Table of Contents

Namespace EasyCodeForVivox

Classes

Claims

Vivox Access Token(VAT) format class to generate valid request tokens. Read more on Vivox Documentation

Class/Model that will be seriliazed by JsonUtility to create the JSON payload that will be used to create the Vivox Access Token

- iss Token Issuer - Get From Vivox Developer Portal
- exp Expiration - Vivox Uses Unix Epoch time - Add Expiration time to Epoch value
- vxa Vivox Action to perform - Login, Join Channel, Kick, Mute etc.
- vxi Unique Identifier - Create from a custom counter or Unique GUID
- sub Subject : The User to mute, unmute, kick etc.
- f From : The User requesting an action, Usually self or Admin.
- t Channel : The Channel to join, mute, kick, transcribe(Speech-To-Text Vivox Paid Service) etc.

Easy3DPositional

EasyAccessToken

Copied from Vivox General Unity Documentation. Creates Secure Token for Vivox API requests, needed for production ready applications

Slightly Altered From Vivox Example To Create Proper Token

EasyAudio

EasyAudioChannel

EasyChannel

EasyLogin

EasyManager

EasyMessages

EasyMute

EasySession

EasySIP

EasyTextChannel

EasyTextToSpeech

EasyUsers

NetCode3DPositional

Enums

VoiceGender

Class Claims

Vivox Access Token(VAT) format class to generate valid request tokens. Read more on Vivox Documentation

Class/Model that will be seriliazed by JsonUtility to create the JSON payload that will be used to create the Vivox Access Token

- iss Token Issuer - Get From Vivox Developer Portal
- exp Expiration - Vivox Uses Unix Epoch time - Add Expiration time to Epoch value
- vxa Vivox Action to perform - Login, Join Channel, Kick, Mute etc.
- vxi Unique Identifier - Create from a custom counter or Unique GUID
- sub Subject : The User to mute, unmute, kick etc.
- f From : The User requesting an action, Usually self or Admin.
- t Channel : The Channel to join, mute, kick, transcribe(Speech-To-Text Vivox Paid Service) etc.

Inheritance

System.Object
Claims

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class Claims
```

Properties

exp

Epoch Time : Vivox uses Unix Epoch time. ex. DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc)

Declaration

```
public int exp { get; set; }
```

Property Value

TYPE	DESCRIPTION
System.Int32	

f

From : The user requesting a claim/action ex. format == sip:.blindmelon-AppName-dev.beef.@tla.vivox.com

Declaration

```
public string f { get; set; }
```

Property Value

TYPE	DESCRIPTION
System.String	

iss

Issuer : Get from Vivox Developer Portal

Declaration

```
public string iss { get; set; }
```

Property Value

TYPE	DESCRIPTION
System.String	

sub

Subject : The user to be muted, kicked, unmuted ex. format == sip:.blindmelon-AppName-dev.jerky.@tla.vivox.com

Declaration

```
public string sub { get; set; }
```

Property Value

TYPE	DESCRIPTION
System.String	

t

Channel : Channel where action/claim takes place. ex. format == sip:confctl-g-blindmelon-AppName-dev.testchannel@tla.vivox.com

Declaration

```
public string t { get; set; }
```

Property Value

TYPE	DESCRIPTION
System.String	

vxa

VixoxAction : ex. login, join, mute

Declaration

```
public string vxa { get; set; }
```

Property Value

TYPE	DESCRIPTION
System.String	

TYPE	DESCRIPTION

vxi

Token Uniqueness Identifier : Can be any number. Recommended to use an incrimental counter so every token generated will always be different. ex. int counter = 0; counter++;

Declaration

```
public int vxi { get; set; }
```

Property Value

TYPE	DESCRIPTION
System.Int32	

Class Easy3DPositional

Inheritance

System.Object
Easy3DPositional

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class Easy3DPositional : MonoBehaviour
```

Fields

listenerPosition

Declaration

```
public Transform listenerPosition
```

Field Value

TYPE	DESCRIPTION
Transform	

speakerPosition

Declaration

```
public Transform speakerPosition
```

Field Value

TYPE	DESCRIPTION
Transform	

Methods

CheckIfChannelExists()

Declaration

```
public bool CheckIfChannelExists()
```

Returns

TYPE	DESCRIPTION
System.Boolean	

Update3DPosition()

Declaration

```
public void Update3DPosition()
```

Class EasyAccessToken

Copied from Vivox General Unity Documentation. Creates Secure Token for Vivox API requests, needed for production ready applications

Slightly Altered From Vivox Example To Create Proper Token

Inheritance

System.Object
EasyAccessToken

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyAccessToken : MonoBehaviour
```

Fields

UnixEpoch

Declaration

```
public static readonly DateTime UnixEpoch
```

Field Value

TYPE	DESCRIPTION
DateTime	Gets Unix Epoch (January 1st, 1970, 00:00:00) to create valid expiration times for Vivox Access Tokens- Used in SecondsSinceUnixEpochPlusDuration(Nullable<TimeSpan>)

Methods

CreateToken(String, String, Int32, String, Int32, String, String, String)

Vivox Access Token(VAT) format class to generate valid request tokens. Read more on Vivox Documentation

This is the only method needed to create all neccessary types of tokens In Vivox

Names and acronyms are mostly consistent with Vivox Documentation to avoid confusion but expanded upon for better understanding

- key Token Key - Get From Vivox Developer Portal
- iss Token Issuer - Get From Vivox Developer Portal
- exp Expiration - Vivox Uses Unix Epoch time - Add Expiration time to Epoch value
- vxa Vivox Action to perform - Refer To Vivox Documentation
- vxi Unique Identifier - Create from a custom counter or Unique GUID
- sub Subject : The User to mute, unmute, kick etc.
- f From : The User requesting an action, Usually self or Admin.
- t Channel : The Channel to join, mute, kick, transcribe(Speech-To-Text Vivox Paid Service) etc.

Declaration

```
public static string CreateToken(string key, string issuer, int exp, string vxa, int vxi, string sub, string
f, string t)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	key	Token Key From Vivox Developer Portal
System.String	issuer	Application Issuer - Vivox Developer Portal
System.Int32	exp	Time in epoch + 90 seconds or prefered timeout
System.String	vxa	Vivox Action to perform : ex. login, kick, join
System.Int32	vxi	Unique identifier to garauntee unique Token. Recommended to use counter on server
System.String	sub	sub == Subject : The User to mute, unmute, kick etc.
System.String	f	f == From : The User requesting an action
System.String	t	t == Channel : The Channel to join, mute, kick, transcribe etc.

Returns

TYPE	DESCRIPTION
System.String	A Valid Token For Production Code with Vivox (JWT with empty header)

Remarks

Token creation for Kicking people from channels, Muting people, Muting All except one person(Presentation Mode).

If (Admin) you can kick people from channels or servers.

If (Admin) you can mute people in channels, muting all except one(Presentation Mode).

SIP URI(Address) required for f, t, and sub.

SUB, F, T Can/Should be Null if not needed for the claim/action request.

ex. login only needs the f paramater, sub == null, t == null.

ex. Token_F("yourTokenKey", "blindmelon-AppName-dev", (int)epochTime, "login", 0001, null, &quot;sip:.blindmelon-AppName-dev.jerky.@tla.vivox.com&quot;, null)

SecondsSinceUnixEpochPlusDuration(Nullable<TimeSpan>)

Copied Implementation From Vivox API. Used for obtaining time in seconds of Unix Epoch to Now(Current Time) with the option of an added duration.

Declaration

```
public static int SecondsSinceUnixEpochPlusDuration(TimeSpan? duration = default(TimeSpan? ))
```

Parameters

TYPE	NAME	DESCRIPTION
System.Nullable<TimeSpan>	duration	Timespan ahead of (DateTime.UtcNow - Unix Epoch) you want to have a timestamp for.

Returns

TYPE	DESCRIPTION
System.Int32	The time in seconds from Unix Epoch (January 1st, 1970, 00:00:00) to DateTime.UtcNow with an added duration.

Class EasyAudio

Inheritance

System.Object
EasyAudio

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyAudio
```

Constructors

EasyAudio(EasySettingsSO, EasyEvents, EasyEventsAsync)

Declaration

```
public EasyAudio(EasySettingsSO settings, EasyEvents events, EasyEventsAsync eventsAsync)
```

Parameters

TYPE	NAME	DESCRIPTION
EasySettingsSO	settings	
EasyEvents	events	
EasyEventsAsync	eventsAsync	

Methods

AdjustLocalPlayerAudioVolume(Int32, VivoxUnity.Client)

Declaration

```
public void AdjustLocalPlayerAudioVolume(int value, VivoxUnity.Client client)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Int32	value	
VivoxUnity.Client	client	

AdjustRemotePlayerAudioVolume(String, IChannelSession, Single)

Declaration

```
public void AdjustRemotePlayerAudioVolume(string userName, IChannelSession channelSession, float value)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
IChannelSession	channelSession	
System.Single	value	

GetAudioInputDevices(VivoxUnity.Client)

Declaration

```
public IEnumerable<IAudioDevice> GetAudioInputDevices(VivoxUnity.Client client)
```

Parameters

TYPE	NAME	DESCRIPTION
VivoxUnity.Client	client	

Returns

TYPE	DESCRIPTION
IEnumerable<IAudioDevice>	

GetAudioOutputDevices(VivoxUnity.Client)

Declaration

```
public IEnumerable<IAudioDevice> GetAudioOutputDevices(VivoxUnity.Client client)
```

Parameters

TYPE	NAME	DESCRIPTION
VivoxUnity.Client	client	

Returns

TYPE	DESCRIPTION
IEnumerable<IAudioDevice>	

RefreshAudioInputDevices(VivoxUnity.Client)

Declaration

```
public void RefreshAudioInputDevices(VivoxUnity.Client client)
```

Parameters

TYPE	NAME	DESCRIPTION
VivoxUnity.Client	client	

RefreshAudioOutputDevices(VivoxUnity.Client)

Declaration

```
public void RefreshAudioOutputDevices(VivoxUnity.Client client)
```

Parameters

TYPE	NAME	DESCRIPTION
VivoxUnity.Client	client	

SetAudioInputDevice(String, VivoxUnity.Client)

Declaration

```
public void SetAudioInputDevice(string deviceName, VivoxUnity.Client client)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	deviceName	
VivoxUnity.Client	client	

SetAudioOutputDevice(String, VivoxUnity.Client)

Declaration

```
public void SetAudioOutputDevice(string deviceName, VivoxUnity.Client client)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	deviceName	
VivoxUnity.Client	client	

SetAutoVoiceActivityDetection(String)

Declaration

```
public void SetAutoVoiceActivityDetection(string userName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	

SetVoiceActivityDetection(String, Int32, Int32, Int32)

Declaration

```
public void SetVoiceActivityDetection(string userName, int hangover = 2000, int sensitivity = 43, int
noiseFloor = 576)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.Int32	hangover	
System.Int32	sensitivity	
System.Int32	noiseFloor	

StartAudioInjection(String, ILoginSession)

Declaration

```
public void StartAudioInjection(string wavToInject, ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	wavToInject	
ILoginSession	loginSession	

StopAudioInjection(ILoginSession)

Declaration

```
public void StopAudioInjection(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

Subscribe(VivoxUnity.Client)

Declaration

```
public void Subscribe(VivoxUnity.Client client)
```

Parameters

TYPE	NAME	DESCRIPTION
VivoxUnity.Client	client	

Unsubscribe(VivoxUnity.Client)

Declaration

```
public void Unsubscribe(VivoxUnity.Client client)
```

Parameters

TYPE	NAME	DESCRIPTION
VivoxUnity.Client	client	

Class EasyAudioChannel

Inheritance

System.Object
EasyAudioChannel

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyAudioChannel
```

Constructors

EasyAudioChannel(EasyEvents, EasyEventsAsync)

Declaration

```
public EasyAudioChannel(EasyEvents events, EasyEventsAsync eventsAsync)
```

Parameters

TYPE	NAME	DESCRIPTION
EasyEvents	events	
EasyEventsAsync	eventsAsync	

Methods

Subscribe(IChannelSession)

Declaration

```
public void Subscribe(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

ToggleAudioInChannel(IChannelSession, Boolean)

Declaration

```
public void ToggleAudioInChannel(IChannelSession channelSession, bool join)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
System.Boolean	join	

ToggleAudioInChannel<T>(IChannelSession, Boolean, T)

Declaration

```
public void ToggleAudioInChannel<T>(IChannelSession channelSession, bool join, T eventParameter)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
System.Boolean	join	
T	eventParameter	

Type Parameters

NAME	DESCRIPTION
T	

Unsubscribe(IChannelSession)

Declaration

```
public void Unsubscribe(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Class EasyChannel

Inheritance

System.Object
EasyChannel

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyChannel
```

Constructors

EasyChannel(EasyUsers, EasyMessages, EasyAudioChannel, EasyTextChannel, EasyEventsAsync, EasyEvents)

Declaration

```
public EasyChannel(EasyUsers users, EasyMessages messages, EasyAudioChannel audioChannel, EasyTextChannel
textChannel, EasyEventsAsync eventsAsync, EasyEvents events)
```

Parameters

TYPE	NAME	DESCRIPTION
EasyUsers	users	
EasyMessages	messages	
EasyAudioChannel	audioChannel	
EasyTextChannel	textChannel	
EasyEventsAsync	eventsAsync	
EasyEvents	events	

Methods

CreateNewChannel(String, String, ChannelType, Channel3DProperties)

Declaration

```
public IChannelSession CreateNewChannel(string userName, string channelName, ChannelType channelType,
Channel3DProperties channel3DProperties = null)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	
ChannelType	channelType	
Channel3DProperties	channel3DProperties	

Returns

TYPE	DESCRIPTION
IChannelSession	

GetChannelSIP(ChannelType, String, Channel3DProperties)

