Namespace VsGlobal

Classes

Events

Network

Structs

Config

Struct Config

Namespace: VsGlobal

Assembly: VSGlobal.dll

```
public struct Config
```

Inherited Members
ValueType.Equals(object) , ValueType.GetHashCode() , ValueType.ToString() , object.Equals(object, object) , object.GetType() , object.ReferenceEquals(object, object)

Fields

api

```
public ICoreClientAPI api
```

Field Value

ICoreClientAPI

module

```
public string module
```

Field Value

string

player

```
public IClientPlayer player
```

Field Value

IClientPlayer

token

```
public Guid token
```

Field Value

Guid

Class Events

Namespace: VsGlobal

Assembly: VSGlobal.dll

```
public static class Events
```

Inheritance

object ← Events

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

OnClientReadyHandler(OnClientReadyEventArgs)

```
public static void OnClientReadyHandler(OnClientReadyEventArgs e)
```

Parameters

e OnClientReadyEventArgs

OnConnectHandler(OnConnectEventArgs)

```
public static void OnConnectHandler(OnConnectEventArgs e)
```

Parameters

e OnConnectEventArgs

OnDisconnectHandler(OnDisconnectEventArgs)

```
public static void OnDisconnectHandler(OnDisconnectEventArgs e)
```

Parameters

e OnDisconnectEventArgs

OnPayloadReceivedHandler(OnPayloadReceivedEventArgs)

```
public static void OnPayloadReceivedHandler(OnPayloadReceivedEventArgs e)
```

Parameters

e OnPayloadReceivedEventArgs

OnReconnectHandler(OnReconnectEventArgs)

```
public static void OnReconnectHandler(OnReconnectEventArgs e)
```

Parameters

e OnReconnectEventArgs

Events

OnClientReady

Invoked when the ICoreClientAPI.World.Player is fully loaded with OnClientReadyEventArgs

Using a lambda:

```
Events.OnClientReady += (e) =>
{
e.config.api.Logger.Info(e.config.module);
};
```

Using a function:

```
public void MyCoolOnClientReady(OnClientReadyEventArgs e)
{
// Do some stuff with e.config
e.config.api.ShowChatMessage(e.config.player.PlayerName);
}
```

```
                        // Then later, in a function body somewhere we register
the handler.
                        Events.OnClientReady += MyCoolOnClientReady;</code>
</pre></example>
```

```
public static event OnClientReadyHandler OnClientReady
```

Event Type

OnClientReadyHandler

OnConnect

Invoked when VsGlobal has connected with OnConnectEventArgs

Using a lambda:

```
Events.OnConnect += (e) =>
{
if(e.module == "my_module_name")
{
// Do stuff just for our module!
}
else
```

```
{
// Do stuff for any other module!
}
};
```

Using a function:

```
public void MyCoolOnConnect(OnConnectedEventArgs e)
{
// Do some stuff with e.module
}
```

```
                            // Then later, in a function body somewhere we register
the handler.
                            Events.OnConnect += MyCoolOnConnect;</code>
</pre></example>
```

```
public static event OnConnectHandler OnConnect
```

Event Type

OnConnectHandler

OnDisconnect

Invoked when VsGlobal has disconnected (banned, server issue, skill issue) with OnDisconnectEventArgs

Using a lambda:

```
Events.OnDisconnect += (e) =>
{
if(e.module == "my_module_name")
{
// Cleanup our mod code because we're DC'd.
}
else
{
// Likely don't care, but might care.
```

```
}
};
```

Using a function:

```
public void MyCoolOnDisconnect(OnDisconnectEventArgs e)
{
// Do some stuff with e.module
}
```

```
                            // Then later, in a function body somewhere we register
the handler.
                            Events.OnDisconnect += MyCoolOnDisconnect;</code>
</pre></example>
```

```
public static event OnDisconnectHandler OnDisconnect
```

