pybase64 Documentation

Release 1.3.2

Matthieu Darbois

Jan 01, 2024

CONTENTS:

1 Getting started **3**
- 1.1 Installation . . . 3
- 1.2 Usage . . . 3
- 1.3 Benchmark . . . 4

2 API Reference **5**
- 2.1 Main API Reference . . . 5
- 2.2 Helpers API Reference . . . 6
- 2.3 Legacy API Reference . . . 7
- 2.4 Information API Reference . . . 7

3 Changelog **9**
- 3.1 1.3.2 . . . 9
- 3.2 1.3.1 . . . 9
- 3.3 1.3.0 . . . 9
- 3.4 1.2.3 . . . 9
- 3.5 1.2.2 . . . 10
- 3.6 1.2.1 . . . 10
- 3.7 1.2.0 . . . 10
- 3.8 1.1.4 . . . 10
- 3.9 1.1.3 . . . 10
- 3.10 1.1.2 . . . 10
- 3.11 1.1.1 . . . 10
- 3.12 1.1.0 . . . 11
- 3.13 1.0.2 . . . 11
- 3.14 1.0.1 . . . 11
- 3.15 1.0.0 . . . 11
- 3.16 0.5.0 . . . 11
- 3.17 0.4.0 . . . 11
- 3.18 0.3.1 . . . 11
- 3.19 0.3.0 . . . 12
- 3.20 0.2.1 . . . 12
- 3.21 0.2.0 . . . 12
- 3.22 0.1.2 . . . 12
- 3.23 0.1.1 . . . 12
- 3.24 0.1.0 . . . 12

4 License **13**
- 4.1 pybase64 . . . 13
- 4.2 libbase64 . . . 13

Index 15

Fast Base64 implementation for Python.

CHAPTER

ONE

GETTING STARTED

pybase64 is a wrapper on libbase64.

It aims to provide a fast base64 implementation for base64 encoding/decoding.

1.1 Installation

```
pip install pybase64
```

1.2 Usage

pybase64 uses the same API as Python base64 "modern interface" (introduced in Python 2.4) for an easy integration.

To get the fastest decoding, it is recommended to use the *b64decode()* and *validate=True* when possible.

```
import pybase64

print(pybase64.b64encode(b'>>>foo???', altchars='_:'))
# b'Pj4_Zm9vPz8:'
print(pybase64.b64decode(b'Pj4_Zm9vPz8:', altchars='_:', validate=True))
# b'>>>foo???'

# Standard encoding helpers
print(pybase64.standard_b64encode(b'>>>foo???'))
# b'Pj4+Zm9vPz8/'
print(pybase64.standard_b64decode(b'Pj4+Zm9vPz8/'))
# b'>>>foo???'

# URL safe encoding helpers
print(pybase64.urlsafe_b64encode(b'>>>foo???'))
# b'Pj4-Zm9vPz8_'
print(pybase64.urlsafe_b64decode(b'Pj4-Zm9vPz8_'))
# b'>>>foo???'
```

Check *API Reference* for more details.

A command-line tool is also provided. It has encode, decode and benchmark subcommands.

```
usage: pybase64 [-h] [-V] {benchmark,encode,decode} ...

pybase64 command-line tool.

positional arguments:
  {benchmark,encode,decode}
                        tool help
    benchmark           -h for usage
    encode              -h for usage
    decode              -h for usage

optional arguments:
  -h, --help            show this help message and exit
  -V, --version         show program's version number and exit
```