Declaration

```
public string GetChannelSIP(ChannelType channelType, string channelName, Channel3DProperties
channel3DProperties = null)
```

Parameters

TYPE	NAME	DESCRIPTION
ChannelType	channelType	
System.String	channelName	
Channel3DProperties	channel3DProperties	

Returns

TYPE	DESCRIPTION
System.String	

GetChannelSIP(String)

Declaration

```
public string GetChannelSIP(string channelName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	channelName	

Returns

TYPE	DESCRIPTION
System.String	

GetChannelToken(String, IChannelSession, Boolean, Channel3DProperties)

Declaration

```
public string GetChannelToken(string userName, IChannelSession channelSession, bool joinMuted = false,
Channel3DProperties channel3DProperties = null)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
IChannelSession	channelSession	
System.Boolean	joinMuted	
Channel3DProperties	channel3DProperties	

Returns

TYPE	DESCRIPTION
System.String	

GetExistingChannelSession(String, String)

Declaration

```
public IChannelSession GetExistingChannelSession(string userName, string channelName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	

Returns

TYPE	DESCRIPTION
IChannelSession	

GetRegionChannelToken(String, IChannelSession, String, String, Boolean, Channel3DProperties)

Declaration

```
public string GetRegionChannelToken(string userName, IChannelSession channelSession, string matchRegion,
string matchHash, bool joinMuted = false, Channel3DProperties channel3DProperties = null)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
IChannelSession	channelSession	
System.String	matchRegion	

TYPE	NAME	DESCRIPTION
System.String	matchHash	
System.Boolean	joinMuted	
Channel3DProperties	channel3DProperties	

Returns

TYPE	DESCRIPTION
System.String	

JoinChannel(String, Boolean, Boolean, Boolean, IChannelSession, Boolean)

Declaration

```
protected void JoinChannel(string userName, bool includeVoice, bool includeText, bool
switchTransmissionToThisChannel, IChannelSession channelSession, bool joinMuted = false)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.Boolean	includeVoice	
System.Boolean	includeText	
System.Boolean	switchTransmissionToThisChannel	
IChannelSession	channelSession	
System.Boolean	joinMuted	

JoinChannel(String, String, Boolean, Boolean, Boolean, ChannelType, Boolean, Channel3DProperties)

Declaration

```
public void JoinChannel(string userName, string channelName, bool includeVoice, bool includeText, bool
switchTransmissionToThisChannel, ChannelType channelType, bool joinMuted = false, Channel3DProperties
channel3DProperties = null)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	
System.Boolean	includeVoice	
System.Boolean	includeText	

TYPE	NAME	DESCRIPTION
System.Boolean	switchTransmissionToThisChannel	
ChannelType	channelType	
System.Boolean	joinMuted	
Channel3DProperties	channel3DProperties	

JoinChannelCustom<T>(String, T, Boolean, Boolean, Boolean, IChannelSession, Boolean)

Declaration

```
protected void JoinChannelCustom<T>(string userName, T value, bool includeVoice, bool includeText, bool
switchTransmissionToThisChannel, IChannelSession channelSession, bool joinMuted = false)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
T	value	
System.Boolean	includeVoice	
System.Boolean	includeText	
System.Boolean	switchTransmissionToThisChannel	
IChannelSession	channelSession	
System.Boolean	joinMuted	

Type Parameters

NAME	DESCRIPTION
T	

JoinChannelCustom<T>(String, String, T, Boolean, Boolean, Boolean, ChannelType, Boolean, Channel3DProperties)

Declaration

```
public void JoinChannelCustom<T>(string userName, string channelName, T eventParameter, bool includeVoice,
bool includeText, bool switchTransmissionToThisChannel, ChannelType channelType, bool joinMuted = false,
Channel3DProperties channel3DProperties = null)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	

TYPE	NAME	DESCRIPTION
T	eventParameter	
System.Boolean	includeVoice	
System.Boolean	includeText	
System.Boolean	switchTransmissionToThisChannel	
ChannelType	channelType	
System.Boolean	joinMuted	
Channel3DProperties	channel3DProperties	

Type Parameters

NAME	DESCRIPTION
T	

JoinChannelRegion(String, String, String, Boolean, Boolean, Boolean, IChannelSession, Boolean)

Declaration

```
protected void JoinChannelRegion(string userName, string matchRegion, string matchHash, bool includeVoice,
bool includeText, bool switchTransmissionToThisChannel, IChannelSession channelSession, bool joinMuted =
false)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	matchRegion	
System.String	matchHash	
System.Boolean	includeVoice	
System.Boolean	includeText	
System.Boolean	switchTransmissionToThisChannel	
IChannelSession	channelSession	
System.Boolean	joinMuted	

JoinChannelRegion(String, String, String, String, Boolean, Boolean, Boolean, ChannelType, Boolean, Channel3DProperties)

Declaration

```
public void JoinChannelRegion(string userName, string channelName, string matchRegion, string matchHash, bool includeVoice, bool includeText, bool switchTransmissionToThisChannel, ChannelType channelType, bool joinMuted = false, Channel3DProperties channel3DProperties = null)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	
System.String	matchRegion	
System.String	matchHash	
System.Boolean	includeVoice	
System.Boolean	includeText	
System.Boolean	switchTransmissionToThisChannel	
ChannelType	channelType	
System.Boolean	joinMuted	
Channel3DProperties	channel3DProperties	

LeaveChannel(ILoginSession, IChannelSession)

Declaration

```
public void LeaveChannel(ILoginSession loginSession, IChannelSession channelToRemove)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
IChannelSession	channelToRemove	

LeaveChannel(String, String)

Declaration

```
public void LeaveChannel(string channelName, string userName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	channelName	
System.String	userName	

OnChannelStatePropertyChanged(Object, PropertyChangedEventArgs)

Declaration

```
public void OnChannelStatePropertyChanged(object sender, PropertyChangedEventArgs channelArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Object	sender	
PropertyChangedEventArgs	channelArgs	

RemoveChannelSession(String)

Declaration

```
public void RemoveChannelSession(string channelName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	channelName	

Subscribe(IChannelSession)

Declaration

```
public void Subscribe(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Unsubscribe(IChannelSession)

Declaration

```
public void Unsubscribe(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Class EasyLogin

Inheritance

System.Object
EasyLogin

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyLogin
```

Constructors

EasyLogin(EasyMessages, EasyTextToSpeech, EasyEvents, EasyEventsAsync, EasySettingsSO, EasyMute)

Declaration

```
public EasyLogin(EasyMessages messages, EasyTextToSpeech textToSpeech, EasyEvents eventsSync, EasyEventsAsync
eventsAync, EasySettingsSO easySettings, EasyMute mute)
```

Parameters

TYPE	NAME	DESCRIPTION
EasyMessages	messages	
EasyTextToSpeech	textToSpeech	
EasyEvents	eventsSync	
EasyEventsAsync	eventsAync	
EasySettingsSO	easySettings	
EasyMute	mute	

Methods

GetChannelId(String, String)

Declaration

```
public ChannelId GetChannelId(string userName, string channelName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	

Returns

TYPE	DESCRIPTION
ChannelId	

LoginToVivox(ILoginSession, Uri, String, Boolean)

Declaration

```
protected void LoginToVivox(ILoginSession loginSession, Uri serverUri, string userName, bool joinMuted =
false)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
Uri	serverUri	
System.String	userName	
System.Boolean	joinMuted	

LoginToVivox(String, String, Boolean)

Declaration

```
public void LoginToVivox(string userName, string displayName = null, bool joinMuted = false)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	displayName	
System.Boolean	joinMuted	

LoginToVivox<T>(ILoginSession, T, Uri, String, Boolean)

Declaration

```
protected void LoginToVivox<T>(ILoginSession loginSession, T value, Uri serverUri, string userName, bool
joinMuted = false)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
T	value	
Uri	serverUri	
System.String	userName	
System.Boolean	joinMuted	

Type Parameters

NAME	DESCRIPTION
T	

LoginToVivox<T>(String, T, String, Boolean)

Declaration

```
public void LoginToVivox<T>(string userName, T value, string displayName = null, bool joinMuted = false)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
T	value	
System.String	displayName	
System.Boolean	joinMuted	

Type Parameters

NAME	DESCRIPTION
T	

LogoutOfVivox(String)

Declaration

```
public void LogoutOfVivox(string userName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	

LogoutOfVivox<T>(String, T)

Declaration

```
public void LogoutOfVivox<T>(string userName, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnLoginAdded(Object, KeyEventArg<AccountId>)

Declaration

```
public void OnLoginAdded(object sender, KeyEventArg<AccountId> accountId)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Object	sender	
KeyEventArg<AccountId>	accountId	

OnLoginPropertyChanged(Object, PropertyChangedEventArgs)

Declaration

```
public void OnLoginPropertyChanged(object sender, PropertyChangedEventArgs propArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Object	sender	
PropertyChangedEventArgs	propArgs	

OnLoginRemoved(Object, KeyEventArg<AccountId>)

Declaration

```
public void OnLoginRemoved(object sender, KeyEventArg<AccountId> accountId)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Object	sender	
KeyEventArg<AccountId>	accountId	

OnLoginUpdated(Object, ValueEventArg<AccountId, ILoginSession>)

Declaration

```
public void OnLoginUpdated(object sender, ValueEventArg<AccountId, ILoginSession> valueEventArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Object	sender	
ValueEventArg<AccountId, ILoginSession>	valueEventArgs	

SetPlayerTransmissionMode(String, TransmissionMode, ChannelId)

Declaration

```
public void SetPlayerTransmissionMode(string userName, TransmissionMode transmissionMode, ChannelId channelId
= null)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
TransmissionMode	transmissionMode	
ChannelId	channelId	

UpdateLoginProperties(String, ParticipantPropertyUpdateFrequency)

Declaration

```
public void UpdateLoginProperties(string userName, ParticipantPropertyUpdateFrequency updateFrequency)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
ParticipantPropertyUpdateFrequency	updateFrequency	

Class EasyManager

Inheritance

System.Object
EasyManager

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyManager : MonoBehaviour
```

Methods

AdjustLocalUserVolume(Int32)

Declaration

```
public void AdjustLocalUserVolume(int volume)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Int32	volume	

AdjustRemoteUserVolume(String, String, Single)

Declaration

```
public void AdjustRemoteUserVolume(string userName, string channelName, float volume)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	
System.Single	volume	

ChooseVoiceGender(VoiceGender, String)

Declaration

```
public void ChooseVoiceGender(VoiceGender voiceGender, string userName)
```

Parameters

TYPE	NAME	DESCRIPTION
VoiceGender	voiceGender	
System.String	userName	

ClearCrossMutedUsersForLoginSession(String)

Declaration

```
public void ClearCrossMutedUsersForLoginSession(string loggedInUserName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	loggedInUserName	

CrossMuteUser(String, String, String, Boolean)

Declaration

```
public void CrossMuteUser(string userName, string channelName, string userToMute, bool mute)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	
System.String	userToMute	
System.Boolean	mute	

CrossMuteUsers(String, String, List<String>, Boolean)

Declaration

```
public void CrossMuteUsers(string userName, string channelName, List<string> usersToMute, bool mute)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	
List<System.String>	usersToMute	
System.Boolean	mute	

EnablePushToTalk(Boolean, KeyCode)

Declaration

```
public void EnablePushToTalk(bool enable, KeyCode keyCode)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Boolean	enable	
KeyCode	keyCode	

GetAudioInputDevices()

Declaration

```
public IEnumerable<IAudioDevice> GetAudioInputDevices()
```

Returns

TYPE	DESCRIPTION
IEnumerable<IAudioDevice>	

GetAudioOutputDevices()

Declaration

```
public IEnumerable<IAudioDevice> GetAudioOutputDevices()
```

Returns

TYPE	DESCRIPTION
IEnumerable<IAudioDevice>	

GetChannelId(String, String)

Declaration

```
public ChannelId GetChannelId(string userName, string channelName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	

Returns

TYPE	DESCRIPTION
ChannelId	

GetTransmittingChannelsForPlayer(String)

Declaration

```
public IEnumerable<ChannelId> GetTransmittingChannelsForPlayer(string userName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	

Returns

TYPE	DESCRIPTION
IEnumerable<ChannelId>	

InitializeClient(VivoxConfig)

Declaration

```
public Task InitializeClient(VivoxConfig vivoxConfig = null)
```

Parameters

TYPE	NAME	DESCRIPTION
VivoxConfig	vivoxConfig	

Returns

TYPE	DESCRIPTION
Task	

IsPlayerTransmittingInChannel(String)

Declaration

```
public bool IsPlayerTransmittingInChannel(string channelName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	channelName	

Returns

TYPE	DESCRIPTION
System.Boolean	

JoinChannel(String, String, Boolean, Boolean, Boolean, ChannelType, Boolean, Channel3DProperties)

Declaration

```
public void JoinChannel(string userName, string channelName, bool includeVoice, bool includeText, bool
switchTransmissionToThisChannel, ChannelType channelType, bool joinMuted = false, Channel3DProperties
channel3DProperties = null)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	
System.Boolean	includeVoice	
System.Boolean	includeText	
System.Boolean	switchTransmissionToThisChannel	
ChannelType	channelType	

TYPE	NAME	DESCRIPTION
System.Boolean	joinMuted	
Channel3DProperties	channel3DProperties	

JoinChannelRegion(String, String, String, String, Boolean, Boolean, Boolean, ChannelType, Boolean, Channel3DProperties)

Declaration

```
public void JoinChannelRegion(string userName, string channelName, string matchRegion, string matchHash, bool
includeVoice, bool includeText, bool switchTransmissionToThisChannel, ChannelType channelType, bool joinMuted
= false, Channel3DProperties channel3DProperties = null)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	
System.String	matchRegion	
System.String	matchHash	
System.Boolean	includeVoice	
System.Boolean	includeText	
System.Boolean	switchTransmissionToThisChannel	
ChannelType	channelType	
System.Boolean	joinMuted	
Channel3DProperties	channel3DProperties	

