OSC simpl Reference

Version 1.2

MonoBehaviours

- OscIn
- OscOut

Classes

- OscBundle
- OscImpulse
- OscMessage
- OscTimeTag

Enums

- OscReceiveMode
- OscRemoteStatus
- OscSendMode

OscIn

MonoBehaviour for receiving OscMessage objects.

int port
Gets the local port that this application is set to listen to. (read only). To set, call the Open method.

OscReceiveMode mode
Gets the transmission mode (read only). Can either be UnicastBroadcast or Multicast. The mode is automatically derived from arguments passed to the Open method.

string multicastAddress
Gets the remote address to the multicast group that this application is set to listen to (read only). To set, call the Open method and provide a valid multicast address.

static string ipAddress
Gets the local network IP address for this device (read only). Returns an empty string if the address is not available. In that case ensure that your device is connected to a network. Using a VPN may block you from getting the local IP.

bool isOpen
Indicates whether the Open method has been called and the object is ready to receive.

bool filterDuplicates
When enabled, only one message per OSC address will be forwarded every Update call. The last (newest) message received will be used. Default is true.

bool addTimeTagsToBundledMessages
When enabled, timetags from bundles are added to contained messages as last argument. Incoming bundles are never exposed, so if you want to access a time tag from a incoming bundle then enable this. Default is false.

int messageCount
Gets the number of messages received since last update.

OscMessageEvent onAnyMessage
Add listener to this event to receive a call when any OscMessage is received on the specified port. The event is influenced by the state of 'filterDuplicates' property.

bool Open(int port, string multicastAddress = "")
Open to receive messages on specified port and (optionally) from specified multicast IP address. Returns success status.

void Close()
Close and stop receiving messages.

void Map(string address, UnityAction<OscMessage> method)
Request that incoming messages with OSC 'address' are forwarded to 'method'.

void MapFloat(string address, UnityAction<float> method)
Request that float type argument is extracted from incoming messages with OSC 'address' and forwarded to 'method'.

void MapDouble(string address, UnityAction<double> method)
Same as above, for double type.

void MapInt(string address, UnityAction<int> method)
Same as above, for int type.

void MapLong(string address, UnityAction<long> method)
Same as above, for long type.

void MapString(string address, UnityAction<string> method)
Same as above, for string type.

void MapChar(string address, UnityAction<char> method)
Same as above, for char type.

void MapBool(string address, UnityAction<bool> method)
Same as above, for bool type.

void MapColor(string address, UnityAction<Color32> method)
Same as above, for color type.

void MapBlob(string address, UnityAction<byte[]> method)

Same as above, for blob type.

void MapTimeTag(string address, UnityAction<OscTimeTag> method)
Same as above, for time tag type.

void MapImpulseNullOrEmpty(string address, UnityAction method)
Request that 'method' is invoked when a message with OSC 'address' is received with type tag Impulse (i), Null (N) or simply without arguments.

void Unmap(UnityAction<OscMessage> method)
Request that 'method' is no longer invoked. Note that only mappings made at runtime can be unmapped.

void UnmapFloat(UnityAction<float> method)
Same as above for float type.

void UnmapDouble(UnityAction<double> method)
Same as above for double type.

void UnmapInt(UnityAction<int> method)
Same as above for int type.

void UnmapLong(UnityAction<long> method)
Same as above for long type.

void UnmapString(UnityAction<string> method)
Same as above for string type.

void UnmapChar(UnityAction<char> method)
Same as above for char type.

void UnmapBool(UnityAction<bool> method)
Same as above for bool type.

void UnmapColor(UnityAction<Color32> method)
Same as above for color type.

void UnmapBlob(UnityAction<byte[]> method)
Same as above for blob type.

void UnmapTimeTag(UnityAction<OscTimeTag> method)
Same as above for time tag type.

void UnmapImpulseNullOrEmpty(UnityAction method)
Same as above for methods with no arguments.

void UnmapAll(string address)
Request that all methods that are mapped to OSC 'address' will no longer be invoked. This is useful for unmapping delegates.

OscOut

MonoBehaviour for sending OscMessage and OscBundle objects.

int port
Gets the port to be send to on the target remote device (read only). To set, call the Open method.

OscSendMode mode
Gets the transmission mode (read only). Can either be UnicastToSelf, Unicast, Broadcast or Multicast. The mode is automatically derived from the IP address passed to the Open method.

string ipAddress
Gets the IP address of the target remote device (read only). To set, call the 'Open' method.

bool isOpen
Indicates whether the Open method has been called and the object is ready to send.

