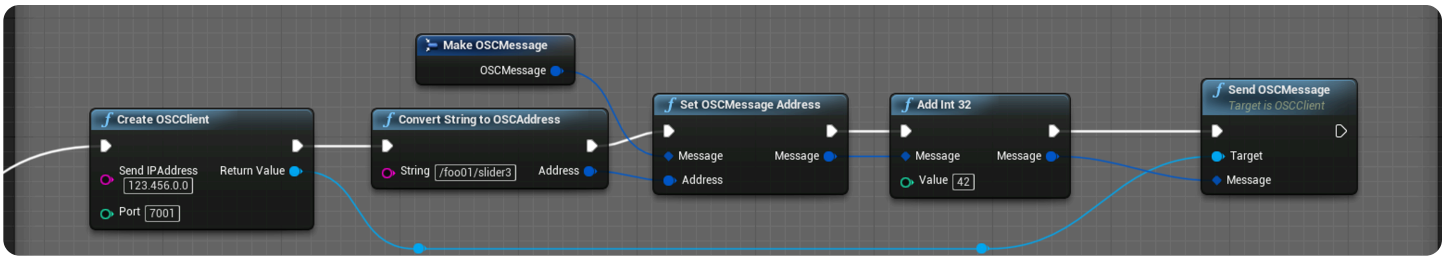
OSC Plugin Overview

An overview of how the Open Sound Control (OSC) plugin works within Unreal Engine.



Open Sound Control (**OSC**) is an open protocol commonly used in many areas of the audio industry. It is primarily used to network generic audio data between clients, although it can also be used for non-audio data.

The OSC plugin provides an intuitive, type-safe Blueprint library that a developer can use to quickly iterate networked audio (and potentially other domain) data in **Unreal Engine**. When this plugin is enabled, you can send and receive OSC events through a simple API in either **C++** or **Blueprint**. It supports sending and receiving messages and bundles, or combinations of bundles and messages.

> The **OSC** plugin must be enabled to use this feature. Go to **Edit > Plugins > Input Devices**, then check the **OSC (Open Sound Control)** option to enable.

Root Types

OSC Server

The **OSC Server** (`UOSCServer`) acts as a listening endpoint for messages sent to the local instance of Unreal Engine. It supports multicast loopback, and parsing of both OSC messages and bundles. It also lets the server specifically state which IP addresses it will listen to (allowlisting).

A user can bind an event to receive messages from all OSC addresses using a provided address pattern (see *Binding An Event to Address Patterns* below), listen for all messages (see *Binding An Event to All Messages* below), or listen for all bundles (similar to *Binding An Event to All Messages* below but requiring unpacking a received bundle).

OSC Client

The **OSC Client** (`UOSCClient`) provides a way of sending OSC messages and bundles.

FOSCAddress

An **OSC Address** (`FOSCAddress`) is a typed path that may or may not also be a valid pattern. As a valid path, it can be created, sent, and received as part of a message packet.

It contains an array of containers and a method in the form of `/Container1/Container2/Method`, and can be queried for validity as either a path or a pattern, converted to or from a string, and manipulated with Blueprint calls. It can also be filtered against another OSC Address that is a pattern when binding to the event `BindEventToOnOSCAddressPatternMatchesPath`.

OSC Bundle/Message

Both **OSC Message** (`FOSCMessage`) and **OSC Bundle** (`FOSCBundle`) are packet types that are received from the OSC Client, or sent with the OSC Server.

An OSC Message contains an address that is also a valid path, and that has a payload of supported OSC protocol types.

An OSC Bundle contains an array of packets of either other OSC Bundles or OSC Messages, or both.

Asset Classes

The OSC plugin does not require any asset classes. All of its required types are transient and expected to be created, managed, and destroyed with Blueprint.

Blueprint API

OSC Server

OnOsc(Bundle/Message)Received	An event that gets called when an OSC bundle/message is received.
SetAllowlistClientsEnabled	When set to true, the server will only process received messages from allowlisted clients.
(UnbindEvent/UnbindAllEvents/BindEventTo/Get)OnOSCAddressPattern(s)	Unbinds, binds, removes or gets event(s) to dispatch when an OSCAddressPattern is matched.
(Add/Remove/Clear/Get)AllowlistedClient(s)	Adds, removes, clears, or gets a set of client IP Addresses as strings (IPv4) that are allowlisted.

OSC Client

(Get/Set)SendIPAddress	Gets the OSC Client IP address and port.
SendOSC(Bundle/Message)	Sends an OSC bundle/message to a specific address.

OSC Bundle/Message

Add(Bundle/Message)ToBundle	Adds provided bundle/message to bundle.
Get(Bundles/Messages)FromBundle	Returns the bundles/messages found in the bundle.
(Add/Get) OSC Message (Strings/String at Index, Integer/Integer at Index, and so on)	Adds/gets POD type to or from OSCMessage. (List of types supported can be found here.

Clear OSC (Message/Bundle)	Clears the message/bundle's payload.