LeaveChannel(String, String)

Declaration

```
public void LeaveChannel(string channelName, string userName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	channelName	
System.String	userName	

LocalMuteAllPlayers(String)

Declaration

```
public void LocalMuteAllPlayers(string channelName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	channelName	

LocalMuteRemoteUser(String, String)

Declaration

```
public void LocalMuteRemoteUser(string userName, string channelName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	

LocalUnmuteAllPlayers(String)

Declaration

```
public void LocalUnmuteAllPlayers(string channelName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	channelName	

LocalUnmuteRemoteUser(String, String)

Declaration

```
public void LocalUnmuteRemoteUser(string userName, string channelName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	

LoginToVivox(String, String, Boolean)

Declaration

```
public void LoginToVivox(string userName, string displayName = null, bool joinMuted = false)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	displayName	

TYPE	NAME	DESCRIPTION
System.Boolean	joinMuted	

LogoutOfVivox(String)

Declaration

```
public void LogoutOfVivox(string userName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	

MuteSelf()

Declaration

```
public void MuteSelf()
```

OnAudioChannelConnected(IChannelSession)

Declaration

```
protected virtual void OnAudioChannelConnected(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnAudioChannelConnecting(IChannelSession)

Declaration

```
protected virtual void OnAudioChannelConnecting(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnAudioChannelDisconnected(IChannelSession)

Declaration

```
protected virtual void OnAudioChannelDisconnected(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnAudioChannelDisconnecting(IChannelSession)

Declaration

```
protected virtual void OnAudioChannelDisconnecting(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnAudioInputDeviceAdded(IAudioDevice)

Declaration

```
protected virtual void OnAudioInputDeviceAdded(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioInputDeviceRemoved(IAudioDevice)

Declaration

```
protected virtual void OnAudioInputDeviceRemoved(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioInputDeviceUpdated(IAudioDevice)

Declaration

```
protected virtual void OnAudioInputDeviceUpdated(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioOutputDeviceAdded(IAudioDevice)

Declaration

```
protected virtual void OnAudioOutputDeviceAdded(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioOutputDeviceRemoved(IAudioDevice)

Declaration

```
protected virtual void OnAudioOutputDeviceRemoved(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioOutputDeviceUpdated(IAudioDevice)

Declaration

```
protected virtual void OnAudioOutputDeviceUpdated(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnChannelConnected(IChannelSession)

Declaration

```
protected virtual void OnChannelConnected(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnChannelConnecting(IChannelSession)

Declaration

```
protected virtual void OnChannelConnecting(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnChannelDisconnected(IChannelSession)

Declaration

```
protected virtual void OnChannelDisconnected(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnChannelDisconnecting(IChannelSession)

Declaration

```
protected virtual void OnChannelDisconnecting(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnChannelMessageRecieved(IChannelTextMessage)

Declaration

```
protected virtual void OnChannelMessageRecieved(IChannelTextMessage textMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelTextMessage	textMessage	

OnCrossMuted(AccountId)

Declaration

```
protected virtual void OnCrossMuted(AccountId accountId)
```

Parameters

TYPE	NAME	DESCRIPTION
AccountId	accountId	

OnCrossUnmuted(AccountId)

Declaration

```
protected virtual void OnCrossUnmuted(AccountId accountId)
```

Parameters

TYPE	NAME	DESCRIPTION
AccountId	accountId	

OnDirectMessageFailed(IFailedDirectedTextMessage)

Declaration

```
protected virtual void OnDirectMessageFailed(IFailedDirectedTextMessage failedMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
IFailedDirectedTextMessage	failedMessage	

OnDirectMessageRecieved(IDirectedTextMessage)

Declaration

```
protected virtual void OnDirectMessageRecieved(IDirectedTextMessage directedTextMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
IDirectedTextMessage	directedTextMessage	

OnEventMessageRecieved(IChannelTextMessage)

Declaration

```
protected virtual void OnEventMessageRecieved(IChannelTextMessage textMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelTextMessage	textMessage	

OnLocalUserMuted()

Declaration

```
protected virtual void OnLocalUserMuted()
```

OnLocalUserUnmuted()

Declaration

```
protected virtual void OnLocalUserUnmuted()
```

OnLoggedIn(ILoginSession)

Declaration

```
protected virtual void OnLoggedIn(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

OnLoggedOut(ILoginSession)

Declaration

```
protected virtual void OnLoggedOut(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

OnLoggingIn(ILoginSession)

Declaration

```
protected virtual void OnLoggingIn(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

OnLoggingOut(ILoginSession)

Declaration

```
protected virtual void OnLoggingOut(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

OnLoginAdded(AccountId)

Declaration

```
protected virtual void OnLoginAdded(AccountId accountId)
```

Parameters

TYPE	NAME	DESCRIPTION
AccountId	accountId	

OnLoginRemoved(AccountId)

Declaration

```
protected virtual void OnLoginRemoved(AccountId accountId)
```

Parameters

TYPE	NAME	DESCRIPTION
AccountId	accountId	

OnLoginUpdated(ILoginSession)

Declaration

```
protected virtual void OnLoginUpdated(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

OnTextChannelConnected(IChannelSession)

Declaration

```
protected virtual void OnTextChannelConnected(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnTextChannelConnecting(IChannelSession)

Declaration

```
protected virtual void OnTextChannelConnecting(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnTextChannelDisconnected(IChannelSession)

Declaration

```
protected virtual void OnTextChannelDisconnected(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnTextChannelDisconnecting(IChannelSession)

Declaration

```
protected virtual void OnTextChannelDisconnecting(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnTTSMessageAdded(ITTSMessageQueueEventArgs)

Declaration

```
protected virtual void OnTTSMessageAdded(ITTSMessageQueueEventArgs ttsArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
ITTSMessageQueueEventArgs	ttsArgs	

OnTTSMessageRemoved(ITTSMessageQueueEventArgs)

Declaration

```
protected virtual void OnTTSMessageRemoved(ITTSMessageQueueEventArgs ttsArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
ITTSMessageQueueEventArgs	ttsArgs	

OnTTSMessageUpdated(ITTSMessageQueueEventArgs)

Declaration

```
protected virtual void OnTTSMessageUpdated(ITTSMessageQueueEventArgs ttsArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
ITTSMessageQueueEventArgs	ttsArgs	

OnUserJoinedChannel(IParticipant)

Declaration

```
protected virtual void OnUserJoinedChannel(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

OnUserLeftChannel(IParticipant)

Declaration

```
protected virtual void OnUserLeftChannel(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

OnUserMuted(IParticipant)

Declaration

```
protected virtual void OnUserMuted(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

OnUserNotSpeaking(IParticipant)

Declaration

```
protected virtual void OnUserNotSpeaking(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

OnUserSpeaking(IParticipant)

Declaration

```
protected virtual void OnUserSpeaking(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

OnUserUnmuted(IParticipant)

Declaration

```
protected virtual void OnUserUnmuted(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

OnUserValuesUpdated(IParticipant)

Declaration

```
protected virtual void OnUserValuesUpdated(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

PlayTTSMessage(String, String)

Declaration

```
public void PlayTTSMessage(string msg, string userName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	msg	
System.String	userName	

PlayTTSMessage(String, String, TTSDestination)

Declaration

```
public void PlayTTSMessage(string msg, string userName, TTSDestination playMode)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	msg	
System.String	userName	
TTSDestination	playMode	

RefreshAudioInputDevices()

Declaration

```
public void RefreshAudioInputDevices()
```

RefreshAudioOutputDevices()

Declaration

```
public void RefreshAudioOutputDevices()
```

SendChannelMessage(String, String, String, String, String)

Declaration

```
public void SendChannelMessage(string userName, string channelName, string msg, string header = "", string
body = "")
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	
System.String	msg	
System.String	header	
System.String	body	

SendDirectMessage(String, String, String, String, String)

Declaration

```
public void SendDirectMessage(string userName, string userToMsg, string msg, string header = "", string body =
"")
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	userToMsg	
System.String	msg	

TYPE	NAME	DESCRIPTION
System.String	header	
System.String	body	

SendEventMessage(String, String, String, String, String)

Declaration

```
public void SendEventMessage(string userName, string channelName, string msg, string header = "", string body
= "")
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	
System.String	msg	
System.String	header	
System.String	body	

SetAudioInputDevice(String)

Declaration

```
public void SetAudioInputDevice(string deviceName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	deviceName	

SetAudioOutputDevice(String)

Declaration

```
public void SetAudioOutputDevice(string deviceName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	deviceName	

SetAutoVoiceActivityDetection(String)

Declaration

```
public void SetAutoVoiceActivityDetection(string userName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	

SetPlayerTransmissionMode(String, TransmissionMode, ChannelId)

Declaration

```
public void SetPlayerTransmissionMode(string userName, TransmissionMode transmissionMode, ChannelId channelId
= null)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
TransmissionMode	transmissionMode	
ChannelId	channelId	

SetVoiceActivityDetection(String, Int32, Int32, Int32)

Declaration

```
public void SetVoiceActivityDetection(string userName, int hangover, int sensitivity, int noiseFloor)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.Int32	hangover	
System.Int32	sensitivity	
System.Int32	noiseFloor	

StartAudioInjection(String, String)

Declaration

```
public void StartAudioInjection(string username, string audioFilePath)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	username	
System.String	audioFilePath	

StopAudioInjection(String)

Declaration

```
public void StopAudioInjection(string username)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	username	

SubscribeToVivoxEvents()

Declaration

```
public void SubscribeToVivoxEvents()
```

ToggleAudioInChannel(String, Boolean)

Declaration

```
public void ToggleAudioInChannel(string channelName, bool connect)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	channelName	
System.Boolean	connect	

ToggleTextInChannel(String, Boolean)

Declaration

```
public void ToggleTextInChannel(string channelName, bool connect)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	channelName	
System.Boolean	connect	

UnitializeClient()

Declaration

```
public void UnitializeClient()
```

UnmuteSelf()

Declaration

```
public void UnmuteSelf()
```

UnsubscribeToVivoxEvents()

Declaration

```
public void UnsubscribeToVivoxEvents()
```

UpdateLoginProperties(String, ParticipantPropertyUpdateFrequency)

Declaration

```
public void UpdateLoginProperties(string userName, ParticipantPropertyUpdateFrequency updateFrequency)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
ParticipantPropertyUpdateFrequency	updateFrequency	

Class EasyMessages

Inheritance

System.Object
EasyMessages

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyMessages
```

Constructors

EasyMessages(EasyEventsAsync, EasyEvents)

Declaration

```
public EasyMessages(EasyEventsAsync eventsAsync, EasyEvents events)
```

Parameters

TYPE	NAME	DESCRIPTION
EasyEventsAsync	eventsAsync	
EasyEvents	events	

Methods

OnChannelMessageRecieved(Object, QueueItemAddedEventArgs<IChannelTextMessage>)

Declaration

```
public void OnChannelMessageRecieved(object sender, QueueItemAddedEventArgs<IChannelTextMessage>
channelMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Object	sender	
QueueItemAddedEventArgs<IChannelTextMessage>	channelMessage	

OnDirectMessageFailedCallback(Object, QueueItemAddedEventArgs<IFailedDirectedTextMessage>)

Declaration

```
public void OnDirectMessageFailedCallback(object sender, QueueItemAddedEventArgs<IFailedDirectedTextMessage>
failedMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Object	sender	
QueueItemAddedEventArgs<IFailedDirectedTextMessage>	failedMessage	

OnDirectMessageRecieved(Object, QueueItemAddedEventArgs<IDirectedTextMessage>)

Declaration

```
public void OnDirectMessageRecieved(object sender, QueueItemAddedEventArgs<IDirectedTextMessage>
directMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Object	sender	
QueueItemAddedEventArgs<IDirectedTextMessage>	directMessage	

SendChannelMessage(IChannelSession, String)

Declaration

```
public void SendChannelMessage(IChannelSession channel, string inputMsg)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channel	
System.String	inputMsg	

SendChannelMessage(IChannelSession, String, String, String)

Declaration

```
public void SendChannelMessage(IChannelSession channel, string inputMsg, string header, string body)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channel	
System.String	inputMsg	
System.String	header	
System.String	body	

SendChannelMessage<T>(IChannelSession, String, T)

Declaration

```
public void SendChannelMessage<T>(IChannelSession channel, string inputMsg, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channel	
System.String	inputMsg	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

SendChannelMessage<T>(IChannelSession, String, T, String, String)

Declaration

```
public void SendChannelMessage<T>(IChannelSession channel, string inputMsg, T value, string header, string
body)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channel	
System.String	inputMsg	
T	value	
System.String	header	
System.String	body	

Type Parameters

NAME	DESCRIPTION
T	

SendDirectMessage(ILoginSession, Dictionary<String, String>, String, String, String, String)

Declaration

```
public void SendDirectMessage(ILoginSession login, Dictionary<string, string> attemptedDirectMessages, string
targetID, string message, string header = null, string body = null)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	login	
Dictionary<System.String, System.String>	attemptedDirectMessages	

TYPE	NAME	DESCRIPTION
System.String	targetID	
System.String	message	
System.String	header	
System.String	body	

SendDirectMessage(ILoginSession, String, String, String, String)

Declaration

```
public void SendDirectMessage(ILoginSession loginSession, string targetID, string message, string header =
null, string body = null)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
System.String	targetID	
System.String	message	
System.String	header	
System.String	body	

SendDirectMessage<T>(ILoginSession, Dictionary<String, String>, String, String, T, String, String)

Declaration

```
public void SendDirectMessage<T>(ILoginSession login, Dictionary<string, string> attemptedDirectMessages,
string targetID, string message, T value, string header = null, string body = null)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	login	
Dictionary<System.String, System.String>	attemptedDirectMessages	
System.String	targetID	
System.String	message	
T	value	
System.String	header	
System.String	body	

