
pyisbn
Release 1.1.0

Jun 15, 2017

Contents

1	Contents	3
1.1	Installation	3
1.2	API documentation	3
1.3	Frequently Asked Questions	7
1.4	Fun and games	8
1.5	Alternatives	8
1.6	Release HOWTO	8
2	Indices and tables	11
	Python Module Index	13

pyisbn is a [GPL v3](#) licensed module for working with various book identification numbers. It includes functions for conversion, verification and generation of checksums. It also includes basic classes for representing ISBNs (International Standard Book Numbers) as objects.

Git repository <https://github.com/JNRowe/pyisbn/>

Issue tracker <https://github.com/JNRowe/pyisbn/issues/>

Contributors <https://github.com/JNRowe/pyisbn/contributors/>

Installation

You can install *pyisbn* either via PYPI (Python Package Index) or from source.

Using PYPI

To install using `pip`:

```
$ pip install pyisbn # to install in Python's site-packages
$ pip install --install-option="--user" pyisbn # to install for a single user
```

To install using `easy_install`:

```
$ easy_install pyisbn
```

From source

If you have downloaded a source tarball you can install it with the following steps:

```
$ python setup.py build
# python setup.py install # to install in Python's site-packages
$ python setup.py install --user # to install for a single user
```

pyisbn has no dependencies outside the standard library.

API documentation

Class based access

Handling ISBNs

The *Isbn* supports SBNs (Standard Book Numbering), ISBN (International Standard Book Number)-10 and -13. If you're handling multiple inputs it is easiest to use this class.

class `pyisbn.Isbn` (*isbn*)

Initialise a new `Isbn` object.

Parameters `isbn` (*str*) – ISBN string

calculate_checksum ()

Calculate ISBN checksum.

Returns ISBN checksum value

Return type `str`

convert (*code*='978')

Convert ISBNs between ISBN-10 and ISBN-13.

Parameters `code` (*str*) – ISBN-13 prefix code

Returns Converted ISBN

Return type `str`

to_url (*site*='amazon', *country*='us')

Generate a link to an online book site.

Parameters

- **site** (*str*) – Site to create link to
- **country** (*str*) – Country specific version of `site`

Returns URL on `site` for book

Return type `str`

Raises

- `SiteError` – Unknown site value
- `CountryError` – Unknown country value

to_urn ()

Generate a RFC 3187 URN.

RFC 3187 is the commonly accepted way to use ISBNs as uniform resource names.

Returns **RFC 3187** compliant URN

Return type `str`

validate ()

Validate an ISBN value.

Returns `True` if ISBN is valid

Return type `bool`

Examples

Validate ISBN

```
>>> book = ISBN('9783540009788')
>>> book.validate()
True
>>> invalid_book = ISBN('0123456654321')
>>> invalid_book.validate()
False
```

Format ISBN

```
>>> book.to_urn()
'URN:ISBN:9783540009788'
>>> book.to_url()
'https://www.amazon.com/s?search-alias=stripbooks&field-isbn=9783540009788'
>>> book.to_url('google')
'https://books.google.com/books?vid=isbn:9783540009788'
```

Handling ISBN-10

class `pyisbn.ISbn10` (*isbn*)

Initialise a new `Isbn10` object.

Parameters `isbn` (*str*) – ISBN-10 string

calculate_checksum ()

Calculate ISBN-10 checksum.

Returns ISBN-10 checksum value

Return type `str`

convert (*code*='978')

Convert ISBN-10 to ISBN-13.

Parameters `code` (*str*) – ISBN-13 prefix code

Returns ISBN-13 string

Return type `str`

Handling ISBN-13

class `pyisbn.ISbn13` (*isbn*)

Initialise a new `Isbn13` object.

Parameters `isbn` (*str*) – ISBN-13 string

calculate_checksum ()

Calculate ISBN-13 checksum.

Returns ISBN-13 checksum value

Return type `str`

convert (*code*=None)

Convert ISBN-13 to ISBN-10.

Parameters `code` – Ignored, only for compatibility with `Isbn`

Returns ISBN-10 string

Return type `str`

Raises `ValueError` – When ISBN-13 isn't a Bookland "978" ISBN

Handling SBNs

class `pyisbn.Sbn` (*sbn*)

Initialise a new `Sbn` object.

Parameters `sbn` (*str*) – SBN string

calculate_checksum ()

Calculate SBN checksum.

Returns SBN checksum value

Return type `str`

convert (*code*='978')

Convert SBN to ISBN-13.

Parameters `code` (*str*) – ISBN-13 prefix code

Returns ISBN-13 string

Return type `str`

Exceptions

exception `pyisbn.CountryError`

Unknown country value.

exception `pyisbn.IsbnError`

Invalid ISBN string.

exception `pyisbn.SiteError`

Unknown site value.

Function based access

Additionally the top-level functions are available, if you do not wish to use the classes.

