pySerial-asyncio Documentation

Release 0.4

pySerial-team

May 07, 2017

Contents

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:

CHAPTER 1

Short introduction

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()
```

CHAPTER 2

pySerial-asyncio API

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()
```

CHAPTER 3

Appendix

License

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

Index