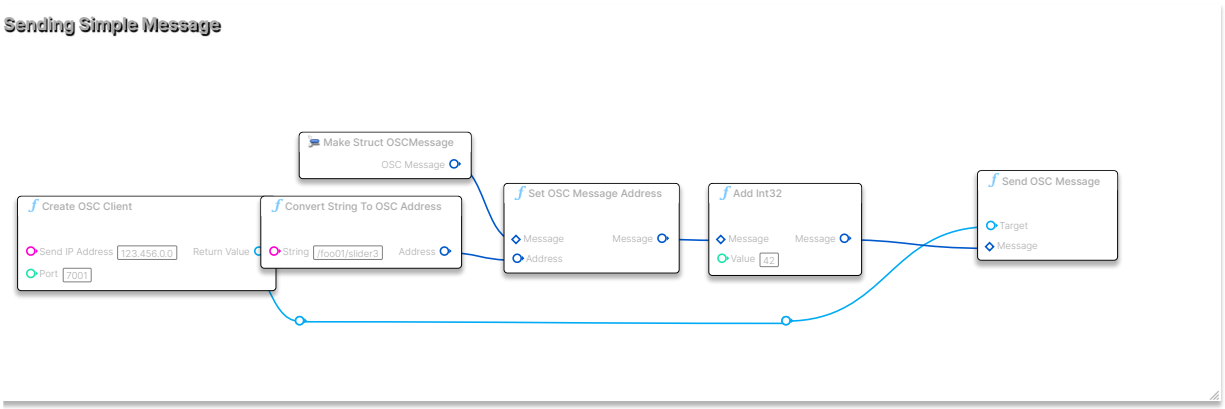
Resources/Examples

OSC implementation is based on the OSC 1.0 protocol found on the [OpenSoundControl website](#).

Click the **Copy Node Graph** button on each image below and paste the text into your Blueprint Graph to see the examples in action.

Sending an OSC Message

Fullscreen Reset Graph Zoom -5



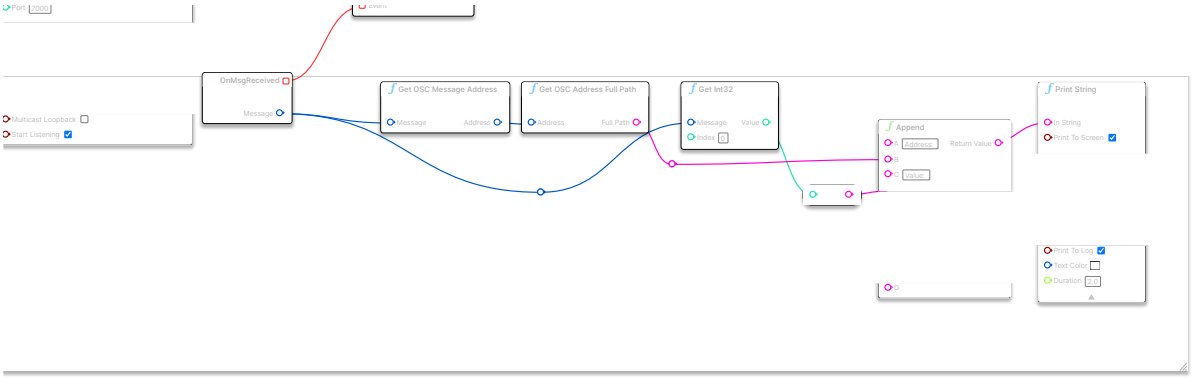
Renderer by Rancoud

BLUEPRINT

Copy code

Binding an Event to All Messages

Fullscreen Reset Graph Zoom -7



Renderer by Rancoud

Copy code

Binding an Event to Address Patterns

Use ctrl + scroll to zoom

Copy code

Blueprint API

Get OSC Message From Bundle At Index	Returns the message found in a bundle at ordered index.
Add OSC Address (As String) to OSC Message	Adds an address (packed as a string) value to the end of OSCMessage
Get OSC Message Address At Index	Sets the Value to the address at the provided Index in OSCMessage if in bounds, and if OSC type matches `String`. (It does *not* return the address of the message; instead, the string is packed in the message and casts to the OSC address.) Returns if the string is found at Index, and if it is a valid OSC address path.
Get OSC Message Addresses	Returns all strings that are valid address paths, in the order received from OSCMessage. (It does *not* include the address of the message, just the strings packed in the message that are valid address paths.)
Find Object at OSC Address	Finds an object with the given OSC Address in path form, where containers

Console Variables

`osc.clients`	Prints diagnostic information for the currently initialized OSC client objects to the output log.

`osc.client.connect`	Connects (or reconnects) the OSC mix client with the provided object name. See `osc.clients` for a list of available clients and their respective names.
	`Id` — Object ID of client to (re)connect to
	`Address` — IP Address to (re)connect to (default: LocalHost)
	`Port` — Port to (re)connect to (default: 8094)
`osc.client.connectById`	Connects (or reconnects) the OSC mix client with the provided Object ID. See `osc.clients` for a list of available clients and their respective IDs.
	`Id` — The Object ID of client to (re)connect to.
	`Address` — IP Address to (re)connect to (default: LocalHost)
	`Port` — Port to (re)connect to (default: 8094)
`osc.servers`	Prints diagnostic information for the currently initialized OSC server Objects to the output log.
`osc.server.connect`	Connects (or reconnects) the OSC mix client with the provided Object name. See `osc.clients` for a list of available clients and their respective names.
	`Name` — Object name of server to (re)connect to
	`Address` — IP Address to (re)connect to (default: LocalHost)
	`Port` — Port to (re)connect to (default: 8095)
`osc.server.connectById`	Connects (or reconnects) the OSC mix client with the provided Object ID. See `osc.clients` for a list of available clients and their respective IDs.
	`Id` — Object ID of client to (re)connect to
	`Address` — IP Address to (re)connect to (default: LocalHost)
	`Port` — Port to (re)connect to (default: 8095)