# trickle Documentation

***Release 0.1***

**A. Jesse Jiryu Davis**

December 16, 2013

# Contents

**Info** An IOStream wrapper for use with Tornado coroutines.

**Author** A. Jesse Jiryu Davis

CHAPTER 1

# Purpose

Tornado's IOStream API is central to Tornado, but it's designed to be used with callbacks, instead of with coroutines and Futures. Trickle is a proof-of-concept for a coroutine-friendly IOStream interface.

CHAPTER 2

# Dependencies

- Tornado >= version 3.1.
- YieldPoints.

CHAPTER 3

# Example

```
import re
import socket

from tornado import gen
from tornado.ioloop import IOLoop
from tornado.netutil import Resolver

from trickle import Trickle


resolver = Resolver()


@gen.coroutine
def download():
    sock = socket.socket(socket.AF_INET)
    trick = Trickle(sock)

    addr_info = yield resolver.resolve(
        'xkcd.com',
        80,
        socket.AF_INET)

    sock_addr = addr_info[0][1]

    yield trick.connect(sock_addr)
    yield trick.write(b'GET / HTTP/1.1\r\nHost: xkcd.com\r\n\r\n')

    headers = yield trick.read_until(b'\r\n\r\n')
    match = re.search(br'Content-Length: (\d+)\r\n', headers)
    content_length = int(match.group(1))

    body = yield trick.read_bytes(content_length)
    print body


IOLoop.current().run_sync(download)
```

CHAPTER 4

# API

**class** trickle.**Trickle**(**args*, ***kwargs*)
A coroutine-friendly IOStream interface.

Takes same parameters as IOStream, or takes a single IOStream as its only parameter.

**closed**()
Returns true if the stream has been closed.

**connect**(*address*, *server_hostname=None*, *timeout=None*)
Connects the socket to a remote address without blocking.

Like IOStream connect(), but returns a Future and takes no callback.

**read_bytes**(*num_bytes*, *timeout=None*)
Read the given number of bytes.

Like IOStream read_bytes(), but returns a Future and takes no callback or streaming_callback.

**read_until**(*delimiter*, *timeout=None*)
Read up to the given delimiter..

Like IOStream read_until(), but returns a Future and takes no callback.

**read_until_close**(**args*, ***kwargs*)
Read all remaining data from the socket.

Like IOStream read_until_close(), but returns a Future and takes no callback or streaming_callback.

**read_until_regex**(*regex*, *timeout=None*)
Read up to the given regex pattern.

Like IOStream read_until_regex(), but returns a Future and takes no callback.

**write**(*data*, *timeout=None*, ***kwargs*)
Write the given data to this stream.

Like IOStream write(), but returns a Future and takes no callback.

yield the returned Future to wait for all data to be written to the stream.

CHAPTER 5

# Testing

Run `python setup.py nosetests` in the root directory.