1.3 Benchmark

Running Python 3.7.2, Apple LLVM version 10.0.0 (clang-1000.11.45.5), Mac OS X 10.14.2 on an Intel Core i7-4870HQ @ 2.50GHz

```
pybase64 0.5.0 (C extension active - AVX2)
bench: altchars=None, validate=False
pybase64._pybase64.encodebytes:   1734.776 MB/s (13,271,472 bytes -> 17,928,129 bytes)
pybase64._pybase64.b64encode:     4039.539 MB/s (13,271,472 bytes -> 17,695,296 bytes)
pybase64._pybase64.b64decode:     1854.423 MB/s (17,695,296 bytes -> 13,271,472 bytes)
base64.encodebytes:                 78.352 MB/s (13,271,472 bytes -> 17,928,129 bytes)
base64.b64encode:                  539.840 MB/s (13,271,472 bytes -> 17,695,296 bytes)
base64.b64decode:                  287.826 MB/s (17,695,296 bytes -> 13,271,472 bytes)
bench: altchars=None, validate=True
pybase64._pybase64.b64encode:     4156.607 MB/s (13,271,472 bytes -> 17,695,296 bytes)
pybase64._pybase64.b64decode:     4107.997 MB/s (17,695,296 bytes -> 13,271,472 bytes)
base64.b64encode:                  559.342 MB/s (13,271,472 bytes -> 17,695,296 bytes)
base64.b64decode:                  143.674 MB/s (17,695,296 bytes -> 13,271,472 bytes)
bench: altchars=b'-_', validate=False
pybase64._pybase64.b64encode:     2786.776 MB/s (13,271,472 bytes -> 17,695,296 bytes)
pybase64._pybase64.b64decode:     1124.136 MB/s (17,695,296 bytes -> 13,271,472 bytes)
base64.b64encode:                  322.427 MB/s (13,271,472 bytes -> 17,695,296 bytes)
base64.b64decode:                  205.195 MB/s (17,695,296 bytes -> 13,271,472 bytes)
bench: altchars=b'-_', validate=True
pybase64._pybase64.b64encode:     2806.271 MB/s (13,271,472 bytes -> 17,695,296 bytes)
pybase64._pybase64.b64decode:     2740.456 MB/s (17,695,296 bytes -> 13,271,472 bytes)
base64.b64encode:                  314.709 MB/s (13,271,472 bytes -> 17,695,296 bytes)
base64.b64decode:                  121.803 MB/s (17,695,296 bytes -> 13,271,472 bytes)
```

CHAPTER

TWO

API REFERENCE

2.1 Main API Reference

pybase64.**b64encode**(*s: Any*, *altchars: Any = None*) → bytes

Encode bytes using the standard Base64 alphabet.

Argument `s` is a bytes-like object to encode.

Optional `altchars` must be a byte string of length 2 which specifies an alternative alphabet for the '+' and '/' characters. This allows an application to e.g. generate url or filesystem safe Base64 strings.

The result is returned as a `bytes` object.

pybase64.**b64encode_as_string**(*s: Any*, *altchars: Any = None*) → str

Encode bytes using the standard Base64 alphabet.

Argument `s` is a bytes-like object to encode.

Optional `altchars` must be a byte string of length 2 which specifies an alternative alphabet for the '+' and '/' characters. This allows an application to e.g. generate url or filesystem safe Base64 strings.

The result is returned as a `str` object.

pybase64.**b64decode**(*s: Any*, *altchars: Any = None*, *validate: bool = False*) → bytes

Decode bytes encoded with the standard Base64 alphabet.

Argument `s` is a bytes-like object or ASCII string to decode.

Optional `altchars` must be a bytes-like object or ASCII string of length 2 which specifies the alternative alphabet used instead of the '+' and '/' characters.

If `validate` is `False` (the default), characters that are neither in the normal base-64 alphabet nor the alternative alphabet are discarded prior to the padding check. If `validate` is `True`, these non-alphabet characters in the input result in a `binascii.Error`.

The result is returned as a `bytes` object.

A `binascii.Error` is raised if `s` is incorrectly padded.

pybase64.**b64decode_as_bytearray**(*s: Any*, *altchars: Any = None*, *validate: bool = False*) → bytearray

Decode bytes encoded with the standard Base64 alphabet.

Argument `s` is a bytes-like object or ASCII string to decode.

Optional `altchars` must be a bytes-like object or ASCII string of length 2 which specifies the alternative alphabet used instead of the '+' and '/' characters.

If `validate` is `False` (the default), characters that are neither in the normal base-64 alphabet nor the alternative alphabet are discarded prior to the padding check. If `validate` is `True`, these non-alphabet characters in the input result in a `binascii.Error`.

The result is returned as a `bytearray` object.

A `binascii.Error` is raised if `s` is incorrectly padded.

2.2 Helpers API Reference

pybase64.**standard_b64encode**(*s: Any*) → bytes

Encode bytes using the standard Base64 alphabet.

Argument `s` is a bytes-like object to encode.

The result is returned as a `bytes` object.

pybase64.**standard_b64decode**(*s: Any*) → bytes

Decode bytes encoded with the standard Base64 alphabet.

Argument `s` is a bytes-like object or ASCII string to decode.

The result is returned as a `bytes` object.

A `binascii.Error` is raised if the input is incorrectly padded.

Characters that are not in the standard alphabet are discarded prior to the padding check.

pybase64.**urlsafe_b64encode**(*s: Any*) → bytes

Encode bytes using the URL- and filesystem-safe Base64 alphabet.

Argument `s` is a bytes-like object to encode.

The result is returned as a `bytes` object.

The alphabet uses '-' instead of '+' and '_' instead of '/'.

pybase64.**urlsafe_b64decode**(*s: Any*) → bytes

Decode bytes using the URL- and filesystem-safe Base64 alphabet.

