Short introduction

Opening serial ports

Open port at “9600,8,N,1”, no timeout:

```
>>> import serial
>>> ser = serial.Serial('/dev/ttyUSB0')  # open serial port
>>> print(ser.name)         # check which port was really used
>>> ser.write(b'hello')     # write a string
>>> ser.close()             # close port
```

Open named port at “19200,8,N,1”, 1s timeout:

```
>>> with serial.Serial('/dev/ttyS1', 19200, timeout=1) as ser:
...     x = ser.read()          # read one byte
...     s = ser.read(10)        # read up to ten bytes (timeout)
...     line = ser.readline()   # read a '\n' terminated line
```

Open port at “38400,8,E,1”, non blocking HW handshaking:

```
>>> ser = serial.Serial('COM3', 38400, timeout=0,
...                     parity=serial.PARITY_EVEN, rtscts=1)
>>> s = ser.read(100)       # read up to one hundred bytes
...                         # or as much is in the buffer
```

Configuring ports later

Get a Serial instance and configure/open it later:

```
>>> ser = serial.Serial()
>>> ser.baudrate = 19200
>>> ser.port = 'COM1'
>>> ser
Serial<id=0xa81c10, open=False>(port='COM1', baudrate=19200, bytesize=8, parity='N', stopbits=1,
timeout=None, xonxoff=0, rtscts=0)
>>> ser.open()
>>> ser.is_open
True
>>> ser.close()
>>> ser.is_open
False
```

Also supported with context manager:

```
with serial.Serial() as ser:
    ser.baudrate = 19200
    ser.port = 'COM1'
    ser.open()
    ser.write(b'hello')
```

Readline

Be careful when using `readline()`. Do specify a timeout when opening the serial port otherwise it could block forever if no newline character is received. Also note that `readlines()` only works with a timeout. `readlines()` depends on having a timeout and interprets that as EOF (end of file). It raises an exception if the port is not opened correctly.

Do also have a look at the example files in the examples directory in the source distribution or online.

Note

The `eol` parameter for `readline()` is no longer supported when pySerial is run with newer Python versions (V2.6+) where the module `io` is available.

EOL

To specify the EOL character for `readline()` or to use universal newline mode, it is advised to use io.TextIOWrapper:

```
import serial
import io
ser = serial.serial_for_url('loop://', timeout=1)
sio = io.TextIOWrapper(io.BufferedRWPair(ser, ser))

sio.write(unicode("hello\n"))
sio.flush() # it is buffering. required to get the data out *now*
hello = sio.readline()
print(hello == unicode("hello\n"))
```

Testing ports

Listing ports

`python -m serial.tools.list_ports` will print a list of available ports. It is also possible to add a regexp

as first argument and the list will only include entries that matched.

Note

The enumeration may not work on all operating systems. It may be incomplete, list unavailable ports or may lack detailed descriptions of the ports.

Accessing ports

pySerial includes a small console based terminal program called serial.tools.miniterm. It can be started with `python -m serial.tools.miniterm <port_name>` (use option `-h` to get a listing of all options).