Type Parameters

NAME	DESCRIPTION
T	

SendDirectMessage<T>(ILoginSession, String, String, T, String, String)

Declaration

```
public void SendDirectMessage<T>(ILoginSession loginSession, string targetID, string message, T value, string
header = null, string body = null)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
System.String	targetID	
System.String	message	
T	value	
System.String	header	
System.String	body	

Type Parameters

NAME	DESCRIPTION
T	

SendEventMessage(IChannelSession, String, String, String)

Declaration

```
public void SendEventMessage(IChannelSession channel, string eventMessage, string header, string body)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channel	
System.String	eventMessage	
System.String	header	
System.String	body	

SubscribeToChannelMessages(IChannelSession)

Declaration

```
public void SubscribeToChannelMessages(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

SubscribeToDirectMessages(ILoginSession)

Declaration

```
public void SubscribeToDirectMessages(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

UnsubscribeFromChannelMessages(IChannelSession)

Declaration

```
public void UnsubscribeFromChannelMessages(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

UnsubscribeFromDirectMessages(ILoginSession)

Declaration

```
public void UnsubscribeFromDirectMessages(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

Class EasyMute

Inheritance

System.Object
EasyMute

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyMute
```

Constructors

EasyMute(EasyEvents, EasyEventsAsync, EasySettingsSO)

Declaration

```
public EasyMute(EasyEvents events, EasyEventsAsync eventsAync, EasySettingsSO settings)
```

Parameters

TYPE	NAME	DESCRIPTION
EasyEvents	events	
EasyEventsAsync	eventsAync	
EasySettingsSO	settings	

Methods

ClearAllCurrentCrossMutedAccounts(String)

Declaration

```
public void ClearAllCurrentCrossMutedAccounts(string loggedInUserName)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	loggedInUserName	

CrossMuteUser(String, String, String, Boolean)

Declaration

```
public void CrossMuteUser(string userName, string channelName, string userToMute, bool mute)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
System.String	channelName	
System.String	userToMute	
System.Boolean	mute	

CrossMuteUsers(String, String, List<String>, Boolean)

Declaration

```
public void CrossMuteUsers(string loggedInUserName, string channelName, List<string> usersToMute, bool mute)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	loggedInUserName	
System.String	channelName	
List<System.String>	usersToMute	
System.Boolean	mute	

LocalMuteAllUsers(IChannelSession)

Declaration

```
public void LocalMuteAllUsers(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

LocalMuteRemoteUser(String, IChannelSession, Boolean)

Declaration

```
public void LocalMuteRemoteUser(string userName, IChannelSession channelSession, bool mute)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	userName	
IChannelSession	channelSession	
System.Boolean	mute	

LocalMuteSelf(VivoxUnity.Client)

Declaration

```
public void LocalMuteSelf(VivoxUnity.Client client)
```

Parameters

TYPE	NAME	DESCRIPTION
VivoxUnity.Client	client	

LocalMuteSelf<T>(VivoxUnity.Client, T)

Declaration

```
public void LocalMuteSelf<T>(VivoxUnity.Client client, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
VivoxUnity.Client	client	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

LocalUnmuteAllUsers(IChannelSession)

Declaration

```
public void LocalUnmuteAllUsers(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

LocalUnmuteSelf(VivoxUnity.Client)

Declaration

```
public void LocalUnmuteSelf(VivoxUnity.Client client)
```

Parameters

TYPE	NAME	DESCRIPTION
VivoxUnity.Client	client	

LocalUnmuteSelf<T>(VivoxUnity.Client, T)

Declaration

```
public void LocalUnmuteSelf<T>(VivoxUnity.Client client, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
VivoxUnity.Client	client	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

Subscribe(ILoginSession)

Declaration

```
public void Subscribe(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

Unsubscribe(ILoginSession)

Declaration

```
public void Unsubscribe(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

Class EasySession

Inheritance

System.Object
EasySession

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public static class EasySession
```

Fields

ChannelSessions

Declaration

```
public static Dictionary<string, IChannelSession> ChannelSessions
```

Field Value

TYPE	DESCRIPTION
Dictionary<System.String, IChannelSession>	

Client

Declaration

```
public static VivoxUnity.Client Client
```

Field Value

TYPE	DESCRIPTION
VivoxUnity.Client	

LoginSessions

Declaration

```
public static Dictionary<string, ILoginSession> LoginSessions
```

Field Value

TYPE	DESCRIPTION
Dictionary<System.String, ILoginSession>	

Properties

APIEndpoint

Declaration

```
public static Uri APIEndpoint { get; set; }
```

Property Value

TYPE	DESCRIPTION
Uri	

Domain

Declaration

```
public static string Domain { get; set; }
```

Property Value

TYPE	DESCRIPTION
System.String	

Issuer

Declaration

```
public static string Issuer { get; set; }
```

Property Value

TYPE	DESCRIPTION
System.String	

SecretKey

Declaration

```
public static string SecretKey { get; set; }
```

Property Value

TYPE	DESCRIPTION
System.String	

UniqueCounter

Declaration

```
public static int UniqueCounter { get; }
```

Property Value

TYPE	DESCRIPTION
System.Int32	

UseDynamicEvents

Declaration

```
public static bool UseDynamicEvents { get; set; }
```

Property Value

TYPE	DESCRIPTION
System.Boolean	

Class EasySIP

Inheritance

System.Object
EasySIP

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class EasySIP : MonoBehaviour
```

Methods

GetChannelSIP(ChannelType, String, String, String, Channel3DProperties)

Gets valid Vivox Channel SIP address

Declaration

```
public static string GetChannelSIP(ChannelType channelType, string issuer, string channelName, string domain,
Channel3DProperties channel3DProperties = null)
```

Parameters

TYPE	NAME	DESCRIPTION
ChannelType	channelType	
System.String	issuer	
System.String	channelName	
System.String	domain	
Channel3DProperties	channel3DProperties	

Returns

TYPE	DESCRIPTION
System.String	

GetChannelSIP(ChannelType, String, String, String, String, String, Channel3DProperties)

Gets valid Vivox Channel SIP address

Declaration

```
public static string GetChannelSIP(ChannelType channelType, string issuer, string channelName, string domain,
string region, string hash, Channel3DProperties channel3DProperties = null)
```

Parameters

TYPE	NAME	DESCRIPTION
ChannelType	channelType	
System.String	issuer	

TYPE	NAME	DESCRIPTION
System.String	channelName	
System.String	domain	
System.String	region	
System.String	hash	
Channel3DProperties	channel3DProperties	

Returns

TYPE	DESCRIPTION
System.String	

GetChannelSip(IChannelSession, Channel3DProperties)

Gets valid Vivox Channel SIP address

Declaration

```
public static string GetChannelSip(IChannelSession channelSession, Channel3DProperties channel3DProperties =
null)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
Channel3DProperties	channel3DProperties	

Returns

TYPE	DESCRIPTION
System.String	

GetUserSIP(ILoginSession)

Gets valid Vivox SIP address

Declaration

```
public static string GetUserSIP(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

Returns

TYPE	DESCRIPTION
System.String	

GetUserSIP(IParticipant)

Gets valid Vivox SIP address

Declaration

```
public static string GetUserSIP(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

Returns

TYPE	DESCRIPTION
System.String	

GetUserSIP(String, String, String)

Gets valid Vivox SIP address

Declaration

```
public static string GetUserSIP(string issuer, string userName, string domain)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	issuer	
System.String	userName	
System.String	domain	

Returns

TYPE	DESCRIPTION
System.String	

Class EasyTextChannel

Inheritance

System.Object
EasyTextChannel

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyTextChannel
```

Constructors

EasyTextChannel(EasyEventsAsync, EasyEvents)

Declaration

```
public EasyTextChannel(EasyEventsAsync eventsAsync, EasyEvents events)
```

Parameters

TYPE	NAME	DESCRIPTION
EasyEventsAsync	eventsAsync	
EasyEvents	events	

Methods

OnChannelTextPropertyChanged(Object, PropertyChangedEventArgs)

Declaration

```
public void OnChannelTextPropertyChanged(object sender, PropertyChangedEventArgs propArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Object	sender	
PropertyChangedEventArgs	propArgs	

Subscribe(IChannelSession)

Declaration

```
public void Subscribe(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

ToggleTextChannelActive<T>(IChannelSession, Boolean, T)

Declaration

```
public void ToggleTextChannelActive<T>(IChannelSession channelSession, bool join, T eventParameter)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
System.Boolean	join	
T	eventParameter	

Type Parameters

NAME	DESCRIPTION
T	

ToggleTextInChannel(IChannelSession, Boolean)

Declaration

```
public void ToggleTextInChannel(IChannelSession channelSession, bool join)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
System.Boolean	join	

Unsubscribe(IChannelSession)

Declaration

```
public void Unsubscribe(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Class EasyTextToSpeech

Inheritance

System.Object
EasyTextToSpeech

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyTextToSpeech
```

Constructors

EasyTextToSpeech(EasyEvents, EasyEventsAsync)

Declaration

```
public EasyTextToSpeech(EasyEvents events, EasyEventsAsync eventsAsync)
```

Parameters

TYPE	NAME	DESCRIPTION
EasyEvents	events	
EasyEventsAsync	eventsAsync	

Properties

FemaleVoice

Declaration

```
public string FemaleVoice { get; }
```

Property Value

TYPE	DESCRIPTION
System.String	

MaleVoice

Declaration

```
public string MaleVoice { get; }
```

Property Value

TYPE	DESCRIPTION
System.String	

Methods

ChooseVoiceGender(VoiceGender, String)

Declaration

```
public void ChooseVoiceGender(VoiceGender voiceGender, string userName)
```

Parameters

TYPE	NAME	DESCRIPTION
VoiceGender	voiceGender	
System.String	userName	

PlayTTSMessage(String, TTSDestination, ILoginSession)

Declaration

```
public void PlayTTSMessage(string message, TTSDestination destination, ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	message	
TTSDestination	destination	
ILoginSession	loginSession	

Subscribe(ILoginSession)

Declaration

```
public void Subscribe(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

Unsubscribe(ILoginSession)

Declaration

```
public void Unsubscribe(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

Class EasyUsers

Inheritance

System.Object
EasyUsers

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyUsers
```

Constructors

EasyUsers(EasyEvents, EasyEventsAsync)

Declaration

```
public EasyUsers(EasyEvents events, EasyEventsAsync eventsAsync)
```

Parameters

TYPE	NAME	DESCRIPTION
EasyEvents	events	
EasyEventsAsync	eventsAsync	

Methods

OnUserJoinedChannel(Object, KeyEventArg<String>)

Declaration

```
public void OnUserJoinedChannel(object sender, KeyEventArg<string> keyArg)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Object	sender	
KeyEventArg<System.String>	keyArg	

OnUserLeftChannel(Object, KeyEventArg<String>)

Declaration

```
public void OnUserLeftChannel(object sender, KeyEventArg<string> keyArg)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Object	sender	
KeyEventArg<System.String>	keyArg	

OnUserValuesUpdated(Object, ValueEventArg<String, IParticipant>)

Declaration

```
public void OnUserValuesUpdated(object sender, ValueEventArg<string, IParticipant> valueArg)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Object	sender	
ValueEventArg<System.String, IParticipant>	valueArg	

SubscribeToParticipantEvents(IChannelSession)

Declaration

```
public void SubscribeToParticipantEvents(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

UnsubscribeFromParticipantEvents(IChannelSession)

Declaration

```
public void UnsubscribeFromParticipantEvents(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Class NetCode3DPositional

Inheritance

System.Object
NetCode3DPositional

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public class NetCode3DPositional : MonoBehaviour
```

Fields

listenerPosition

Declaration

```
public Transform listenerPosition
```

Field Value

TYPE	DESCRIPTION
Transform	

speakerPosition

Declaration

```
public Transform speakerPosition
```

Field Value

TYPE	DESCRIPTION
Transform	

Methods

CheckIfChannelExists()

Declaration

```
public bool CheckIfChannelExists()
```

Returns

TYPE	DESCRIPTION
System.Boolean	

Update3DPosition()

Declaration

```
public void Update3DPosition()
```

Enum VoiceGender

Namespace: EasyCodeForVivox

Assembly: cs.temp.dll.dll

Syntax

```
public enum VoiceGender
```

Fields

NAME	DESCRIPTION
female	
male	

Namespace EasyCodeForVivox.DemoScene

Structs

PlayerInfo

Struct PlayerInfo

Implements

INetworkSerializable
IEquatable<PlayerInfo>

Inherited Members

System.ValueType.Equals(System.Object)
System.ValueType.GetHashCode()
System.ValueType.ToString()
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetType()

Namespace: EasyCodeForVivox.DemoScene

Assembly: cs.temp.dll.dll

Syntax

```
public struct PlayerInfo : INetworkSerializable, IEquatable<PlayerInfo>
```

Fields

playerId

Declaration

```
public ulong playerId
```

Field Value

TYPE	DESCRIPTION
System.UInt64	

playerName

Declaration

```
public FixedString32Bytes playerName
```

Field Value

TYPE	DESCRIPTION
FixedString32Bytes	

Methods

Equals(PlayerInfo)

Declaration

```
public bool Equals(PlayerInfo other)
```

Parameters

TYPE	NAME	DESCRIPTION
PlayerInfo	other	

Returns

TYPE	DESCRIPTION
System.Boolean	

NetworkSerialize<T>(BufferSerializer<T>)

Declaration

```
public void NetworkSerialize<T>(BufferSerializer<T> serializer)
    where T : IReaderWriter
```

