

---

# **osc Documentation**

***Release 0.1***

**Oleksii Lytvyn**

October 02, 2015



<b>1</b>	<b>Features</b>	<b>3</b>
1.1	Table of contents . . . . .	3



Simple OSC implementation in pure Python.

This library was developped following the specifications at [http://opensoundcontrol.org/spec-1\\_0](http://opensoundcontrol.org/spec-1_0) and is currently in a stable state.



---

## Features

---

- UDP client and server
- int, float, string, blob, true, false OSC arguments support
- blocking/threading/forking server implementations
- simple API

## 1.1 Table of contents

### 1.1.1 Classes

OSCPacket

OSCMessage

OSCBundle

OSCServer

OSCClient

### 1.1.2 Examples

Simple Server and Client

#### Server

This server will print all incoming messages and bundles

```
import argparse
import math

from osc import OSCServer

class SimpleServer(OSCServer):
```

```
def handle(self, address, message, time):
    if message.is_bundle():
        for msg in message:
            print(time, address, msg.address, msg.args)
    else:
        print(time, address, message.address, message.args)

if __name__ == "__main__":
    parser = argparse.ArgumentParser()
    parser.add_argument("--ip",
                        default="127.0.0.1", help="The ip to listen on")
    parser.add_argument("--port",
                        type=int, default=8000, help="The port to listen on")
    args = parser.parse_args()

    server = SimpleServer(args.ip, args.port)

    print("Serving on {}".format(server.server_address))
    server.serve_forever()
```

## Client

Client will send OSCBundle and OSCMessages

```
import argparse
import random
import time

from osc import OSCMessage, OSCClient, OSCBundle

if __name__ == "__main__":
    parser = argparse.ArgumentParser()
    parser.add_argument("--ip", default="127.0.0.1",
                        help="The ip of the OSC server")
    parser.add_argument("--port", type=int, default=8000,
                        help="The port the OSC server is listening on")
    args = parser.parse_args()

    client = OSCClient(args.ip, args.port)

    for value in ["Lorem ipsum dolor sit amet", 123, True, bytes("utf-8 text", "utf-8")]:
        msg = OSCMessage(address = "/debug")
        msg.add( value )

        client.send(msg)

    msg1 = OSCMessage(address = "/bundle/a")
    msg1.add( 123 )

    msg2 = OSCMessage(address = "/bundle/b")
    msg2.add( False )

    bun = OSCBundle()
    bun.add( msg1 )
    bun.add( msg2 )
```



```
client.send( bun )

for x in range(10):
    msg = OSCMessage(address = "/filter")
    msg.add(random.random())

    client.send(msg)
    time.sleep(1)
```