OSC

Andrea Cuius

andrea@nocte.co.uk
www.nocte.co.uk
@q_says / @noctestudio

nocte

Open Sound Control (OSC) is a protocol for communication among computers, sound synthesizers, and other multimedia devices that is optimized for modern networking technology.

http://opensoundcontrol.org/

http://en.wikipedia.org/wiki/Open_Sound_Control

OSC support:

Cinder, OpenFrameworks, Ableton Live, VDMX, VVVV, Processing, Resolume, Modul8, Quartz Composer, MAX/MSP, Pure Data, Touch OSS, Super collider, etc..

Messages and bundles

OSC can send and receive single messages or bundles.

Each OSC message includes the address and zero or more arguments(also the arguments types).

A bundle is a more efficient collection of messages.

Data format

One of the main aspects of OSC is its simplicity, messages can be identified by an address and carry different type of data.

/test_address 1 hello 5.0

address:	“test_address”
3 arguments:	1, “hello”, 5.0
argument types(tags):	int, string, float

Types

- **int32**: 32-bit big-endian two's complement integer
- **OSC-timetag**: 64-bit big-endian fixed-point time tag
- **float32**: 32-bit big-endian IEEE 754 floating point number
- **OSC-string**: A sequence of non-null ASCII characters
- **OSC-blob**: An int32 size count, followed by that many 8-bit bytes of arbitrary binary data

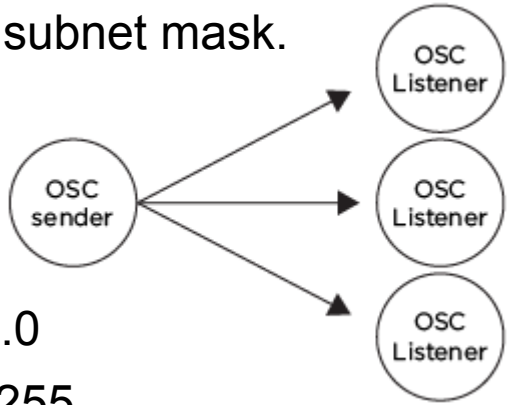
OSC Sender


```
#include "OscSender.h"

void MyApp::setup() {
    // setup the sender with the receiver IP and port number
    oscSender.setup( “192.168.1.100”, 8000 );
    osc::Message message;                       // create a new message
    message.setAddress("/cinder/osc/1");        // set message address
    message.addFloatArg( 1.0f );                // add float argument
    oscSender.sendMessage(message);             // send the message
}
```

OSC Sender broadcast

The OSC sender can be setup for message broadcasting or multicasting. The broadcast IP address can be calculated using the machine IP address and the subnet mask, the broadcast address is the bitwise OR operation between the machine IP and the bit complement of the subnet mask.



http://en.wikipedia.org/wiki/Broadcast_address

ie: IP = 192.168.1.100 and subnet mask = 255.255.255.0

broadcast_IP = 192.168.1.100 | 0.0.0.255 = 192.168.1.255

sender.setup(broadcast_IP, 8000, **true**);

OSC Listener

```
#include "OscListener.h"
…

void MyApp::setup()
{
	int port = 8000;
	oscListener.setup( port );
}
```

OSC Listener

```
void MyApp::update() {
	while( oscListener.hasWaitingMessages() ) {
		osc::Message message;
		oscListener.getNextMessage( &message );

		for (int i = 0; i < message.getNumArgs(); i++) {
			// parse message content
		 }
	}
}
```

OSC and UDP

The OSC Cinder block is based on UPD

- UDP provides checksums for data integrity
- Little overhead
- There is **no guarantee** of delivery!

Andrea Cuius

andrea@nocte.co.uk
www.nocte.co.uk
@q_says / @noctestudio

nocte