Parameters

TYPE	NAME	DESCRIPTION
BufferSerializer<T>	serializer	

Type Parameters

NAME	DESCRIPTION
T	

Implements

INetworkSerializable
IEquatable<>

Namespace EasyCodeForVivox.Events

Classes

AudioChannelEventAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox Audio Channel Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen AudioChannelStatus event happens

```
[AudioChannelEventAsync(AudioChannelStatus.AudioChannelConnecting)]
private async void OnAudioChannelConnectingAsync(IChannelSession channelSession)
{
Debug.Log($"{channelSession.Channel.Name} Is Connecting");
await LoadPlayerData();
}
```

Check out the Docs Dynamic Async Events - Audio Channel Events

AudioChannelEventAttribute

Place this on a method to subscribe to Vivox Audio Channel Events

Method will be called when chosen AudioChannelStatus event happens

Example Method

```
[AudioChannelEvent(AudioChannelStatus.AudioChannelConnecting)]
private void OnAudioChannelConnecting(IChannelSession channelSession)
{
    Debug.Log($"{channelSession.Channel.Name} Is Connecting");
}
```

Check out the Docs Audio Channel Events

AudioDeviceEventAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox Audio Device Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen AudioDeviceStatus event happens

```
[AudioDeviceEventAsync(AudioDeviceStatus.AudioInputDeviceAdded)]
private async void OnAudioInputDeviceAddedAsync(IAudioDevice audioDevice)
{
Debug.Log($"Audio Input device has been added {audioDevice?.Name}");
await SavePlayerData();
}
```

Check out the Docs Dynamic Async Events - Audio Device Events

AudioDeviceEventAttribute

Place this on a method to subscribe to Vivox Audio Device Events

Method will be called when chosen AudioDeviceStatus event happens

Example Method

```
[AudioDeviceEvent(AudioDeviceStatus.AudioInputDeviceAdded)]
private void OnAudioInputDeviceAdded(IAudioDevice audioDevice)
{
    Debug.Log($"Audio Input device has been added {audioDevice?.Name}");
}
```

Check out the Docs Audio Device Events

ChannelEventAsyncAttribute

Place this on an async void or async Task method to subscribe to Vivox Channel Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen ChannelStatus event happens

```
[ChannelEventAsync(ChannelStatus.Connected)]
private async void OnChannelConnectedAsync(IChannelSession channelSession)
{
    Debug.Log($"{channelSession.Channel.Name} Is Connecting");
    await LoadPlayerData();
}
```

Check out the Docs Dynamic Async Events - Channel Events

ChannelEventAttribute

Place this on a method to subscribe to Vivox Channel Events

Method will be called when chosen ChannelStatus event happens

Example Method

```
[ChannelEvent(ChannelStatus.ChannelConnected)]
private void OnChannelConnected(IChannelSession channelSession)
{
    Debug.Log($"{channelSession.Channel.Name} Has Connected : Channel Type ==
{channelSession.Channel.Type}");
}
```

Check out the Docs Channel Events

ChannelMessageEventAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox Channel Message Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen ChannelMessageStatus event happens

```
[ChannelMessageEventAsync(ChannelMessageStatus.ChannelMessageRecieved)]
private async void OnChannelMessageRecievedAsync(IChannelTextMessage textMessage)
{
Debug.Log($"From {textMessage.Sender.DisplayName} : {textMessage.ReceivedTime} : {textMessage.Message}");
await SavePlayerData();
}
```

Check out the Docs Dynamic Async Events - Channel Message Events

ChannelMessageEventAttribute

Place this on a method to subscribe to Vivox Channel Message Events

Place this on a method to subscribe to Vivox Channel Message Events

Method will be called when chosen ChannelMessageStatus event happens

Example Method

```
[ChannelMessageEvent(ChannelMessageStatus.ChannelMessageRecieved)]
private void OnChannelMessageRecieved(IChannelTextMessage textMessage)
{
    Debug.Log($"From {textMessage.Sender.DisplayName} : {textMessage.ReceivedTime} : {textMessage.Message}");
}
```

Check out the Docs Channel Message Events

DirectMessageEventAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox Direct Message Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen DirectMessageStatus event happens

```
[DirectMessageEventAsync(DirectMessageStatus.DirectMessageRecieved)]
private async void OnDirectMessageRecievedAsync(IDirectedTextMessage directedTextMessage)
{
Debug.Log($"Recived Message From : {directedTextMessage.Sender.DisplayName} :
{directedTextMessage.ReceivedTime} : {directedTextMessage.Message}");
await SavePlayerData();
}
```

Check out the Docs Dynamic Async Events - Direct Message Events

DirectMessageEventAttribute

Place this on a method to subscribe to Vivox Direct Message Events

Method will be called when chosen DirectMessageStatus event happens

Example Method

```
[DirectMessageEvent(DirectMessageStatus.DirectMessageRecieved)]
private void OnDirectMessageRecieved(IDirectedTextMessage directedTextMessage)
{
    Debug.Log($"Recived Message From : {directedTextMessage.Sender.DisplayName} :
{directedTextMessage.ReceivedTime} : {directedTextMessage.Message}");
}
```

Check out the Docs Directed Message Events

EasyEvents

HandleDynamicEvents

LoginEventAsyncAttribute

Place this on an async void or async Task method to subscribe to Vivox Login Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen LoginStatus event happens

```
[LoginEventAsync(LoginStatus.LoggingIn)]
private async void OnPlayerLoggingInAsync(ILoginSession loginSession)
{
    Debug.Log($"Logging In : {loginSession.LoginSessionId.DisplayName}");
    await GetJoinedLobbies();
}
```

Check out the Docs Dynamic Async Events - Login Events

LoginEventAttribute

Place this on a method to subscribe to Vivox Login Events

Method will be called when chosen LoginStatus event happens

```
[LoginEvent(LoginStatus.LoggedIn)]"
public void UserLoggedIn(ILoginSession loginSession)
{
    $"Logged In {loginSession.LoginSessionId.DisplayName}";
}
```

Check out the Docs Login Events

TextChannelEventAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox Text Channel Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen TextChannelStatus event happens

```
[TextChannelEventAsync(TextChannelStatus.TextChannelConnecting)]
private async void OnTextChannelConnectingAsync(IChannelSession channelSession)
{
Debug.Log($"{channelSession.Channel.Name} Is Connecting");
await LoadPlayerData();
}
```

Check out the Docs Dynamic Async Events - Text Channel Events

TextChannelEventAttribute

Place this on a method to subscribe to Vivox Text Channel Events

Method will be called when chosen TextChannelStatus event happens

Example Method

```
[TextChannelEvent(TextChannelStatus.TextChannelConnecting)]
private void OnTextChannelConnecting(IChannelSession channelSession)
{
    Debug.Log($"{channelSession.Channel.Name} Is Connecting");
}
```

Check out the Docs Text Channel Events

TextToSpeechEventAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox Text-To-Speech Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen TextToSpeechStatus event happens

```
[TextToSpeechEventAsync(TextToSpeechStatus.TTSMessageAdded)]
private async void  OnTTSMessageAddedAsync(ITTSMessageQueueEventArgs ttsArgs)
{
Debug.Log($"TTS Message Has Been Added : {ttsArgs.Message.Text}");
await SavePlayerData();
}
```

Check out the Docs Dynamic Async Events - Text To Speech Events

TextToSpeechEventAttribute

Place this on a method to subscribe to Vivox Text-To-Speech Events

Method will be called when chosen TextToSpeechStatus event happens

Example Method

```
[TextToSpeechEvent(TextToSpeechStatus.TTSMessageAdded)]
private void OnTTSMessageAdded(ITTSMessageQueueEventArgs ttsArgs)
{
    Debug.Log($"TTS Message Has Been Added : {ttsArgs.Message.Text}");
}
```

Check out the Docs Text To Speech Events

UserEventsAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox User Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen UserStatus event happens

```
[UserEventsAsync(UserStatus.LocalUserMuted)]
private async void OnLocalUserMutedAsync()
{
Debug.Log("Local User is Muted");
await SavePlayerData();
}
```

Check out the Docs Dynamic Async Events - User Events

UserEventsAttribute

Place this on a method to subscribe to Vivox User Events

Online Docs (Dynamic Events - User Participant Events)

Method will be called when chosen UserStatus event happens

Example Method

```
[UserEvents(UserStatus.UserMuted)]
private void OnUserMuted(IParticipant participant)
{
    Debug.Log($"{participant.Account.DisplayName} Is Muted : (Muted For All : {participant.IsMutedForAll})");
}
```

Check out the Docs User Events

Enums

AudioChannelStatus

The type of **Vivox Audio Channel Event** that you want this method to be subscribed to. Works for AudioChannelEventAttribute and AudioChannelEventAsyncAttribute.

AudioDeviceStatus

The type of **Vivox Audio Device Event** that you want this method to be subscribed to. Works for AudioDeviceEventAttribute and AudioDeviceEventAsyncAttribute.

ChannelMessageStatus

The type of **Vivox Channel Message Event** that you want this method to be subscribed to. Works for ChannelMessageEventAttribute and ChannelMessageEventAsyncAttribute.

ChannelStatus

The type of **Vivox Channel Event** that you want this method to be subscribed to. Works for ChannelEventAttribute and ChannelEventAsyncAttribute.

DirectMessageStatus

The type of **Vivox Direct Message Event** that you want this method to be subscribed to. Works for DirectMessageEventAttribute and DirectMessageEventAsyncAttribute.

LoginStatus

The type of **Vivox Login Event** you want this method to be subscribed to. Works for LoginEventAttribute and LoginEventAsyncAttribute

TextChannelStatus

The type of **Vivox Text Channel Event** that you want this method to be subscribed to. Works for TextChannelEventAttribute and TextChannelEventAsyncAttribute.

TextToSpeechStatus

The type of **Vivox Text-to-Speech Event** that you want this method to be subscribed to. Works for TextToSpeechEventAttribute and TextToSpeechEventAsyncAttribute.

UserStatus

The type of **Vivox User Event** that you want this method to be subscribed to. Works for UserEventsAttribute and UserEventsAsyncAttribute.

Class AudioChannelEventAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox Audio Channel Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen AudioChannelStatus event happens

```
[AudioChannelEventAsync(AudioChannelStatus.AudioChannelConnecting)]
private async void OnAudioChannelConnectingAsync(IChannelSession channelSession)
{
Debug.Log($"{channelSession.Channel.Name} Is Connecting");
await LoadPlayerData();
}
```

Check out the Docs Dynamic Async Events - Audio Channel Events

Inheritance

System.Object
AudioChannelEventAsyncAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class AudioChannelEventAsyncAttribute : Attribute
```

Constructors

AudioChannelEventAsyncAttribute(AudioChannelStatus)

EasyCode uses Reflection to find methods that contain AudioChannelEventAsyncAttribute and invokes these methods dynamically with the specified options.

Declaration

```
public AudioChannelEventAsyncAttribute(AudioChannelStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
AudioChannelStatus	options	The status for the Audio Channel event.

Properties

Options

Gets or sets the status for the Audio Channel event.

Declaration

```
public AudioChannelStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
AudioChannelStatus	

Class AudioChannelEventAttribute

Place this on a method to subscribe to Vivox Audio Channel Events

Method will be called when chosen AudioChannelStatus event happens

Example Method

```
[AudioChannelEvent(AudioChannelStatus.AudioChannelConnecting)]
private void OnAudioChannelConnecting(IChannelSession channelSession)
{
    Debug.Log($"{channelSession.Channel.Name} Is Connecting");
}
```

Check out the Docs Audio Channel Events

Inheritance

System.Object
AudioChannelEventAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class AudioChannelEventAttribute : Attribute
```

Constructors

AudioChannelEventAttribute(AudioChannelStatus)

Declaration

```
public AudioChannelEventAttribute(AudioChannelStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
AudioChannelStatus	options	

Properties

Options

Declaration

```
public AudioChannelStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
AudioChannelStatus	

Enum AudioChannelStatus

The type of **Vivox Audio Channel Event** that you want this method to be subscribed to. Works for AudioChannelEventAttribute and AudioChannelEventAsyncAttribute.

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public enum AudioChannelStatus
```

Fields

NAME	DESCRIPTION
AudioChannelConnected	Event is invoked/fired when player has successfully joined a Vivox Audio Channel Will be fired for Echo, Non-Positional, and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelSession
AudioChannelConnecting	Event is invoked/fired when player begins joining a Vivox Audio Channel Will be fired for Echo, Non-Positional, and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelSession
AudioChannelDisconnected	Event is invoked/fired when player has successfully disconnected from a Vivox Audio Channel Will be fired for Echo, Non-Positional, and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelSession
AudioChannelDisconnecting	Event is invoked/fired when player begins disconnecting from a Vivox Audio Channel Will be fired for Echo, Non-Positional, and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelSession

Class AudioDeviceEventAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox Audio Device Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen AudioDeviceStatus event happens

```
[AudioDeviceEventAsync(AudioDeviceStatus.AudioInputDeviceAdded)]
private async void OnAudioInputDeviceAddedAsync(IAudioDevice audioDevice)
{
Debug.Log($"Audio Input device has been added {audioDevice?.Name}");
await SavePlayerData();
}
```

Check out the Docs Dynamic Async Events - Audio Device Events

Inheritance

System.Object
AudioDeviceEventAsyncAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class AudioDeviceEventAsyncAttribute : Attribute
```

Constructors

AudioDeviceEventAsyncAttribute(AudioDeviceStatus)

EasyCode uses Reflection to find methods that contain AudioDeviceEventAsyncAttribute and invokes these methods dynamically with the specified options.