Event Type

OnDisconnectHandler

OnPayloadReceived

Invoked when VsGlobal receives a payload with OnPayloadReceivedEventArgs

Using a lambda:

```
Events.OnPayloadReceived += (e) =>
{
// This will be called whenever a packet arrives, regardless of module or sender.
```

```
if(e.payload.Module == "my_module_name")

{

// Cleanup our mod code because we're DC'd.
```

```
MyCustomClass myCustomThing = e.payload.DeserializePacket<MyCustomClass>();

Console.WriteLine(myCustomThing.whateverValueOrFunction);

}

else

{

// It's someone else's packet. Could be handy!

}

};</code></pre><p>Using a function:</p><pre><code class="lang-csharp">public void
ReceiveMessagePacket(OnPayloadReceivedEventArgs e)
                            {
                                // Same as the lambda, we have access to any payload
coming in here.
                                Message? msg = e.payload.DeserializePacket<Message>
();
                                // We can also be quite cheeky and attempt to
deserialize it to our custom type regardless of module.
                                // If it doesn't, it's not ours- So I suppose that's
valid as well.
                                if(msg != null) { /* Do something with our received
custom message! */}
                            }

                            // Then later, in a function body somewhere we register
the handler.
                            Events.OnPayloadReceived += ReceiveMessagePacket;</code>
</pre></example>
```

```
public static event OnPayloadReceivedHandler OnPayloadReceived
```

Event Type

[OnPayloadReceivedHandler]

OnReconnect

Invoked when VsGlobal is trying to reconnect with OnReconnectEventArgs

Using a lambda:

```
Events.OnReconnect += (e) =>
{
if(e.module == "my_module_name")
{
// Cleanup our mod code because we're DC'd.
var myValue = e.attempts;
}
else
{
// Likely don't care, but might care.
}
};
```

Using a function:

```
public void MyCoolOnReconnect(OnReconnectEventArgs e)
{
if(e.attempts == 3 && e.module == "my_module_name") { /* Do stuff */ }
}
```

```
                            // Then later, in a function body somewhere we register
the handler.
                            Events.OnReconnect += MyCoolOnReconnect;</code>
</pre></example>
```

```
public static event OnReconnectHandler OnReconnect
```

Event Type

OnReconnectHandler

Class Network

Namespace: VsGlobal

Assembly: VSGlobal.dll

```
public static class Network
```

Inheritance

object ← Network

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

Broadcast<T>(T, string?)

Invoked when VsGlobal receives a payload with OnPayloadReceivedEventArgs

Note; we don't have to be connected to call this. It's thread safe and sitting there patiently for the websocket state to be 'Open' (connected)

Using a lambda:

```
//First, we define our network packet somewhere like so.
[ProtoContract(ImplicitFields = ImplicitFields.AllPublic)]
public class CustomNetworkMessage
{
public bool didSomething;
public IClientPlayer sender;
public string message = "Default Message";
}

// Later on, in a function body ...


// Now, all we have to do is call broadcast. It's generic, so you can throw
_anything_ in there. string, class, struct- Whatever.
```

```
// What that will do is send the packet to the server and relay it to others.

// Once received, it'll invoke

VsGlobal.Broadcast(new CustomNetworkMessage(){didSomething = true, sender =
api.World.Player, message = "Grungus"});</code></pre></example>
```

```
public static void Broadcast<T>(T packet, string? module = null)
```

Parameters

`packet` T

`module` string

Type Parameters

`T`

GetDefaultModule()

Test

```
public static string GetDefaultModule()
```

Returns

string

String

Initialize(ICoreClientAPI, string)

```
public static void Initialize(ICoreClientAPI api, string moduleName)
```

Parameters

`api` ICoreClientAPI

`moduleName` string

SetupHandlers(ICoreClientAPI)

```
public static void SetupHandlers(ICoreClientAPI api)
```

Parameters

`api` ICoreClientAPI

Namespace VsGlobal.AuthToken

Classes

Config

Class Config

Namespace: VsGlobal.AuthToken

Assembly: VSGlobal.dll

```
public static class Config
```

Inheritance

object ← Config

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

TryGetAuthToken(ICoreClientAPI)

```
public static Guid? TryGetAuthToken(ICoreClientAPI capi)
```

Parameters

`capi` ICoreClientAPI

Returns

Guid?

Namespace VsGlobal.EventArguments

Classes

OnClientReadyEventArgs

OnConnectEventArgs

OnDisconnectEventArgs

OnPayloadReceivedEventArgs

OnReconnectEventArgs

Class OnClientReadyEventArgs

Namespace: VsGlobal.EventArguments

Assembly: VSGlobal.dll

```
public class OnClientReadyEventArgs : EventArgs
```

Inheritance

object ← EventArgs ← OnClientReadyEventArgs

Inherited Members

EventArgs.Empty , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

config

```
public required Config config
```

Field Value

Config

Class OnConnectEventArgs

Namespace: VsGlobal.EventArguments

Assembly: VSGlobal.dll

```
public class OnConnectEventArgs : EventArgs
```

Inheritance

object ← EventArgs ← OnConnectEventArgs

Inherited Members

EventArgs.Empty , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

module

```
public required string module
```

Field Value

string

Class OnDisconnectEventArgs

Namespace: VsGlobal.EventArguments

Assembly: VSGlobal.dll

```
public class OnDisconnectEventArgs : EventArgs
```

Inheritance

object ← EventArgs ← OnDisconnectEventArgs

Inherited Members

EventArgs.Empty , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

module

```
public required string module
```

Field Value

string

Class OnPayloadReceivedEventArgs

Namespace: VsGlobal.EventArguments

Assembly: VSGlobal.dll

```
public class OnPayloadReceivedEventArgs : EventArgs
```

