
mprpc Documentation

Release 0.1.14

Studio Ousia

Apr 11, 2017

Contents

1	Introduction	3
1.1	Installation	3
1.2	Examples	3
2	Performance	5
2.1	Results	5
2.2	Environment	6
3	API Reference	7
4	Indices and tables	9

mprpc is a lightweight [MessagePack RPC](#) library. It enables you to easily build a distributed server-side system by writing a small amount of code. It is built on top of [gevent](#) and [MessagePack](#).

Contents:

Installation

To install mprpc, simply:

```
$ pip install mprpc
```

Alternatively,

```
$ easy_install mprpc
```

Examples

RPC server

```
from gevent.server import StreamServer
from mprpc import RPCServer

class SumServer(RPCServer):
    def sum(self, x, y):
        return x + y

server = StreamServer(('127.0.0.1', 6000), SumServer())
server.serve_forever()
```

RPC client

```
from mprpc import RPCClient
```

```
client = RPCClient('127.0.0.1', 6000)
print client.call('sum', 1, 2)
```

RPC client with connection pooling using gsocketpool

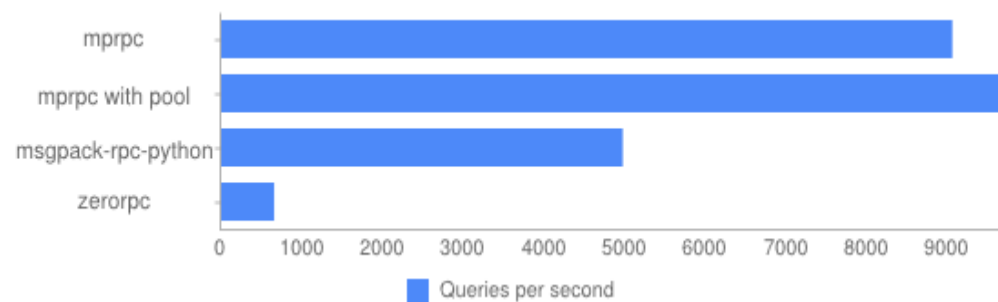
```
import gsocketpool.pool
from mprpc import RPCPoolClient

client_pool = gsocketpool.pool.Pool(RPCPoolClient, dict(host='127.0.0.1', port=6000))

with client_pool.connection() as client:
    print client.call('sum', 1, 2)
```


mprpc significantly outperforms the official MessagePack RPC (**1.8x** faster), which is built using Facebook's Tornado and MessagePack, and ZeroRPC (**14x** faster), which is built using ZeroMQ and MessagePack.

Results



mprpc

```
% python benchmarks/benchmark.py  
call: 9508 qps  
call_using_connection_pool: 10172 qps
```

Official MessagePack RPC

```
% pip install msgpack-rpc-python  
% python benchmarks/benchmark_msgpackrpc_official.py  
call: 4976 qps
```

ZeroRPC

```
% pip install zerorpc
% python benchmarks/benchmark_zerorpc.py
call: 655 qps
```

Environment

- OS: Mac OS X 10.8.5
- CPU: Intel Core i7 2GHz
- Memory: 8GB
- Python: 2.7.3

class `mprpc.RPCClient`
RPC client.

Usage:

```
>>> from mprpc import RPCClient
>>> client = RPCClient('127.0.0.1', 6000)
>>> print client.call('sum', 1, 2)
3
```

Parameters

- **host** (*str*) – Hostname.
- **port** (*int*) – Port number.
- **timeout** (*int*) – (optional) Socket timeout.
- **lazy** (*bool*) – (optional) If set to True, the socket connection is not established until you specifically call `open()`
- **pack_encoding** (*str*) – (optional) Character encoding used to pack data using Messagepack.
- **unpack_encoding** (*str*) – (optional) Character encoding used to unpack data using Messagepack.
- **pack_params** (*dict*) – (optional) Parameters to pass to Messagepack Packer
- **unpack_params** (*dict*) – (optional) Parameters to pass to Messagepack

:param `tcp_no_delay` (optional) If set to True, use `TCP_NODELAY`. :param `keep_alive` (optional) If set to True, use socket `keep alive`.

Unpacker

call ()

Calls a RPC method.

Parameters

- **method** (*str*) – Method name.
- **args** – Method arguments.

close()

Closes the connection.

getpeername()

Return the address of the remote endpoint.

is_connected()

Returns whether the connection has already been established.

Return type bool**open()**

Opens a connection.

class `mprpc.RPCServer`

RPC server.

This class is assumed to be used with `gevent StreamServer`.**Parameters**

- **pack_encoding** (*str*) – (optional) Character encoding used to pack data using Messagepack.
- **unpack_encoding** (*str*) – (optional) Character encoding used to unpack data using Messagepack
- **pack_params** (*dict*) – (optional) Parameters to pass to Messagepack Packer
- **unpack_params** (*dict*) – (optional) Parameters to pass to Messagepack Unpacker

Usage:

```
>>> from gevent.server import StreamServer
>>> import mprpc
>>>
>>> class SumServer(mprpc.RPCServer):
...     def sum(self, x, y):
...         return x + y
...
>>>
>>> server = StreamServer(('127.0.0.1', 6000), SumServer())
>>> server.serve_forever()
```

CHAPTER 4

Indices and tables

- `genindex`
- `modindex`
- `search`

C

call() (mprpc.RPCClient method), 7
close() (mprpc.RPCClient method), 8

G

getpeername() (mprpc.RPCClient method), 8

I

is_connected() (mprpc.RPCClient method), 8

O

open() (mprpc.RPCClient method), 8

R

RPCClient (class in mprpc), 7
RPCServer (class in mprpc), 8