Declaration

```
public AudioDeviceEventAsyncAttribute(AudioDeviceStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
AudioDeviceStatus	options	The status for the Audio Device event.

Properties

Options

Gets or sets the status for the Audio Device event.

Declaration

```
public AudioDeviceStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
AudioDeviceStatus	

Class AudioDeviceEventAttribute

Place this on a method to subscribe to Vivox Audio Device Events

Method will be called when chosen AudioDeviceStatus event happens

Example Method

```
[AudioDeviceEvent(AudioDeviceStatus.AudioInputDeviceAdded)]
private void OnAudioInputDeviceAdded(IAudioDevice audioDevice)
{
    Debug.Log($"Audio Input device has been added {audioDevice?.Name}");
}
```

Check out the Docs Audio Device Events

Inheritance

System.Object
AudioDeviceEventAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class AudioDeviceEventAttribute : Attribute
```

Constructors

AudioDeviceEventAttribute(AudioDeviceStatus)

Declaration

```
public AudioDeviceEventAttribute(AudioDeviceStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
AudioDeviceStatus	options	

Properties

Options

Declaration

```
public AudioDeviceStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
AudioDeviceStatus	

Enum AudioDeviceStatus

The type of **Vivox Audio Device Event** that you want this method to be subscribed to. Works for AudioDeviceEventAttribute and AudioDeviceEventAsyncAttribute.

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public enum AudioDeviceStatus
```

Fields

NAME	DESCRIPTION
AudioInputDeviceAdded	Event is invoked/fired when Vivox detects a new Audio Input Device (Microphone) is connected to your pc/console/device Method must contain only **1** parameter of type VivoxUnity.IAudioDevice
AudioInputDeviceRemoved	Event is invoked/fired when Vivox detects a new Audio Input Device (Microphone) is disconnected from your pc/console/device Method must contain only **1** parameter of type VivoxUnity.IAudioDevice
AudioInputDeviceUpdated	Event is invoked/fired when Vivox detects a new Audio Input Device (Microphone) is updated on your pc/console/device Method must contain only **1** parameter of type VivoxUnity.IAudioDevice
AudioOutputDeviceAdded	Event is invoked/fired when Vivox detects a new Audio Output Device (Speaker/Headphones) is connected to your pc/console/device Method must contain only **1** parameter of type VivoxUnity.IAudioDevice
AudioOutputDeviceRemoved	Event is invoked/fired when Vivox detects a new Audio Output Device (Speaker/Headphones) is disconnected from your pc/console/device Method must contain only **1** parameter of type VivoxUnity.IAudioDevice
AudioOutputDeviceUpdated	Event is invoked/fired when Vivox detects a new Audio Output Device (Speaker/Headphones) is updated on your pc/console/device Method must contain only **1** parameter of type VivoxUnity.IAudioDevice

Class ChannelEventAsyncAttribute

Place this on an async void or async Task method to subscribe to Vivox Channel Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen ChannelStatus event happens

```
[ChannelEventAsync(ChannelStatus.Connected)]
private async void OnChannelConnectedAsync(IChannelSession channelSession)
{
    Debug.Log($"{channelSession.Channel.Name} Is Connecting");
    await LoadPlayerData();
}
```

Check out the Docs Dynamic Async Events - Channel Events

Inheritance

System.Object
ChannelEventAsyncAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class ChannelEventAsyncAttribute : Attribute
```

Constructors

ChannelEventAsyncAttribute(ChannelStatus)

EasyCode uses Reflection to find methods that contain ChannelEventAsyncAttribute and subscribes these methods to Channel event. EasyCode then invokes these methods dynamically with the specified options.

Declaration

```
public ChannelEventAsyncAttribute(ChannelStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
ChannelStatus	options	The Channel event status you want to subscribe to.

Properties

Options

Gets or sets the status for the Channel event.

Declaration

```
public ChannelStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
ChannelStatus	

Class ChannelEventAttribute

Place this on a method to subscribe to Vivox Channel Events

Method will be called when chosen ChannelStatus event happens

Example Method

```
[ChannelEvent(ChannelStatus.ChannelConnected)]
private void OnChannelConnected(IChannelSession channelSession)
{
    Debug.Log($"{channelSession.Channel.Name} Has Connected : Channel Type == {channelSession.Channel.Type}");
}
```

Check out the Docs Channel Events

Inheritance

System.Object
ChannelEventAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class ChannelEventAttribute : Attribute
```

Constructors

ChannelEventAttribute(ChannelStatus)

Declaration

```
public ChannelEventAttribute(ChannelStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
ChannelStatus	options	

Properties

Options

Declaration

```
public ChannelStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
ChannelStatus	

Class ChannelMessageEventAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox Channel Message Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen ChannelMessageStatus event happens

```
[ChannelMessageEventAsync(ChannelMessageStatus.ChannelMessageRecieved)]
private async void OnChannelMessageRecievedAsync(IChannelTextMessage textMessage)
{
Debug.Log($"From {textMessage.Sender.DisplayName} : {textMessage.ReceivedTime} : {textMessage.Message}");
await SavePlayerData();
}
```

Check out the Docs Dynamic Async Events - Channel Message Events

Inheritance

System.Object
ChannelMessageEventAsyncAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class ChannelMessageEventAsyncAttribute : Attribute
```

Constructors

ChannelMessageEventAsyncAttribute(ChannelMessageStatus)

EasyCode uses Reflection to find methods that contain ChannelMessageEventAsyncAttribute and invokes these methods dynamically with the specified options.

Declaration

```
public ChannelMessageEventAsyncAttribute(ChannelMessageStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
ChannelMessageStatus	options	The status for the Channel Message event.

Properties

Options

Gets or sets the status for the Channel Message event.

Declaration

```
public ChannelMessageStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
ChannelMessageStatus	

Class ChannelMessageEventAttribute

Place this on a method to subscribe to Vivox Channel Message Events

Method will be called when chosen ChannelMessageStatus event happens

Example Method

```
[ChannelMessageEvent(ChannelMessageStatus.ChannelMessageRecieved)]
private void OnChannelMessageRecieved(IChannelTextMessage textMessage)
{
    Debug.Log($"From {textMessage.Sender.DisplayName} : {textMessage.ReceivedTime} : {textMessage.Message}");
}
```

Check out the Docs Channel Message Events

Inheritance

System.Object

ChannelMessageEventAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class ChannelMessageEventAttribute : Attribute
```

Constructors

ChannelMessageEventAttribute(ChannelMessageStatus)

Declaration

```
public ChannelMessageEventAttribute(ChannelMessageStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
ChannelMessageStatus	options	

Properties

Options

Declaration

```
public ChannelMessageStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
ChannelMessageStatus	

Enum ChannelMessageStatus

The type of **Vivox Channel Message Event** that you want this method to be subscribed to. Works for ChannelMessageEventAttribute and ChannelMessageEventAsyncAttribute.

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public enum ChannelMessageStatus
```

Fields

NAME	DESCRIPTION
ChannelMessageRecieved	Event is invoked/fired when player recieves a message from a connected Vivox Text Channel Will be fired for Non-Positional and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelTextMessage
ChannelMessageSent	Event is invoked/fired when player sends a message from a connected Vivox Text Channel Will be fired for Non-Positional and 3D Positional channels Method must have **0** parameters
EventMessageRecieved	Event is invoked/fired when developer wants to send a secret message in a connected Vivox Text Channel that players wont see If using a networking stack like NetCodeForGameObjects it is better to send a message with NetCode than with Vivox Will be fired for Non-Positional and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelTextMessage

Enum ChannelStatus

The type of **Vivox Channel Event** that you want this method to be subscribed to. Works for ChannelEventAttribute and ChannelEventAsyncAttribute.

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public enum ChannelStatus
```

Fields

NAME	DESCRIPTION
ChannelConnected	Event is invoked/fired when player has successfully joined a Vivox Channel Will be fired for Echo, Non-Positional, and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelSession
ChannelConnecting	Event is invoked/fired when player begins joining a Vivox Channel Will be fired for Echo, Non-Positional, and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelSession
ChannelDisconnected	Event is invoked/fired when player has successfully disconnected from a Vivox Channel Will be fired for Echo, Non-Positional, and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelSession
ChannelDisconnecting	Event is invoked/fired when player begins disconnecting from a Vivox Channel Will be fired for Echo, Non-Positional, and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelSession

Class DirectMessageEventAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox Direct Message Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen DirectMessageStatus event happens

```
[DirectMessageEventAsync(DirectMessageStatus.DirectMessageRecieved)]
private async void OnDirectMessageRecievedAsync(IDirectedTextMessage directedTextMessage)
{
Debug.Log($"Recived Message From : {directedTextMessage.Sender.DisplayName} :
{directedTextMessage.ReceivedTime} : {directedTextMessage.Message}");
await SavePlayerData();
}
```

Check out the Docs Dynamic Async Events - Direct Message Events

Inheritance

System.Object

DirectMessageEventAsyncAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class DirectMessageEventAsyncAttribute : Attribute
```

Constructors

DirectMessageEventAsyncAttribute(DirectMessageStatus)

EasyCode uses Reflection to find methods that contain DirectMessageEventAsyncAttribute and invokes these methods dynamically with the specified options.

Declaration

```
public DirectMessageEventAsyncAttribute(DirectMessageStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
DirectMessageStatus	options	The status for the Direct Message event.

Properties

Options

Gets or sets the status for the Direct Message event.

Declaration

```
public DirectMessageStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
DirectMessageStatus	

TYPE	DESCRIPTION

Class DirectMessageEventAttribute

Place this on a method to subscribe to Vivox Direct Message Events

Method will be called when chosen DirectMessageStatus event happens

Example Method

```
[DirectMessageEvent(DirectMessageStatus.DirectMessageRecieved)]
private void OnDirectMessageRecieved(IDirectedTextMessage directedTextMessage)
{
    Debug.Log($"Recived Message From : {directedTextMessage.Sender.DisplayName} :
{directedTextMessage.ReceivedTime} : {directedTextMessage.Message}");
}
```

Check out the Docs Directed Message Events

Inheritance

System.Object

DirectMessageEventAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class DirectMessageEventAttribute : Attribute
```

Constructors

DirectMessageEventAttribute(DirectMessageStatus)

Declaration

```
public DirectMessageEventAttribute(DirectMessageStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
DirectMessageStatus	options	

Properties

Options

Declaration

```
public DirectMessageStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
DirectMessageStatus	

Enum DirectMessageStatus

The type of **Vivox Direct Message Event** that you want this method to be subscribed to. Works for DirectMessageEventAttribute and DirectMessageEventAsyncAttribute.

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public enum DirectMessageStatus
```

Fields

NAME	DESCRIPTION
DirectMessageFailed	Event is invoked/fired when player sends a direct message (DM) to another Vivox user who is **not logged in**. Vivox treats the message as failed and it is up to the developer to implement retries (sending the message again) or storing the failed message on the player's computer in a SQLite Database, PlayerPrefs, or in a txt/json file. You can also upload to the cloud using Unity's Cloud Save, AWS S3, or Database of your choice. Method must contain only **1** parameter of type VivoxUnity.IFailedDirectedTextMessage
DirectMessageRecieved	Event is invoked/fired when player sends a direct message (DM) to another Vivox user who is logged in Method must have **0** parameters
DirectMessageSent	Event is invoked/fired when player recieves a direct message (DM) from another Vivox user who is logged in Method must contain only **1** parameter of type VivoxUnity.IDirectedTextMessage

Class EasyEvents

Inheritance

System.Object
EasyEvents

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyEvents
```

Constructors

EasyEvents(EasySettingsSO)

Declaration

```
public EasyEvents(EasySettingsSO settings)
```

Parameters

TYPE	NAME	DESCRIPTION
EasySettingsSO	settings	

Methods

CreateDelegateAndInvoke<T1, T2>(Enum, T1, T2)

Declaration

```
public void CreateDelegateAndInvoke<T1, T2>(Enum eventKey, T1 value1, T2 value2)
```

Parameters

TYPE	NAME	DESCRIPTION
Enum	eventKey	
T1	value1	
T2	value2	

Type Parameters

NAME	DESCRIPTION
T1	

NAME	DESCRIPTION
T2	

InvokeMethods(Enum)

Declaration

```
public void InvokeMethods(Enum eventKey)
```

Parameters

TYPE	NAME	DESCRIPTION
Enum	eventKey	

InvokeMethods<T>(Enum, T)

Declaration

```
public void InvokeMethods<T>(Enum eventKey, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
Enum	eventKey	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

InvokeMethods<T1, T2>(Enum, T1, T2)

Declaration

```
public void InvokeMethods<T1, T2>(Enum eventKey, T1 value1, T2 value2)
```

Parameters

TYPE	NAME	DESCRIPTION
Enum	eventKey	
T1	value1	
T2	value2	

Type Parameters

NAME	DESCRIPTION
T1	

NAME	DESCRIPTION
T2	

OnAudioChannelConnected(IChannelSession)

Declaration

```
public void OnAudioChannelConnected(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnAudioChannelConnected<T>(IChannelSession, T)

Declaration

```
public void OnAudioChannelConnected<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnAudioChannelConnecting(IChannelSession)

Declaration

```
public void OnAudioChannelConnecting(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnAudioChannelConnecting<T>(IChannelSession, T)

Declaration

```
public void OnAudioChannelConnecting<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

TYPE	NAME	DESCRIPTION
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnAudioChannelDisconnected(IChannelSession)

Declaration

```
public void OnAudioChannelDisconnected(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnAudioChannelDisconnected<T>(IChannelSession, T)

Declaration

```
public void OnAudioChannelDisconnected<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnAudioChannelDisconnecting(IChannelSession)

Declaration

```
public void OnAudioChannelDisconnecting(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnAudioChannelDisconnecting<T>(IChannelSession, T)

Declaration

```
public void OnAudioChannelDisconnecting<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnAudioInputDeviceAdded(IAudioDevice)

Declaration

```
public void OnAudioInputDeviceAdded(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioInputDeviceRemoved(IAudioDevice)

Declaration

```
public void OnAudioInputDeviceRemoved(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioInputDeviceUpdated(IAudioDevice)

Declaration

```
public void OnAudioInputDeviceUpdated(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioOutputDeviceAdded(IAudioDevice)

Declaration

```
public void OnAudioOutputDeviceAdded(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioOutputDeviceRemoved(IAudioDevice)

Declaration