Note: While the layout of this module is a result of it moving from a strictly function-based layout to a class-based layout these functions will not be removed. Backwards compatibility is important, and will be maintained.

Functions for handling ISBNs

`pyisbn.calculate_checksum` (*isbn*)

Calculate ISBN checksum.

Parameters `isbn` (*str*) – SBN, ISBN-10 or ISBN-13

Returns Checksum for given ISBN or SBN

Return type `str`

```
>>> calculate_checksum('978354000978')
'8'
```

`pyisbn.convert(isbn, code='978')`
Convert ISBNs between ISBN-10 and ISBN-13.

Note: No attempt to hyphenate converted ISBNs is made, because the specification requires that *any* hyphenation must be correct but allows ISBNs without hyphenation.

Parameters

- **isbn** (`str`) – SBN, ISBN-10 or ISBN-13
- **code** (`str`) – EAN Bookland code

Returns Converted ISBN-10 or ISBN-13

Return type `str`

Raise: `IsbnError`: When ISBN-13 isn't convertible to an ISBN-10

```
>>> convert('9783540009788')
'3540009787'
```

`pyisbn.validate(isbn)`
Validate ISBNs.

Warning: Publishers have been known to go to press with broken ISBNs, and therefore validation failures do not completely guarantee an ISBN is incorrectly entered. It should however be noted that it is massively more likely *you* have entered an invalid ISBN than the published ISBN is incorrectly produced. An example of this probability in the real world is that [Amazon](#) consider it so unlikely that they refuse to search for invalid published ISBNs.

Parameters **isbn** (`str`) – SBN, ISBN-10 or ISBN-13

Returns `True` if ISBN is valid

Return type `bool`

```
>>> validate('9783540009788')
True
```

Frequently Asked Questions

Can you release this under a more permissive licence?

I'm sorry, but no.

For pet projects I choose to use reciprocal licences because I like them, not because I'm unaware of their impact.

Can we buy a licence to embed this within a closed source product?

I'm sorry, but no.

This isn't an issue of money, so there is no need to make each other feel uncomfortable with a bidding discussion.

Fun and games

With ISBN-13 a book can have a valid checksum and have a simple transcription error, if digits with a difference of five are transposed.

Using “The Statistical Mechanics of Financial Markets” as an example, we can see that 9783540009788 is the [given ISBN](#), and is valid. However, 9738540009788 with the third and fourth characters transposed is also valid, yet is incorrect¹.

I'll leave it as an exercise for the reader to figure out how often books with transposable ISBN-13 occur in a given library of n books.

Alternatives

Before diving in and spitting out this package I looked for alternatives, but back in 2006 there were none to be found. There are, now, a few available and I'll list them here when people point them out. If I have missed something please drop me a [mail](#).

`python-stdnum`

`python-stdnum` by Arthur de Jong is a package to validate identifiers in a huge range of formats, including ISBNs.

Release HOWTO

Test

In the general case tests can be run via `nose2`:

```
$ nose2 tests
```

When preparing a release it is important to check that `pyisbn` works with all currently supported Python versions, and that the documentation is correct.

Prepare release

With the tests passing, perform the following steps

- Update the version data in `pyisbn/_version.py`
- Update `NEWS.rst`, if there are any user visible changes
- Commit the release notes and version changes

¹ This example was chosen to show that sometimes it is still possible to catch during data entry as 973 isn't a valid prefix

- Create a signed tag for the release
- Push the changes, including the new tag, to the GitHub repository

Update PyPI

Create and upload the new release tarballs to PyPI:

```
$ ./setup.py sdist bdist_wheel register upload --sign
```

Fetch the uploaded tarballs, and check for errors.

You should also perform test installations from PyPI, to check the experience *pyisbn* users will have.

CHAPTER 2

Indices and tables

- `genindex`
- `modindex`
- `search`

p

pyisbn, 1

C

calculate_checksum() (in module pyisbn), 6
calculate_checksum() (pyisbn.Isbn method), 4
calculate_checksum() (pyisbn.Isbn10 method), 5
calculate_checksum() (pyisbn.Isbn13 method), 5
calculate_checksum() (pyisbn.Sbn method), 6
convert() (in module pyisbn), 7
convert() (pyisbn.Isbn method), 4
convert() (pyisbn.Isbn10 method), 5
convert() (pyisbn.Isbn13 method), 5
convert() (pyisbn.Sbn method), 6
CountryError, 6

I

Isbn (class in pyisbn), 4
Isbn10 (class in pyisbn), 5
Isbn13 (class in pyisbn), 5
IsbnError, 6

P

pyisbn (module), 1

R

RFC
RFC 3187, 4

S

Sbn (class in pyisbn), 6
SiteError, 6

T

to_url() (pyisbn.Isbn method), 4
to_urn() (pyisbn.Isbn method), 4

V

validate() (in module pyisbn), 7
validate() (pyisbn.Isbn method), 4