

---

# **pybase64 Documentation**

***Release 0.2.1***

**Matthieu Darbois**

**Sep 03, 2017**



---

## Contents:

---

<b>1</b>	<b>Getting started</b>	<b>3</b>
1.1	Installation . . . . .	3
1.2	Usage . . . . .	3
1.3	Benchmark . . . . .	4
<b>2</b>	<b>API Reference</b>	<b>5</b>
2.1	Main API Reference . . . . .	5
2.2	Helpers API Reference . . . . .	5
2.3	Information API Reference . . . . .	6
<b>3</b>	<b>License</b>	<b>7</b>
3.1	pybase64 . . . . .	7
3.2	libbase64 . . . . .	8



Fast Base64 implementation for Python.



# CHAPTER 1

---

## 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 3.6.0, Apple LLVM version 8.1.0 (clang-802.0.42), Mac OS X 10.12.6 on an Intel Core i7-4870HQ @ 2.50GHz

```
pybase64 0.2.0 (C extension active - AVX2)
bench: altchars=None, validate=False
pybase64._pybase64.b64encode:      2941.397 MB/s (13,271,472 bytes -> 17,695,296 bytes)
pybase64._pybase64.b64decode:      328.250 MB/s (17,695,296 bytes -> 13,271,472 bytes)
base64.b64encode:                   565.744 MB/s (13,271,472 bytes -> 17,695,296 bytes)
base64.b64decode:                   327.075 MB/s (17,695,296 bytes -> 13,271,472 bytes)
bench: altchars=None, validate=True
pybase64._pybase64.b64encode:      2995.909 MB/s (13,271,472 bytes -> 17,695,296 bytes)
pybase64._pybase64.b64decode:      3996.267 MB/s (17,695,296 bytes -> 13,271,472 bytes)
base64.b64encode:                   577.565 MB/s (13,271,472 bytes -> 17,695,296 bytes)
base64.b64decode:                   104.835 MB/s (17,695,296 bytes -> 13,271,472 bytes)
bench: altchars=b'-'_, validate=False
pybase64._pybase64.b64encode:      2237.740 MB/s (13,271,472 bytes -> 17,695,296 bytes)
pybase64._pybase64.b64decode:      262.021 MB/s (17,695,296 bytes -> 13,271,472 bytes)
base64.b64encode:                   313.977 MB/s (13,271,472 bytes -> 17,695,296 bytes)
base64.b64decode:                   219.487 MB/s (17,695,296 bytes -> 13,271,472 bytes)
bench: altchars=b'-'_, validate=True
pybase64._pybase64.b64encode:      2349.481 MB/s (13,271,472 bytes -> 17,695,296 bytes)
pybase64._pybase64.b64decode:      2790.047 MB/s (17,695,296 bytes -> 13,271,472 bytes)
base64.b64encode:                   314.182 MB/s (13,271,472 bytes -> 17,695,296 bytes)
base64.b64decode:                   89.855 MB/s (17,695,296 bytes -> 13,271,472 bytes)
```



### 2.1 Main API Reference

`pybase64.b64encode(s, altchars=None)`

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.b64decode(s, altchars=None, validate=False)`

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.

### 2.2 Helpers API Reference

`pybase64.standard_b64encode(s)`

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)`

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)`

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)`

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 Information API Reference

`pybase64.get_version()`

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()`

Returns pybase64 license information as a `str` object.

The result includes libbase64 license information as well.

### 3.1 pybase64

BSD 2-Clause License

Copyright (c) 2017, 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.

## 3.2 libbase64

```
Copyright (c) 2005-2007, Nick Galbreath
Copyright (c) 2013-2017, Alfred Klomp
Copyright (c) 2015-2017, Wojciech Mula
Copyright (c) 2016-2017, 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.

### B

[b64decode\(\)](#) (in module `pybase64`), [5](#)  
[b64encode\(\)](#) (in module `pybase64`), [5](#)

### G

[get\\_license\\_text\(\)](#) (in module `pybase64`), [6](#)  
[get\\_version\(\)](#) (in module `pybase64`), [6](#)

### S

[standard\\_b64decode\(\)](#) (in module `pybase64`), [6](#)  
[standard\\_b64encode\(\)](#) (in module `pybase64`), [5](#)

### U

[urlsafe\\_b64decode\(\)](#) (in module `pybase64`), [6](#)  
[urlsafe\\_b64encode\(\)](#) (in module `pybase64`), [6](#)