Inheritance

object ← EventArgs ← OnPayloadReceivedEventArgs

Inherited Members

EventArgs.Empty , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

payload

```
public required Payload payload
```

Field Value

Payload

Class OnReconnectEventArgs

Namespace: VsGlobal.EventArguments

Assembly: VSGlobal.dll

```
public class OnReconnectEventArgs : EventArgs
```

Inheritance

object ← EventArgs ← OnReconnectEventArgs

Inherited Members

EventArgs.Empty , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

attempts

```
public required int attempts
```

Field Value

int

module

```
public required string module
```

Field Value

string

Namespace VsGlobal.Handlers

Delegates

OnClientReadyHandler

OnConnectHandler

OnDisconnectHandler

OnPayloadReceivedHandler

OnReconnectHandler

Delegate OnClientReadyHandler

Namespace: VsGlobal.Handlers

Assembly: VSGlobal.dll

```
public delegate void OnClientReadyHandler(OnClientReadyEventArgs e)
```

Parameters

e OnClientReadyEventArgs

Delegate OnConnectHandler

Namespace: VsGlobal.Handlers

Assembly: VSGlobal.dll

```
public delegate void OnConnectHandler(OnConnectEventArgs e)
```

Parameters

e OnConnectEventArgs

Delegate OnDisconnectHandler

Namespace: VsGlobal.Handlers

Assembly: VSGlobal.dll

```
public delegate void OnDisconnectHandler(OnDisconnectEventArgs e)
```

Parameters

e OnDisconnectEventArgs

Delegate OnPayloadReceivedHandler

Namespace: VsGlobal.Handlers

Assembly: VSGlobal.dll

```
public delegate void OnPayloadReceivedHandler(OnPayloadReceivedEventArgs e)
```

Parameters

e OnPayloadReceivedEventArgs

Delegate OnReconnectHandler

Namespace: VsGlobal.Handlers

Assembly: VSGlobal.dll

```
public delegate void OnReconnectHandler(OnReconnectEventArgs e)
```

Parameters

e OnReconnectEventArgs

Namespace VsGlobal.Proto

Classes

Payload

PayloadExtensionMethods

Class Payload

Namespace: VsGlobal.Proto

Assembly: VSGlobal.dll

```
[ProtoContract]
public class Payload
```

Inheritance

object ← Payload

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Extension Methods

PayloadExtensionMethods.DeserializePacket<T>(Payload) , PayloadExtensionMethods.Serialize<T>(Payload, T)

Constructors

Payload()

```
public Payload()
```

Payload(string)

```
public Payload(string module)
```

Parameters

`module` string

Properties

Module

```
[ProtoMember(1)]
public string Module { get; set; }
```

Property Value

string

PacketType

```
[ProtoMember(3)]
public string PacketType { get; set; }
```

Property Value

string

PacketValue

```
[ProtoMember(4)]
public byte[] PacketValue { get; set; }
```

Property Value

byte[]

Processed

```
[ProtoMember(2)]
public bool Processed { get; set; }
```

Property Value

bool

Methods

Deserialize(byte[], int)

```
public static Payload Deserialize(byte[] buffer, int responseSize)
```

Parameters

`buffer` byte[]

`responseSize` int

Returns

Payload

Class PayloadExtensionMethods

Namespace: VsGlobal.Proto

Assembly: VSGlobal.dll

```
public static class PayloadExtensionMethods
```

Inheritance

object ← PayloadExtensionMethods

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

DeserializePacket<T>(Payload)

```
public static T? DeserializePacket<T>(this Payload payload)
```

Parameters

`payload` Payload

Returns

T

Type Parameters

`T`

Serialize<T>(Payload, T)

```
public static byte[] Serialize<T>(this Payload payload, T packetValue)
```

Parameters

`payload` Payload

`packetValue` T

Returns

byte[]

Type Parameters

`T`