OSC simpl Reference

Version 2.3.1

MonoBehaviours

- OscGlobals
- OscIn
- OscOut

Classes

- OscBundle
- OscMessage
- OscPool

OscGlobals

Singleton Monobehaviour containing global settings and stats for OSC simpl. Add this script to the scene and manipualte using the inspector or access statically fromt other scripts. Use is optional.

static bool logStatuses
When true all status messages will be logged in the console.

static bool logWarnings
When true all warning messages will be logged in the console.

OscIn

MonoBehaviour for receiving OscMessage objects. The "OSC Server".

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 localIpAddress
Gets the primary local network IP address for this device (read only). If the the loopback address "127.0.0.1" is returned ensure that your device is connected to a network. Using a VPN may block you from getting the local IP.

static ReadOnlyCollection<string> localIpAddressAlternatives
Gets the alternative local network IP addresses for this device (read only). Your device may be connected through multiple network adapters, for example through wifi and ethernet.

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 udpBufferSize
Gets or sets the size of the UDP buffer.

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

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, Action<OscMessage> method)
Request that incoming messages with 'address' are forwarded to 'method'.

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

void MapDouble(string address, Action<double> method)
Request that a double type argument is extracted from incoming messages with matching 'address' and forwarded to 'method'.

void MapInt(string address, Action<int> method)
Request that a int type argument is extracted from incoming messages with matching 'address' and forwarded to 'method'.

void MapLong(string address, Action<long> method)
Request that a long type argument is extracted from incoming messages with matching 'address' and

forwarded to 'method'.

void MapString(string address, Action<string> method)
Request that a string type argument is extracted from incoming messages with matching 'address' and forwarded to 'method'. This method produces heap garbage. Use Map(string, UnityAction) instead and then use TryGet(int, ref string) to read into a cached string. See how in the Optimisations example.

void MapChar(string address, Action<char> method)
Request that a char type argument is extracted from incoming messages with matching 'address' and forwarded to 'method'.

void MapBool(string address, Action<bool> method)
Request that a bool type argument is extracted from incoming messages with matching 'address' and forwarded to 'method'.

void MapColor(string address, Action<Color32> method)
Request that a Color32 type argument is extracted from incoming messages with matching 'address' and forwarded to 'method'.

void MapBlob(string address, Action<byte[]> method)
Request that a byte blob argument is extracted from incoming messages with matching 'address' and forwarded to 'method'. This method produces heap garbage. Use Map(string, UnityAction) instead and then use TryGet(int, ref byte[]) to read into a cached array. See how in the Optimisations example.

void MapTimeTag(string address, Action<OscTimeTag> method)
Request that a time tag argument is extracted from incoming messages with matching 'address' and forwarded to 'method'.

void MapMidi(string address, Action<OscMidiMessage> method)
Request that a OscMidiMessage type argument is extracted from incoming messages with matching 'address' and forwarded to 'method'.

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

void Unmap(Action<OscMessage> method)
Request that 'method' is no longer invoked by OscIn.

void UnmapFloat(Action<float> method)
Request that 'method' is no longer invoked by OscIn.

void UnmapInt(Action<int> method)
Request that 'method' is no longer invoked by OscIn.

void UnmapString(Action<string> method)
Request that 'method' is no longer invoked by OscIn.

void UnmapBool(Action<bool> method)
Request that 'method' is no longer invoked by OscIn.

void UnmapColor(Action<Color32> method)
Request that 'method' is no longer invoked by OscIn.

void UnmapChar(Action<char> method)
Request that 'method' is no longer invoked by OscIn.

void UnmapDouble(Action<double> method)
Request that 'method' is no longer invoked by OscIn.

void UnmapLong(Action<long> method)
Request that 'method' is no longer invoked by OscIn.

void UnmapTimeTag(Action<OscTimeTag> method)
Request that 'method' is no longer invoked by OscIn.

void UnmapMidi(Action<OscMidiMessage> method)
Request that 'method' is no longer invoked by OscIn.

void UnmapBlob(Action<byte[]> method)
Request that 'method' is no longer invoked by OscIn.

void UnmapImpulseNullOrEmpty(UnityAction method)

Request that 'method' is no longer invoked by OscIn.

void UnmapAll(string address)
Request that all methods that are mapped to OSC 'address' will no longer be invoked.

void MapAnyMessage(Action<OscMessage> method)
Subscribe to all outgoing messages. The state of 'filterDuplicates' does apply.

void UnmapAnyMessage(Action<OscMessage> method)
Unsubscribe to all outgoing messages.