Argument `s` is a bytes-like object or ASCII string to decode.

The result is returned as a `bytes` object.

A `binascii.Error` is raised if the input is incorrectly padded.

Characters that are not in the URL-safe base-64 alphabet, and are not a plus '+' or slash '/', are discarded prior to the padding check.

The alphabet uses '-' instead of '+' and '_' instead of '/'.

2.3 Legacy API Reference

pybase64.**encodebytes**(*s: Any*) → bytes

Encode bytes into a bytes object with newlines (b'\n') inserted after every 76 bytes of output, and ensuring that there is a trailing newline, as per **RFC 2045** (MIME).

Argument `s` is a bytes-like object to encode.

The result is returned as a `bytes` object.

2.4 Information API Reference

pybase64.**get_version**() → str

Returns pybase64 version as a `str` object.

The result reports if the C extension is used or not. e.g. *1.0.0 (C extension active - AVX2)*

pybase64.**get_license_text**() → str

Returns pybase64 license information as a `str` object.

The result includes libbase64 license information as well.

CHAPTER

THREE

CHANGELOG

3.1 1.3.2

- Update base64 library
- PyPy: fix wrong outcome with non C-contiguous buffer

3.2 1.3.1

- Add missing py.typed marker

3.3 1.3.0

- Update base64 library
- Add AVX512-VBMI implementation
- Rework extension build to remove adherence on distutils
- Publish python 3.12 wheels
- Documentation now uses furo theme

3.4 1.2.3

- Update base64 library
- Publish python 3.11 wheels

3.5 1.2.2

- Update base64 library
- Fix C extension build on musl distros
- Publish musllinux wheels

3.6 1.2.1

- Publish PyPy 3.8 (pypy38_pp73) wheels

3.7 1.2.0

- Release the GIL
- Publish CPython 3.10 wheels
- Drop python 3.5 support

3.8 1.1.4

- Add macOS arm64 wheel

3.9 1.1.3

- GitHub Actions: fix build on tag

3.10 1.1.2

- Add PyPy wheels
- Add aarch64, ppc64le & s390x manylinux wheels

3.11 1.1.1

- Move CI from TravisCI/AppVeyor to GitHub Actions
- Fix publication of Linux/macOS wheels

3.12 1.1.0

- Add b64encode_as_string, same as b64encode but returns a str object instead of a bytes object
- Add b64decode_as_bytearray, same as b64decode but returns a bytarray object instead of a bytes object
- Speed-Up decoding from UCS1 strings

3.13 1.0.2

- Update base64 library
- Publish python 3.9 wheels

3.14 1.0.1

- Publish python 3.8 wheels

3.15 1.0.0

- Drop python 3.4 support
- Drop python 2.7 support

3.16 0.5.0

- Publish python 3.7 wheels
- Drop python 3.3 support

3.17 0.4.0

- Speed-up decoding when validate==False

3.18 0.3.1

- Fix deployment issues

3.19 0.3.0

- Add encodebytes function

3.20 0.2.1

- Fixed invalid results on Windows

3.21 0.2.0

- Added documentation
- Added subcommands to the main script:
  - help
  - version
  - encode
  - decode
  - benchmark

3.22 0.1.2

- Updated base64 native library

3.23 0.1.1

- Fixed deployment issues

3.24 0.1.0

- First public release

CHAPTER

FOUR

LICENSE

4.1 pybase64

```
BSD 2-Clause License

Copyright (c) 2017-2022, Matthieu Darbois
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 HOLDER 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.
```

4.2 libbase64

```
Copyright (c) 2005-2007, Nick Galbreath
Copyright (c) 2015-2018, Wojciech Muła
Copyright (c) 2016-2017, Matthieu Darbois
Copyright (c) 2013-2022, Alfred Klomp
All rights reserved.

Redistribution and use in source and binary forms, with or without
```

(continues on next page)

(continued from previous page)

```
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
HOLDER 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.
```

INDEX

B

b64decode() (*in module pybase64*), 5
b64decode_as_bytearray() (*in module pybase64*), 5
b64encode() (*in module pybase64*), 5
b64encode_as_string() (*in module pybase64*), 5

E

encodebytes() (*in module pybase64*), 7

G

get_license_text() (*in module pybase64*), 7
get_version() (*in module pybase64*), 7

R

RFC
    RFC 2045, 7

S

standard_b64decode() (*in module pybase64*), 6
standard_b64encode() (*in module pybase64*), 6

U

urlsafe_b64decode() (*in module pybase64*), 6
urlsafe_b64encode() (*in module pybase64*), 6