



pySerial-asyncio Documentation

Release 0.4

pySerial-team

May 07, 2017

Contents

1	Short introduction	3
2	pySerial-asyncio API	5
2.1	asyncio	5
3	Appendix	7
3.1	License	7
4	Indices and tables	9
	Python Module Index	11

Async I/O extension for the [Python Serial Port](#) package for OSX, Linux, BSD

It depends on pySerial and is compatible with Python 3.4 and later.

Other pages (online)

- [project page on GitHub](#)
- [Download Page with releases](#)
- This page, when viewed online is at <https://pyserial-asyncio.readthedocs.io/en/latest/> or <http://pythonhosted.org/pyserial-asyncio/> .

Contents:

Example:

```
import asyncio
import serial_asyncio

class Output(asyncio.Protocol):
    def connection_made(self, transport):
        self.transport = transport
        print('port opened', transport)
        transport.serial.rts = False # You can manipulate Serial object via transport
        transport.write(b'Hello, World!\n') # Write serial data via transport

    def data_received(self, data):
        print('data received', repr(data))
        if b'\n' in data:
            self.transport.close()

    def connection_lost(self, exc):
        print('port closed')
        self.transport.loop.stop()

    def pause_writing(self):
        print('pause writing')
        print(self.transport.get_write_buffer_size())

    def resume_writing(self):
        print(self.transport.get_write_buffer_size())
        print('resume writing')

loop = asyncio.get_event_loop()
coro = serial_asyncio.create_serial_connection(loop, Output, '/dev/ttyUSB0',
↳baudrate=115200)
loop.run_until_complete(coro)
loop.run_forever()
loop.close()
```


asyncio

Warning: This implementation is currently in an experimental state. Use at your own risk.

Experimental asyncio support is available for Python 3.4 and newer. The module `serial_asyncio` provides a `asyncio.Transport: SerialTransport`.

A factory function (`asyncio.coroutine`) is provided:

`serial_asyncio.create_serial_connection(loop, protocol_factory, *args, **kwargs)`

Parameters

- **loop** – The event handler
- **protocol_factory** – Factory function for a `asyncio.Protocol`
- **args** – Passed to the `serial.Serial` init function
- **kwargs** – Passed to the `serial.Serial` init function

Platform Posix

Get a connection making coroutine.

Example:

```
class Output(asyncio.Protocol):
    def connection_made(self, transport):
        self.transport = transport
        print('port opened', transport)
        transport.serial.rts = False
        transport.write(b'hello world\n')

    def data_received(self, data):
```

```
        print('data received', repr(data))
        self.transport.close()

    def connection_lost(self, exc):
        print('port closed')
        asyncio.get_event_loop().stop()

loop = asyncio.get_event_loop()
coro = serial_asyncio.create_serial_connection(loop, Output, '/dev/ttyUSB0',
↳baudrate=115200)
loop.run_until_complete(coro)
loop.run_forever()
loop.close()
```

License

Copyright (c) 2015-2017 pySerial-team (see CREDITS.rst) 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.
- Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

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.

CHAPTER 4

Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)

S

`serial_asyncio`, 5

C

`create_serial_connection()` (in module `serial_asyncio`), 5

S

`serial_asyncio` (module), 5