OscOut

MonoBehaviour for sending OscPackets. The "OSC Client".

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 remoteIpAddress
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.

int udpBufferSize
Gets or sets the size of the UDP buffer.

bool bundleMessagesAutomatically
Set to false ONLY if the receiving end does not support OSC bundles. Without bundling, messages that are send successively are prone to be lost. By default, messages are wrapped in a bundle (or multiple bundles if the buffer size is exceeded) that is send by the end of the Unity frame.

bool Open(int port, string remoteIpAddress = "")
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(OscPacket packet)
Send an OscMessage or OscBundle. Data is serialized and no reference is stored, so you can safely change values and send packet immediately again. Returns success status.

void Send(string address, float value)
Send an OscMessage with a single argument.

void Send(string address, double value)
Send an OscMessage with a single argument.

void Send(string address, int value)
Send an OscMessage with a single argument.

void Send(string address, long value)
Send an OscMessage with a single argument.

void Send(string address, string value)
Send an OscMessage with a single argument.

void Send(string address, char value)
Send an OscMessage with a single argument.

void Send(string address, bool value)
Send an OscMessage with a single argument.

void Send(string address, Color32 value)
Send an OscMessage with a single argument.

void Send(string address, byte[] value)
Send an OscMessage with a single argument.

void Send(string address, OscTimeTag value)
Send an OscMessage with a single argument.

void Send(string address, OscMidiMessage value)
Send an OscMessage with a single argument.

void Send(string address, OscNull value)
Send an OscMessage with a single argument.

void Send(string address, OscImpulse value)
Send an OscMessage with a single argument.

void Send(string address)
Send an OscMessage with no arguments.

void MapAnyMessage(Action<OscMessage> method)
Subscribe to all outgoing messages.

void UnmapAnyMessage(Action<OscMessage> method)
Unsubscribe to all outgoing messages.

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.

override int Size()
Returns the byte size of the bundle.

OscMessage

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

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

OscMessage()
Creates a new OSC Message.

OscMessage(string address)
Creates a new OSC Message with address.

void Clear()
Clears all arguments.

int Count()
Returns the number of arguments.

void RemoveAt(int index)
Removes argument at index, shifting the subsequent arguments one index down.

bool TryGetArgType(int index, out OscArgType type)
Returns the argument type at index.

bool TryGetArgTag(int index, out char tag)
Tries to get argument tag at index. Returns success status.

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)
Tries to get argument at index of type double. Returns success status.

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

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

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

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

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

bool TryGet(int index, out Color32 value)
Tries to get argument at index of type color. Returns success status.

bool TryGet(int index, ref byte[] value)
Tries to get argument at index of type blob (byte[]). Returns success status.

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

bool TryGet(int index, out OscMidiMessage value)
Tries to get argument at index of midi message. Returns success status.

bool TryGet(int index, OscNull nothing)
Tries to get argument at index of type null. Returns success status.

bool TryGet(int index, OscImpulse impulse)
Tries to get argument at index of type impulse. Returns success status.

OscMessage Set(int index, float value)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Set(int index, double value)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Set(int index, int value)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Set(int index, long value)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Set(int index, string value)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Set(int index, char value)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Set(int index, bool value)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Set(int index, Color32 value)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Set(int index, byte[] value)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Set(int index, OscTimeTag value)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Set(int index, OscMidiMessage value)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Set(int index, OscNull nothing)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Set(int index, OscImpulse impulse)
Set argument at specified index, expanding message capacity if necessary.

OscMessage Add(float value)
Adds argument.