OscRemoteStatus remoteStatus
Gets the remote connection status (read only). Can either be Connected, Disconnected or Unknown.

int messageCount
Gets the number of messages send since last update.

bool multicastLoopback
Indicates whether outgoing multicast messages are also delivered to the sending application. Default is true.

bool bundleMessagesOnEndOfFrame
When enabled, messages will automatically be buffered in a single OscBundle and send at the end of the frame (i.e. Unity's WaitForEndOfFrame). Default is false.

bool Open(int port, string ipAddress = "")
Open to send messages to specified port and (optional) IP address. If no IP address is given, messages will be send locally on this device. Returns success status.

void Close()
Close and stop sending messages.

bool Send(string address, params object[] args)
Send a OscMessage with specified address and arguments. Returns success status.

bool Send(OscPacket packet)
Send an OscMessage or OscBundle. Returns success status.

OscBundle

Class representing an OSC bundle. Bundles have a OscTimeTag and can contain OscMessage and OscBundle objects.

OscTimeTag timeTag
Gets or sets the timetag for this bundle.

List<OscPacket> packets
Gets the list of OscMessage and OscBundle objects.

OscBundle()
Constructor for creating a bundle with a timetag containing the current time.

OscBundle(OscTimeTag timeTag)
Constructor for creating a bundle with specified timetag.

void Add(OscPacket packet)
Add a OscMessage or OscBundle to this bundle. Shorthand for bundle.packets.Add.

void Clear()
Remove all OscMessage and OscBundle object in this bundle, and do so recursively for all contained bundles.

OscImpulse

Class representing the OSC 1.1 argument type Impulse. In OSC 1.0 this was called 'Infinitum'.

OscMessage

Class representing an OSC message. Messages have an OSC address an a number of OSC arguments.

Supported argument types and their Unity translations:

- float (f) <-> float
- integer (i) <-> int
- string (s), symbol (S) <-> string
- blob (b) <-> byte[]
- long (h) <-> long
- double (d) <-> double
- boolean (T,F) <-> bool
- character (c) <-> char
- color (r) <-> Color32, Color
- timetag (t) <-> OscTimeTag, DateTime
- impulse (I) <-> OscImpulse
- nil (N)<-> null

string address
Gets or sets the address of the message. Must start with '/'.

List<object> args
Gets or sets the OSC arguments. Manipulate the list directly.

OscMessage(string address, params object[] args)
Constructor taking an address and an arbitrary number of OSC type arguments.

void Add(params object[] args)
Add one or more OSC type arguments. Shorthand for message.args.Add.

void Clear()
Clear all arguments. Shorthand for message.args.Clear.

bool TryGet(int index, out float value)
Tries to get argument at index of type float. Returns success status.

bool TryGet(int index, out double value)
Same as above, double type.

bool TryGet(int index, out int value)
Same as above, int type.

bool TryGet(int index, out long value)
Same as above, long type.

bool TryGet(int index, out string value)
Same as above, string type.

bool TryGet(int index, out char value)
Same as above, char type.

bool TryGet(int index, out bool value)
Same as above, bool type.

bool TryGet(int index, out Color32 value)
Same as above, color type.

bool TryGet(int index, out byte[] value)
Same as above, blob type.

bool TryGet(int index, out OscTimeTag value)
Same as above, osc timetag type.

bool TryGetNull(int index)
Same as above, null type.

bool TryGetImpulse(int index)
Same as above, impulse type.

OscTimeTag

Class representing a OSC timetag. OscTimeTag objects are send implicitly with bundles and explicitly as message arguments. Incoming bundles are never exposed to the user. Instead, the contained messages are unwrapped automatically and send to mapped methods. If you want to receive timetags from bundles, then enable 'addTimeTagsToBundledMessages' on OscIn and grab the timetag from the last argument of your incoming bundled message. Timed scheduling of received bundled messages is not supported.

DateTime time
Gets or sets the time with DateTime tick precision.

bool immediately
Gets or sets the 'immediately' flag. Some OSC implementations may interpret the flag as "process immediately on receipt" - as opposed to "schedule the recieved bundle for processing at time" - while other implementations may ignore it. Default is true.

ulong oscNtp
Get or sets the OSC flavoured NTP encoded value in which a time and a 'immediately' flag is stored. Don't manipulate this property directly, unless you have read the OSC 1.0 specification.

OscTimeTag()
Create new timetag with time set to now.

OscTimeTag(DateTime time)
Create new timetag with 'time'.

OscTimeTag(DateTime time, bool immediately)
Create new timetag with 'time' and 'immediately' flag.

OscReceiveMode

Enum representing the mode of transmission for OscIn. Can either be UnicastBroadcast or UnicastBroadcastMulticast.

OscRemoteStatus

Enum representing the connection status to a remote device. Can be either Connected, Disconnected or Unknown. For broadcast and multicast mode the status will always be Unknown.

OscSendMode

Enum representing the mode of transmission for OscOut. Can either be UnicastToSelf, Unicast, Broadcast or Multicast.