```
public void OnAudioOutputDeviceRemoved(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioOutputDeviceUpdated(IAudioDevice)

Declaration

```
public void OnAudioOutputDeviceUpdated(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnChannelConnected(IChannelSession)

Declaration

```
public void OnChannelConnected(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnChannelConnected<T>(IChannelSession, T)

Declaration

```
public void OnChannelConnected<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnChannelConnecting(IChannelSession)

Declaration

```
public void OnChannelConnecting(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnChannelConnecting<T>(IChannelSession, T)

Declaration

```
public void OnChannelConnecting<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnChannelDisconnected(IChannelSession)

Declaration

```
public void OnChannelDisconnected(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnChannelDisconnected<T>(IChannelSession, T)

Declaration

```
public void OnChannelDisconnected<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnChannelDisconnecting(IChannelSession)

Declaration

```
public void OnChannelDisconnecting(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnChannelDisconnecting<T>(IChannelSession, T)

Declaration

```
public void OnChannelDisconnecting<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnChannelMessageRecieved(IChannelTextMessage)

Declaration

```
public void OnChannelMessageRecieved(IChannelTextMessage channelTextMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelTextMessage	channelTextMessage	

OnChannelMessageRecieved<T>(IChannelTextMessage, T)

Declaration

```
public void OnChannelMessageRecieved<T>(IChannelTextMessage channelTextMessage, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelTextMessage	channelTextMessage	

TYPE	NAME	DESCRIPTION
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnChannelMessageSent()

Declaration

```
public void OnChannelMessageSent()
```

OnChannelMessageSent<T>(T)

Declaration

```
public void OnChannelMessageSent<T>(T value)
```

Parameters

TYPE	NAME	DESCRIPTION
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnDirectMessageFailed(IFailedDirectedTextMessage)

Declaration

```
public void OnDirectMessageFailed(IFailedDirectedTextMessage failedMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
IFailedDirectedTextMessage	failedMessage	

OnDirectMessageFailed<T>(IFailedDirectedTextMessage, T)

Declaration

```
public void OnDirectMessageFailed<T>(IFailedDirectedTextMessage failedMessage, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IFailedDirectedTextMessage	failedMessage	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnDirectMessageRecieved(IDirectedTextMessage)

Declaration

```
public void OnDirectMessageRecieved(IDirectedTextMessage message)
```

Parameters

TYPE	NAME	DESCRIPTION
IDirectedTextMessage	message	

OnDirectMessageRecieved<T>(IDirectedTextMessage, T)

Declaration

```
public void OnDirectMessageRecieved<T>(IDirectedTextMessage message, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IDirectedTextMessage	message	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnDirectMessageSent()

Declaration

```
public void OnDirectMessageSent()
```

OnDirectMessageSent<T>(T)

Declaration

```
public void OnDirectMessageSent<T>(T value)
```

Parameters

TYPE	NAME	DESCRIPTION
T	value	

Type Parameters

NAME	DESCRIPTION
T	

NAME	DESCRIPTION

OnEventMessageRecieved(IChannelTextMessage)

Declaration

```
public void OnEventMessageRecieved(IChannelTextMessage channelTextMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelTextMessage	channelTextMessage	

OnEventMessageRecieved<T>(IChannelTextMessage, T)

Declaration

```
public void OnEventMessageRecieved<T>(IChannelTextMessage channelTextMessage, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelTextMessage	channelTextMessage	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnLocalUserMuted()

Declaration

```
public void OnLocalUserMuted()
```

OnLocalUserMuted<T>(T)

Declaration

```
public void OnLocalUserMuted<T>(T value)
```

Parameters

TYPE	NAME	DESCRIPTION
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnLocalUserUnmuted()

Declaration

```
public void OnLocalUserUnmuted()
```

OnLocalUserUnmuted<T>(T)

Declaration

```
public void OnLocalUserUnmuted<T>(T value)
```

Parameters

TYPE	NAME	DESCRIPTION
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnLoggedIn(ILoginSession)

Declaration

```
public void OnLoggedIn(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

OnLoggedIn<T>(ILoginSession, T)

Declaration

```
public void OnLoggedIn<T>(ILoginSession loginSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnLoggedOut(ILoginSession)

Declaration

```
public void OnLoggedOut(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

OnLoggedOut<T>(ILoginSession, T)

Declaration

```
public void OnLoggedOut<T>(ILoginSession loginSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnLoggingIn(ILoginSession)

Declaration

```
public void OnLoggingIn(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

OnLoggingIn<T>(ILoginSession, T)

Declaration

```
public void OnLoggingIn<T>(ILoginSession loginSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnLoggingOut(ILoginSession)

Declaration

```
public void OnLoggingOut(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

OnLoggingOut<T>(ILoginSession, T)

Declaration

```
public void OnLoggingOut<T>(ILoginSession loginSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnLoginAdded(AccountId)

Declaration

```
public void OnLoginAdded(AccountId accountId)
```

Parameters

TYPE	NAME	DESCRIPTION
AccountId	accountId	

OnLoginRemoved(AccountId)

Declaration

```
public void OnLoginRemoved(AccountId accountId)
```

Parameters

TYPE	NAME	DESCRIPTION
AccountId	accountId	

OnLoginUpdated(ILoginSession)

Declaration

```
public void OnLoginUpdated(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

OnTextChannelConnected(IChannelSession)

Declaration

```
public void OnTextChannelConnected(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnTextChannelConnected<T>(IChannelSession, T)

Declaration

```
public void OnTextChannelConnected<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnTextChannelConnecting(IChannelSession)

Declaration

```
public void OnTextChannelConnecting(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnTextChannelConnecting<T>(IChannelSession, T)

Declaration

```
public void OnTextChannelConnecting<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

TYPE	NAME	DESCRIPTION
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnTextChannelDisconnected(IChannelSession)

Declaration

```
public void OnTextChannelDisconnected(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnTextChannelDisconnected<T>(IChannelSession, T)

Declaration

```
public void OnTextChannelDisconnected<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnTextChannelDisconnecting(IChannelSession)

Declaration

```
public void OnTextChannelDisconnecting(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

OnTextChannelDisconnecting<T>(IChannelSession, T)

Declaration

```
public void OnTextChannelDisconnecting<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Type Parameters

NAME	DESCRIPTION
T	

OnTTSMessageAdded(ITTSMessageQueueEventArgs)

Declaration

```
public void OnTTSMessageAdded(ITTSMessageQueueEventArgs ttsArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
ITTSMessageQueueEventArgs	ttsArgs	

OnTTSMessageRemoved(ITTSMessageQueueEventArgs)

Declaration

```
public void OnTTSMessageRemoved(ITTSMessageQueueEventArgs ttsArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
ITTSMessageQueueEventArgs	ttsArgs	

OnTTSMessageUpdated(ITTSMessageQueueEventArgs)

Declaration

```
public void OnTTSMessageUpdated(ITTSMessageQueueEventArgs ttsArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
ITTSMessageQueueEventArgs	ttsArgs	

OnUserCrossMuted(AccountId)

Declaration

```
public void OnUserCrossMuted(AccountId accountId)
```

Parameters

TYPE	NAME	DESCRIPTION
AccountId	accountId	

OnUserCrossUnmuted(AccountId)

Declaration

```
public void OnUserCrossUnmuted(AccountId accountId)
```

Parameters

TYPE	NAME	DESCRIPTION
AccountId	accountId	

OnUserJoinedChannel(IParticipant)

Declaration

```
public void OnUserJoinedChannel(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

OnUserLeftChannel(IParticipant)

Declaration

```
public void OnUserLeftChannel(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

OnUserMuted(IParticipant)

Declaration

```
public void OnUserMuted(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

OnUserNotSpeaking(IParticipant)

Declaration

```
public void OnUserNotSpeaking(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

OnUserSpeaking(IParticipant)

Declaration

```
public void OnUserSpeaking(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

OnUserUnmuted(IParticipant)

Declaration

```
public void OnUserUnmuted(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

OnUserValuesUpdated(IParticipant)

Declaration

```
public void OnUserValuesUpdated(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

Events

AudioChannelConnected

Declaration

```
public event Action<IChannelSession> AudioChannelConnected
```

Event Type

TYPE	DESCRIPTION
Action<IChannelSession>	

AudioChannelConnecting

Declaration

```
public event Action<IChannelSession> AudioChannelConnecting
```

Event Type

TYPE	DESCRIPTION
Action<IChannelSession>	

AudioChannelDisconnected

Declaration

```
public event Action<IChannelSession> AudioChannelDisconnected
```

Event Type

TYPE	DESCRIPTION
Action<IChannelSession>	

AudioChannelDisconnecting

Declaration

```
public event Action<IChannelSession> AudioChannelDisconnecting
```

Event Type

TYPE	DESCRIPTION
Action<IChannelSession>	

AudioInputDeviceAdded

Declaration

```
public event Action<IAudioDevice> AudioInputDeviceAdded
```

Event Type

TYPE	DESCRIPTION
Action<IAudioDevice>	

AudioInputDeviceRemoved

Declaration

```
public event Action<IAudioDevice> AudioInputDeviceRemoved
```

Event Type

TYPE	DESCRIPTION
Action<IAudioDevice>	

AudioInputDeviceUpdated

Declaration

```
public event Action<IAudioDevice> AudioInputDeviceUpdated
```

Event Type

TYPE	DESCRIPTION
Action<IAudioDevice>	

AudioOutputDeviceAdded

Declaration

```
public event Action<IAudioDevice> AudioOutputDeviceAdded
```

Event Type

TYPE	DESCRIPTION
Action<IAudioDevice>	

AudioOutputDeviceRemoved

Declaration

```
public event Action<IAudioDevice> AudioOutputDeviceRemoved
```

Event Type

TYPE	DESCRIPTION
Action<IAudioDevice>	

AudioOutputDeviceUpdated

Declaration

```
public event Action<IAudioDevice> AudioOutputDeviceUpdated
```

Event Type

TYPE	DESCRIPTION
Action<IAudioDevice>	

ChannelConnected

Declaration

```
public event Action<IChannelSession> ChannelConnected
```

Event Type

TYPE	DESCRIPTION
Action<IChannelSession>	

ChannelConnecting

Declaration

```
public event Action<IChannelSession> ChannelConnecting
```

Event Type

TYPE	DESCRIPTION
Action<IChannelSession>	

ChannelDisconnected

Declaration

```
public event Action<IChannelSession> ChannelDisconnected
```

Event Type

TYPE	DESCRIPTION
Action<IChannelSession>	

ChannelDisconnecting

Declaration

```
public event Action<IChannelSession> ChannelDisconnecting
```

Event Type

TYPE	DESCRIPTION
Action<IChannelSession>	

ChannelMessageRecieved

Declaration

```
public event Action<IChannelTextMessage> ChannelMessageRecieved
```

Event Type

TYPE	DESCRIPTION
Action<IChannelTextMessage>	

ChannelMesssageSent

Declaration

```
public event Action ChannelMesssageSent
```

Event Type

TYPE	DESCRIPTION
Action	

DirectMessageFailed

Declaration

```
public event Action<IFailedDirectedTextMessage> DirectMessageFailed
```

Event Type

TYPE	DESCRIPTION
Action<IFailedDirectedTextMessage>	

DirectMessageRecieved

Declaration

```
public event Action<IDirectedTextMessage> DirectMessageRecieved
```

Event Type

TYPE	DESCRIPTION
Action<IDirectedTextMessage>	

DirectMesssageSent

Declaration

```
public event Action DirectMesssageSent
```

Event Type

TYPE	DESCRIPTION
Action	

EventMessageRecieved

Declaration

```
public event Action<IChannelTextMessage> EventMessageRecieved
```

Event Type

TYPE	DESCRIPTION
Action<IChannelTextMessage>	

LocalUserMuted

Declaration

```
public event Action LocalUserMuted
```

Event Type

TYPE	DESCRIPTION
Action	

LocalUserUnmuted

Declaration

```
public event Action LocalUserUnmuted
```

Event Type

TYPE	DESCRIPTION
Action	

LoggedIn

Declaration

```
public event Action<ILoginSession> LoggedIn
```

Event Type

TYPE	DESCRIPTION
Action<ILoginSession>	

LoggedOut

Declaration

```
public event Action<ILoginSession> LoggedOut
```

Event Type

TYPE	DESCRIPTION
Action<ILoginSession>	

LoggingIn

Declaration

```
public event Action<ILoginSession> LoggingIn
```

Event Type

TYPE	DESCRIPTION
Action<ILoginSession>	

LoggingOut

Declaration

```
public event Action<ILoginSession> LoggingOut
```

Event Type

TYPE	DESCRIPTION
Action<ILoginSession>	

LoginAdded

Declaration

```
public event Action<AccountId> LoginAdded
```

Event Type

TYPE	DESCRIPTION
Action<AccountId>	

LoginRemoved

Declaration

```
public event Action<AccountId> LoginRemoved
```

Event Type

TYPE	DESCRIPTION
Action<AccountId>	

LoginUpdated

Declaration

```
public event Action<ILoginSession> LoginUpdated
```

Event Type

TYPE	DESCRIPTION
Action<ILoginSession>	

TextChannelConnected

Declaration

```
public event Action<IChannelSession> TextChannelConnected
```

Event Type

TYPE	DESCRIPTION
Action<IChannelSession>	

TextChannelConnecting

Declaration

```
public event Action<IChannelSession> TextChannelConnecting
```

Event Type

TYPE	DESCRIPTION
Action<IChannelSession>	

TextChannelDisconnected

Declaration

```
public event Action<IChannelSession> TextChannelDisconnected
```

Event Type

TYPE	DESCRIPTION
Action<IChannelSession>	

TextChannelDisconnecting

Declaration

```
public event Action<IChannelSession> TextChannelDisconnecting
```

Event Type

TYPE	DESCRIPTION
Action<IChannelSession>	

TTSMessageAdded

Declaration

