pyisbn

Release 1.1.0

Apr 05, 2017

Contents

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 as objects.

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

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

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

CHAPTER 1

Contents

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, ISBN-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

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

Handling ISBN-13

class pyisbn.**Isbn13**(*isbn*)
Initialise a new Isbn13 object.

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

Handling SBNs

class pyisbn.**Sbn**(*sbn*)
Initialise a new Sbn object.

Parameters **sbn** (*str*) – SBN string

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

Sadly, these two questions come up often enough that I've felt the need to write a pre-emptive response here. I hoping it will allow me to skip the awkwardness I feel when writing individual replies...

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 incorrect[1]_ .

I'll leave it as an exercise for the reader to figure out how often books with transposable ISBN-13s 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 -vv 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
- 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

Python Module Index

p

Index