Twistless Documentation

Release 1.0.0

Taylor "Nekroze" Lawson

November 01, 2016

Contents

A simple to use bridge between stackless python (available in pypy) and the twisted networking library.

The underlying implementation is based off of the excelent article by Stephen Coursen found here http://www.stevecoursen.com/209/stackless-python-meets-twisted-matrix/ and was a huge help when writing *twistless*.

Twistless is designed to give a quick and easy start to using stackless python and twisted together. The following shows an overly brief example of a twisted reactor being started with stackless support.

```
from twistless import twistless
from twisted.internet import reactor


@twistless
def entry():
    reactor.run()


if __name__ == "__main__":
    entry()
```

Contents:

CHAPTER 1

Usage

The following is a simple example of using the `twistless.twistless` decorator to start a twisted reactor with stackless support.

```
"""
This example is based off of the echo server example from twisted matrix
documentation.

The original client will work fine and can be found here
http://twistedmatrix.com/documents/13.1.0/core/examples/simpleclient.py
"""
import time
from twistless import twistless, tasklet
from stackless import schedule
from twisted.internet import reactor, protocol


@tasklet
def async():
    """A deferred executed in another tasklet."""
    #Schedule this function to be continued at a later time.
    schedule()
    #Do something lengthy
    time.sleep(5)
    print("Tasklets!")


class Echo(protocol.Protocol):
    """This is just about the simplest possible protocol"""

    def dataReceived(self, data):
        """
        As soon as any data is received, write it back ASAP. But first setup a
        function to be called when there is time for it.
        """
        #call the async deferred function in another tasklet
        #The server will echo a response and then return to the tasklet
        #schedule which has the async method waiting to be returned to.
        async()
        self.transport.write(data)


@twistless
def main():
```

```
    """This runs the protocol on port 8000"""
    factory = protocol.ServerFactory()
    factory.protocol = Echo
    reactor.listenTCP(8000, factory)
    reactor.run()


# this only runs if the module was *not* imported
if __name__ == '__main__':
    main()
```

CHAPTER 2

Changelog

2.1 Version 1.0.0

- Currently the blocking decorator doesnt function correctly
- Causes segfaults when used with pypy

CHAPTER 3

Feedback

If you have any suggestions or questions about *Twistless* feel free to email me at nekroze@eturnilnetwork.com.

You can check out more of what I am doing at http://nekroze.eturnilnetwork.com my blog.

If you encounter any errors or problems with *Twistless*, please let me know! Open an Issue at the GitHub http://github.com/Nekroze/twistless main repository.