascii locale (see issue #121, reported by ghisvail).
- Support and test Python 3.6.

1.4.8 0.3.5 (22 February 2017)

- Fix extremely high memory usage when handling large bibliographies (reported by agjohnson, see issue #102).
- Fix tests for Sphinx 1.5.1 (see issue #111, reported by jamesjer).

1.4.9 0.3.4 (20 May 2016)

- Document LaTeX workaround for `:cite:` in figure captions (contributed by xuhdev, see issue #92 and pull request #93).
- Add `bibtex_default_style` config value to override the default bibliography style (see issue #91 and pull request #97).
- Support Python 3.5 (see issue #100).

1.4.10 0.3.3 (23 October 2015)

- Add per-bibliography key prefixes, enabling local bibliographies to be used in isolation from each other (see issue #87, reported by marscher).
- Documentation now points to new location of pybtex on bitbucket.
- Simplified testing code by using the new sphinx_testing package.

1.4.11 0.3.2 (20 March 2015)

- Document how to create custom label styles (see issue #77, reported by tino).
- Disable parallel_read_safe for Sphinx 1.3 and later (see issue #80, reported by andreacassioli).

1.4.12 0.3.1 (10 July 2014)

- Fix for `type_.lower()` bug: pybtex 0.18 expects type to be a string (this fixes issue #68 reported by jluttine).

1.4.13 0.3.0 (4 May 2014)

- **BACKWARD INCOMPATIBLE** The alpha style is now default, so citations are labelled in a way that is more standard for Sphinx. To get the old behaviour back, add `:style: plain` to your bibliography directives.
- **BACKWARD INCOMPATIBLE** `is_cited()` has been removed. Use `get_cited_docnames()` instead, which will return an empty list for keys that are not cited.
- Improved support for local bibliographies (see issues #52, #62, and #63; test case provided by Boris Kheyfets):
  - New `docname` and `docnames` filter identifiers.
  - Filter expressions now also support set literals and the operators `in`, `not in`, `&`, and `|`.

  See documentation for details.
- Multiple comma-separated citation keys per cite command (see issue #61, suggested by Boris Kheyfets).
- Add support for pypy and Python 3.4.
- Drop support for Python 2.6 and Python 3.2.
- Drop 2to3 and instead use six to support both Python 2 and 3 from a single code base.
- Simplify instructions for custom styles.
- Various test suite improvements.

1.4.14 0.2.9 (9 October 2013)

- Upgrade to the latest pybtex-docutils to produce more optimal html output (specifically: no more nested `<span>`s).
- Remove latex codec code, and rely on latexcodec package instead.
- `FilterVisitor` has been removed from the public API. Use `get_bibliography_entries()` instead.
- Fix upstream Sphinx bug concerning LaTeX citation hyperlinks (contributed by erikb85; see pull request #45).
- Fix most pylint warnings, refactor code.

1.4.15 0.2.8 (7 August 2013)

- Use pybtex-docutils to remove dependency on pybtex.backends.doctree.

1.4.16 0.2.7 (4 August 2013)

- Integrate with coveralls.io, first release with 100% test coverage.
- Minor bug fixes and code improvements.
- Remove ordereddict dependency for Python 2.7 and higher (contributed by Paul Romano, see pull requests #27 and #28).
- New `:filter:` option for advanced filtering (contributed by d9pouces, see pull requests #30 and #31).
- Refactor documentation of advanced features.
- Document how to create custom pybtex styles (see issues #25, #29, and #34).
- Code is now mostly pep8 compliant.

1.4.17 0.2.6 (2 March 2013)

- For unsorted styles, citation entries are now sorted in the order they are cited, instead of following the order in the bib file, to reflect more closely the way LaTeX handles unsorted styles (addresses issue #15).
- Skip citation label warnings on Sphinx [source] links (issue #17, contributed by Simon Clift).

1.4.18 0.2.5 (18 October 2012)

- Duplicate label detection (issue #14).
- New `:labelprefix:` option to avoid duplicate labels when having multiple bibliographies with a numerical label style (addresses issue #14).

1.4.19 0.2.4 (24 August 2012)

- New options for the bibliography directive for rendering the bibliography as bullet lists or enumerated lists: `:list:`, `:enumtype:`, and `:start:`.
- Minor latex codec fixes.
- Turn exception into warning when a citation cannot be relabeled (fixes issue #2).
- Document LaTeX encoding, and how to turn it off (issue #4).
- Use pybtex labels (fixes issue #6 and issue #7).
- Cache tracked citation keys and labels, and bibliography enumeration counts (fixes issues with citations in repeated Sphinx runs).
- Bibliography ids are now unique across documents (fixes issue that could cause the wrong bibliography to be inserted).
- The plain style is now the default (addresses issue #9).

1.4.20 0.2.3 (30 July 2012)

- Document workaround for Tinkerer (issue #1).
- Use tox for testing.
- Full 2to3 compatibility.
- Document supported versions of Python (2.6, 2.7, 3.1, and 3.2).

1.4.21 0.2.2 (6 July 2012)

- Documentation and manifest fixes.

1.4.22 0.2.1 (19 June 2012)

- First public release.

1.5 License

sphinxcontrib-bibtex is a Sphinx extension for BibTeX style citations
Copyright (c) 2011-2020 by Matthias C. M. Troffaes
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.6 Related Projects

Below is a list of projects which include functionality that is similar or related to sphinxcontrib-bibtex. If you know of any other, leave a message on the issue tracker.

- Andrey Golovizin's pybtex, a general purpose Python library for working with bibtex files. Drives sphinxcontrib-bibtex.
- Matthew Brett's bibstuff. Includes a Sphinx extension similar to sphinxcontrib-bibtex, as well as an assorted collection of bibtex tools. This is a fork of Dylan W. Schwilk and Alan G. Isaac's dschwilk/bibstuff.
- Weston Nielson's sphinx-natbib. This extension is similar to sphinxcontrib-bibtex, and aims to support natbib style citations. Sadly, sphinx-natbib appears no longer maintained and the original repository is no longer available. A patched version of sphinx-natbib, with various bug fixes, is maintained in the test suite of sphinxcontrib-bibtex, for the purpose of comparison. Additionally, a few forks of the original repository can be found on github.
- Jeff Terrace's Sphinx Thesis Resource sphinxtr, is a fork of Sphinx which includes a fork of sphinx-natbib.

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

BIBLIOGRAPHY

[Nel87a] Edward Nelson. *Radically Elementary Probability Theory*. Princeton University Press, 1987.

PYTHON MODULE INDEX

S

INDEX

G

I

K

L

M

N

P

R

S

T