base64io

Release 1.0.3

Jan 24, 2023

Contents

1 Getting Started **3**
- 1.1 Installation . . . 3

2 Use **5**
- 2.1 Encode data . . . 5
- 2.2 Decode data . . . 6

3 License **7**

4 Modules **9**
- 4.1 base64io . . . 9

5 Changelog **13**
- 5.1 1.0.3 – 2018-12-10 . . . 13
- 5.2 1.0.2 – 2018-08-01 . . . 13
- 5.3 1.0.1 – 2018-06-12 . . . 13
- 5.4 1.0.0 – 2018-05-16 . . . 13

Python Module Index **15**

Index **17**

This project is designed to develop a class, *base64io.Base64IO*, that implements a streaming interface for Base64 encoding.

Python has supported native Base64 encoding since version 2.4. However, there is no streaming interface for Base64 encoding, and none is available from the community.

The legacy `base64.encode` and `base64.decode` interface lets you shuffle data between two streams, but it assumes that you have two complete streams. We wanted a standard stream that applies Base64 encoding and decoding.

base64io.Base64IO provides an *io* streaming interface with context manager support that transparently Base64-encodes data read from it. You can use it to transform large files without caching the entire context in memory, or to transform an existing stream.

For the latest full documentation, see Read the Docs.

Find us on GitHub.

Security issue notifications

CHAPTER 1

Getting Started

base64io.Base64IO has no dependencies other than the standard library and should work with any version of Python greater than 2.6. We test it on CPython 2.6, 2.7, 3.3, 3.4, 3.5, 3.6, and 3.7.

1.1 Installation

```
$ pip install base64io
```

CHAPTER 2

Use

base64io.Base64IO wraps the input stream and transparently encodes or decodes data written to or read from the input stream.

- `write()` encodes data before writing it to the wrapped stream
- `read()` decodes data after reading it from the wrapped stream

Because the position of the *base64io.Base64IO* stream and the wrapped stream will almost always be different, *base64io.Base64IO* does not support:

- `seek()`
- `tell()`

Also, *base64io.Base64IO* does not support:

- `fileno()`
- `truncate()`

2.1 Encode data

Warning: If you are not using *base64io.Base64IO* as a context manager, when you write to a *base64io.Base64IO* stream, you **must** close the stream after your final write. The Base64 transformation might hold up to two bytes of unencoded data in an internal buffer before writing it to the wrapped stream. Calling `close()` flushes this buffer and writes the padded result to the wrapped stream. The *base64io.Base64IO* context manager does this for you.

```
from base64io import Base64IO

with open("source_file", "rb") as source, open("encoded_file", "wb") as target:
    with Base64IO(target) as encoded_target:
```

(continues on next page)

(continued from previous page)

```
        for line in source:
            encoded_target.write(line)
```

2.2 Decode data

Note: When it reads data from the wrapping stream, it might read up to three additional bytes from the underlying stream.

```
from base64io import Base64IO

with open("encoded_file", "rb") as encoded_source, open("target_file", "wb") as␣
↪target:
    with Base64IO(encoded_source) as source:
        for line in source:
            target.write(line)
```

CHAPTER 3

License

This library is licensed under the Apache 2.0 License.

CHAPTER 4

Modules

base64io	Base64 stream with context manager support.

4.1 base64io

Base64 stream with context manager support.

Classes

Base64IO(wrapped)	Base64 stream with context manager support.

class base64io.**Base64IO**(*wrapped*)

Bases: io.IOBase

Base64 stream with context manager support.

Wraps a stream, base64-decoding read results before returning them and base64-encoding written bytes before writing them to the stream. Instances of this class are not reusable in order maintain consistency with the io.IOBase behavior on close().

Note: Provides iterator and context manager interfaces.

Warning: Because up to two bytes of data must be buffered to ensure correct base64 encoding of all data written, this object **must** be closed after you are done writing to avoid data loss. If used as a context manager, we take care of that for you.

Parameters **wrapped** – Stream to wrap

Check for required methods on wrapped stream and set up read buffer.

Raises **TypeError** – if `wrapped` does not have attributes needed to determine the stream's state

close()
Close this stream, encoding and writing any buffered bytes is present.

Note: This does **not** close the wrapped stream.

---

writable()
Determine if the stream can be written to.

Delegates to wrapped stream when possible. Otherwise returns False.

Return type bool

readable()
Determine if the stream can be read from.

Delegates to wrapped stream when possible. Otherwise returns False.

Return type bool

flush()
Flush the write buffer of the wrapped stream.

write(*b*)
Base64-encode the bytes and write them to the wrapped stream.

Any bytes that would require padding for the next write call are buffered until the next write or close.

Warning: Because up to two bytes of data must be buffered to ensure correct base64 encoding of all data written, this object **must** be closed after you are done writing to avoid data loss. If used as a context manager, we take care of that for you.

Parameters **b** (*bytes*) – Bytes to write to wrapped stream

Raises

- **ValueError** – if called on closed Base64IO object
- **IOError** – if underlying stream is not writable

writelines(*lines*)
Write a list of lines.

Parameters **lines** (*list*) – Lines to write

read(*b=-1*)
Read bytes from wrapped stream, base64-decoding before return.

Note: The number of bytes requested from the wrapped stream is adjusted to return the requested number of bytes after decoding returned bytes.

Parameters **b** (*int*) – Number of bytes to read

Returns Decoded bytes from wrapped stream

Return type bytes

readline(*limit=-1*)
Read and return one line from the stream.

If limit is specified, at most limit bytes will be read.

Note: Because the source that this reads from may not contain any OEL characters, we read "lines" in chunks of length `io.DEFAULT_BUFFER_SIZE`.

Return type bytes

readlines(*hint=-1*)
Read and return a list of lines from the stream.

`hint` can be specified to control the number of lines read: no more lines will be read if the total size (in bytes/characters) of all lines so far exceeds hint.

Returns Lines of data

Return type list of bytes

next()
Python 2 iterator hook.

CHAPTER 5

Changelog

5.1 1.0.3 – 2018-12-10

- Add support for strings on input for decoding to match functionality of `base64.b64decode`. #21 #23 #24

5.2 1.0.2 – 2018-08-01

- Move the `base64io-python` repository from `awslabs` to `aws`.

5.3 1.0.1 – 2018-06-12

- Fix minor compatibility issues with `read()` and `__exit__()` #6

5.4 1.0.0 – 2018-05-16

- Initial public release

Python Module Index

b

base64io, 9

Index

B

`Base64IO` (*class in base64io*), 9
`base64io` (*module*), 9

C

`close()` (*base64io.Base64IO method*), 10

F

`flush()` (*base64io.Base64IO method*), 10

N

`next()` (*base64io.Base64IO method*), 11

R

`read()` (*base64io.Base64IO method*), 10
`readable()` (*base64io.Base64IO method*), 10
`readline()` (*base64io.Base64IO method*), 11
`readlines()` (*base64io.Base64IO method*), 11

W

`writable()` (*base64io.Base64IO method*), 10
`write()` (*base64io.Base64IO method*), 10
`writelines()` (*base64io.Base64IO method*), 10