OscMessage Add(double value)
Adds argument.

OscMessage Add(int value)
Adds argument.

OscMessage Add(long value)
Adds argument.

OscMessage Add(string value)
Adds argument.

OscMessage Add(char value)
Adds argument.

OscMessage Add(bool value)
Adds argument.

OscMessage Add(Color32 value)
Adds argument.

OscMessage Add(byte[] value)
Adds argument.

OscMessage Add(OscTimeTag value)
Adds argument.

OscMessage Add(OscMidiMessage value)
Adds argument.

OscMessage Add(OscNull nothing)
Adds argument.

OscMessage Add(OscImpulse impulse)
Adds argument.

bool TryGetBlob(int index, out Vector2 value)
Tries to get value at index of from a byte blob. Returns success status.

bool TryGetBlob(int index, out Vector2Int value)
Tries to get value at index of from a byte blob. Returns success status.

bool TryGetBlob(int index, out Vector3 value)
Tries to get value at index of from a byte blob. Returns success status.

bool TryGetBlob(int index, out Vector3Int value)
Tries to get value at index of from a byte blob. Returns success status.

bool TryGetBlob(int index, out Vector4 value)
Tries to get value at index of from a byte blob. Returns success status.

bool TryGetBlob(int index, out Quaternion value)
Tries to get value at index of from a byte blob. Returns success status.

bool TryGetBlob(int index, out Rect value)
Tries to get value at index of from a byte blob. Returns success status.

bool TryGetBlob(int index, out Matrix4x4 value)
Tries to get value at index of from a byte blob. Returns success status.

bool TryGetBlob(int index, ref List<float> values)
Tries to read a list of values from a byte blob at argument index. Returns success status.

bool TryGetBlob(int index, ref List<int> values)
Tries to read a list of values from a byte blob at argument index. Returns success status.

bool TryGetBlob(int index, Encoding encoding, out string value)
Tries to read string with a given encoding from a byte blob at argument index. Returns success status.

OscMessage SetBlob(int index, Vector2 value)
Set a value as a byte blob at specified index, expanding message capacity if necessary.

OscMessage SetBlob(int index, Vector2Int value)
Set a value as a byte blob at specified index, expanding message capacity if necessary.

OscMessage SetBlob(int index, Vector3 value)
Set a value as a byte blob at specified index, expanding message capacity if necessary.

OscMessage SetBlob(int index, Vector3Int value)
Set a value as a byte blob at specified index, expanding message capacity if necessary.

OscMessage SetBlob(int index, Vector4 value)
Set a value as a byte blob at specified index, expanding message capacity if necessary.

OscMessage SetBlob(int index, Quaternion value)
Set a value as a byte blob at specified index, expanding message capacity if necessary.

OscMessage SetBlob(int index, Rect value)
Set a value as a byte blob at specified index, expanding message capacity if necessary.

OscMessage SetBlob(int index, Matrix4x4 value)
Set a value as a byte blob at specified index, expanding message capacity if necessary.

OscMessage SetBlob(int index, IList<float> values)
Writes a list of values to a byte blob at specified index, expanding message capacity if necessary.

OscMessage SetBlob(int index, IList<int> values)
Writes a list of values to a byte blob at specified index, expanding message capacity if necessary.

OscMessage SetBlob(int index, Encoding encoding, string value)
Writes a string with a given encoding to a byte blob at specified index, expanding message capacity if necessary.

override int Size()
Returns the byte size of the message.

void ToString(StringBuilder sb, bool appendNewLineAtEnd = false, int insertIndex = -1)

Alternative ToString() method that appends to a StringBuilder to avoid creating a new string. Set optional argument appendNewLineAtEnd append a new line at the end. Set optional argument insertIndex to Insert instead of Append.

OscPool

The Osc pool holds packets for reuse to reduce garbage generation.

static void Recycle(OscMessage message)
Recycle the specified message.

static void Recycle(OscBundle bundle)
Recycle the specified bundle.

static void Recycle(OscPacket packet)
Recycle the specified packet.