```
public event Action<ITTSMessageQueueEventArgs> TTSMessageAdded
```

Event Type

TYPE	DESCRIPTION
Action<ITTSMessageQueueEventArgs>	

TTSMessageRemoved

Declaration

```
public event Action<ITTSMessageQueueEventArgs> TTSMessageRemoved
```

Event Type

TYPE	DESCRIPTION
Action<ITTSMessageQueueEventArgs>	

TTSMessageUpdated

Declaration

```
public event Action<ITTSMessageQueueEventArgs> TTSMessageUpdated
```

Event Type

TYPE	DESCRIPTION
Action<ITTSMessageQueueEventArgs>	

UserCrossMuted

Declaration

```
public event Action<AccountId> UserCrossMuted
```

Event Type

TYPE	DESCRIPTION
Action<AccountId>	

UserCrossUnmuted

Declaration

```
public event Action<AccountId> UserCrossUnmuted
```

Event Type

TYPE	DESCRIPTION
Action<AccountId>	

UserJoinedChannel

Declaration

```
public event Action<IParticipant> UserJoinedChannel
```

Event Type

TYPE	DESCRIPTION
Action<IParticipant>	

UserLeftChannel

Declaration

```
public event Action<IParticipant> UserLeftChannel
```

Event Type

TYPE	DESCRIPTION
Action<IParticipant>	

UserMuted

Declaration

```
public event Action<IParticipant> UserMuted
```

Event Type

TYPE	DESCRIPTION
Action<IParticipant>	

UserNotSpeaking

Declaration

```
public event Action<IParticipant> UserNotSpeaking
```

Event Type

TYPE	DESCRIPTION
Action<IParticipant>	

UserSpeaking

Declaration

```
public event Action<IParticipant> UserSpeaking
```

Event Type

TYPE	DESCRIPTION
Action<IParticipant>	

UserUnmuted

Declaration

```
public event Action<IParticipant> UserUnmuted
```

Event Type

TYPE	DESCRIPTION
Action<IParticipant>	

UserValuesUpdated

Declaration

```
public event Action<IParticipant> UserValuesUpdated
```

Event Type

TYPE	DESCRIPTION
Action<IParticipant>	

Class HandleDynamicEvents

Inheritance

System.Object
HandleDynamicEvents

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public static class HandleDynamicEvents
```

Fields

InternalAssemblyNames

Declaration

```
public static readonly HashSet<string> InternalAssemblyNames
```

Field Value

TYPE	DESCRIPTION
HashSet<System.String>	

Methods

Declaration

```
public static Dictionary<Enum, List<MethodInfo>> Methods
```

Field Value

TYPE	DESCRIPTION
Dictionary<Enum, List<MethodInfo>>	

Methods

AddDynamicEvent(Enum, MethodInfo)

Declaration

```
public static void AddDynamicEvent(Enum value, MethodInfo methodInfo)
```

Parameters

TYPE	NAME	DESCRIPTION
Enum	value	

TYPE	NAME	DESCRIPTION
MethodInfo	methodInfo	

LogRegisteredEventsCount(Boolean)

Declaration

```
public static void LogRegisteredEventsCount(bool logAllDynamicMethods)
```

Parameters

TYPE	NAME	DESCRIPTION
System.Boolean	logAllDynamicMethods	

RegisterAudioChannelEvents(Type[], BindingFlags)

Declaration

```
public static void RegisterAudioChannelEvents(Type[] types, BindingFlags flags)
```

Parameters

TYPE	NAME	DESCRIPTION
Type[]	types	
BindingFlags	flags	

RegisterAudioDeviceEvents(Type[], BindingFlags)

Declaration

```
public static void RegisterAudioDeviceEvents(Type[] types, BindingFlags flags)
```

Parameters

TYPE	NAME	DESCRIPTION
Type[]	types	
BindingFlags	flags	

RegisterChannelEvents(Type[], BindingFlags)

Declaration

```
public static void RegisterChannelEvents(Type[] types, BindingFlags flags)
```

Parameters

TYPE	NAME	DESCRIPTION
Type[]	types	
BindingFlags	flags	

RegisterChannelMessageEvents(Type[], BindingFlags)

Declaration

```
public static void RegisterChannelMessageEvents(Type[] types, BindingFlags flags)
```

Parameters

TYPE	NAME	DESCRIPTION
Type[]	types	
BindingFlags	flags	

RegisterDirectMessageEvents(Type[], BindingFlags)

Declaration

```
public static void RegisterDirectMessageEvents(Type[] types, BindingFlags flags)
```

Parameters

TYPE	NAME	DESCRIPTION
Type[]	types	
BindingFlags	flags	

RegisterEvents(List<String>, Boolean, Boolean)

Declaration

```
public static Task RegisterEvents(List<string> onlySearchTheseAssemblies, bool logAssemblySearches = true,
bool logAllDynamicMethods = false)
```

Parameters

TYPE	NAME	DESCRIPTION
List<System.String>	onlySearchTheseAssemblies	
System.Boolean	logAssemblySearches	
System.Boolean	logAllDynamicMethods	

Returns

TYPE	DESCRIPTION
Task	

RegisterLoginEvents(Type[], BindingFlags)

Declaration

```
public static void RegisterLoginEvents(Type[] types, BindingFlags flags)
```

Parameters

TYPE	NAME	DESCRIPTION

TYPE	NAME	DESCRIPTION
Type[]	types	
BindingFlags	flags	

RegisterTextChannelEvents(Type[], BindingFlags)

Declaration

```
public static void RegisterTextChannelEvents(Type[] types, BindingFlags flags)
```

Parameters

TYPE	NAME	DESCRIPTION
Type[]	types	
BindingFlags	flags	

RegisterTextToSpeechEvents(Type[], BindingFlags)

Declaration

```
public static void RegisterTextToSpeechEvents(Type[] types, BindingFlags flags)
```

Parameters

TYPE	NAME	DESCRIPTION
Type[]	types	
BindingFlags	flags	

RegisterUserEvents(Type[], BindingFlags)

Declaration

```
public static void RegisterUserEvents(Type[] types, BindingFlags flags)
```

Parameters

TYPE	NAME	DESCRIPTION
Type[]	types	
BindingFlags	flags	

Class LoginEventAsyncAttribute

Place this on an async void or async Task method to subscribe to Vivox Login Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen LoginStatus event happens

```
[LoginEventAsync(LoginStatus.LoggingIn)]
private async void OnPlayerLoggingInAsync(ILoginSession loginSession)
{
    Debug.Log($"Logging In : {loginSession.LoginSessionId.DisplayName}");
    await GetJoinedLobbies();
}
```

Check out the Docs Dynamic Async Events - Login Events

Inheritance

System.Object
LoginEventAsyncAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class LoginEventAsyncAttribute : Attribute
```

Constructors

LoginEventAsyncAttribute(LoginStatus)

EasyCode uses Reflection to find methods that contain LoginEventAsyncAttribute and subscribes these methods to Login event. EasyCode then invokes these methods dynamically with the specified options.

Declaration

```
public LoginEventAsyncAttribute(LoginStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
LoginStatus	options	The Login event status you want to subscribe to.

Properties

Options

Gets or sets the status for the Login event.

Declaration

```
public LoginStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
LoginStatus	

Class LoginEventAttribute

Place this on a method to subscribe to Vivox Login Events

Method will be called when chosen LoginStatus event happens

```
[LoginEvent(LoginStatus.LoggedIn)]"
public void UserLoggedIn(ILoginSession loginSession)
{
    $"Logged In {loginSession.LoginSessionId.DisplayName}";
}
```

Check out the Docs Login Events

Inheritance

System.Object

LoginEventAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class LoginEventAttribute : Attribute
```

Constructors

LoginEventAttribute(LoginStatus)

Declaration

```
public LoginEventAttribute(LoginStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
LoginStatus	options	

Properties

Options

Declaration

```
public LoginStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
LoginStatus	

Enum LoginStatus

The type of **Vivox Login Event** you want this method to be subscribed to. Works for LoginEventAttribute and LoginEventAsyncAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public enum LoginStatus
```

Fields

NAME	DESCRIPTION
LoggedIn	Event is invoked/fired when player is successfully logged into Vivox Method must contain only **1** parameter of type VivoxUnity.ILoginSession
LoggedOut	Event is invoked/fired when player is successfully logged out of Vivox Method must contain only **1** parameter of type VivoxUnity.ILoginSession
LoggingIn	Event is invoked/fired when player begins logging into Vivox Method must contain only **1** parameter of type VivoxUnity.ILoginSession
LoggingOut	Event is invoked/fired when player begins logging out of Vivox Method must contain only **1** parameter of type VivoxUnity.ILoginSession
LoginAdded	Event is invoked/fired when player is successfully logged into Vivox A player can log in multiple times under different usernames. Each time a new LoginSession is created EasyCode will keep track of the newly added LoginSessions You can access all current LoginSessions with EasySession `EasySession.LoginSessions["userName"]` Method must contain only **1** parameter of type VivoxUnity.AccountId

NAME	DESCRIPTION
LoginRemoved	Event is invoked/fired when player is successfully logged out of Vivox EasyCode will keep track of LoginSessions and automatically remove the LoginSession of the logged out player You can attempt to access a current LoginSession with EasySession to see if it exists. If it does **loginSession** will not be null `EasySession.LoginSessions.TryGetValue("userName", out ILoginSession loginSession);` Method must contain only **1** parameter of type VivoxUnity.AccountId
LoginValuesUpdated	Event is invoked/fired when player LoginSession has changed such as player has changed their name EasyCode will keep track of LoginSessions automatically You can attempt to access a current LoginSession with EasySession to see if it exists. If it does **loginSession** will not be null `EasySession.LoginSessions.TryGetValue("userName", out ILoginSession loginSession);` Method must contain only **1** parameter of type VivoxUnity.ILoginSession

Class TextChannelEventAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox Text Channel Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen TextChannelStatus event happens

```
[TextChannelEventAsync(TextChannelStatus.TextChannelConnecting)]
private async void OnTextChannelConnectingAsync(IChannelSession channelSession)
{
Debug.Log($"{channelSession.Channel.Name} Is Connecting");
await LoadPlayerData();
}
```

Check out the Docs Dynamic Async Events - Text Channel Events

Inheritance

System.Object

TextChannelEventAsyncAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class TextChannelEventAsyncAttribute : Attribute
```

Constructors

TextChannelEventAsyncAttribute(TextChannelStatus)

EasyCode uses Reflection to find methods that contain TextChannelEventAsyncAttribute and invokes these methods dynamically with the specified options.

Declaration

```
public TextChannelEventAsyncAttribute(TextChannelStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
TextChannelStatus	options	The status for the Text Channel event.

Properties

Options

Gets or sets the status for the Text Channel event.

Declaration

```
public TextChannelStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
TextChannelStatus	

Class TextChannelEventAttribute

Place this on a method to subscribe to Vivox Text Channel Events

Method will be called when chosen TextChannelStatus event happens

Example Method

```
[TextChannelEvent(TextChannelStatus.TextChannelConnecting)]
private void OnTextChannelConnecting(IChannelSession channelSession)
{
    Debug.Log($"{channelSession.Channel.Name} Is Connecting");
}
```

Check out the Docs Text Channel Events

Inheritance

System.Object

TextChannelEventAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class TextChannelEventAttribute : Attribute
```

Constructors

TextChannelEventAttribute(TextChannelStatus)

Declaration

```
public TextChannelEventAttribute(TextChannelStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
TextChannelStatus	options	

Properties

Options

Declaration

```
public TextChannelStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
TextChannelStatus	

Enum TextChannelStatus

The type of **Vivox Text Channel Event** that you want this method to be subscribed to. Works for TextChannelEventAttribute and TextChannelEventAsyncAttribute.

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public enum TextChannelStatus
```

Fields

NAME	DESCRIPTION
TextChannelConnected	Event is invoked/fired when player has successfully joined a Vivox Text Channel Will be fired for Echo, Non-Positional, and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelSession
TextChannelConnecting	Event is invoked/fired when player begins joining a Vivox Text Channel Will be fired for Echo, Non-Positional, and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelSession
TextChannelDisconnected	Event is invoked/fired when player has successfully disconnected from a Vivox Text Channel Will be fired for Echo, Non-Positional, and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelSession
TextChannelDisconnecting	Event is invoked/fired when player begins disconnecting from a Vivox Text Channel Will be fired for Echo, Non-Positional, and 3D Positional channels Method must contain only **1** parameter of type VivoxUnity.IChannelSession

Class TextToSpeechEventAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox Text-To-Speech Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen TextToSpeechStatus event happens

```
[TextToSpeechEventAsync(TextToSpeechStatus.TTSMessageAdded)]
private async void  OnTTSMessageAddedAsync(ITTSMessageQueueEventArgs ttsArgs)
{
Debug.Log($"TTS Message Has Been Added : {ttsArgs.Message.Text}");
await SavePlayerData();
}
```

Check out the Docs Dynamic Async Events - Text To Speech Events

Inheritance

System.Object
TextToSpeechEventAsyncAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class TextToSpeechEventAsyncAttribute : Attribute
```

Constructors

TextToSpeechEventAsyncAttribute(TextToSpeechStatus)

EasyCode uses Reflection to find methods with that contain TextToSpeechEventAsyncAttribute and invokes these methods dynamically with the specified options.

Declaration

```
public TextToSpeechEventAsyncAttribute(TextToSpeechStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
TextToSpeechStatus	options	The status for the Text-to-Speech event.

Properties

Options

Gets or sets the status for the Text-to-Speech event.

Declaration

```
public TextToSpeechStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
TextToSpeechStatus	

Class TextToSpeechEventAttribute

Place this on a method to subscribe to Vivox Text-To-Speech Events

Method will be called when chosen TextToSpeechStatus event happens

Example Method

```
[TextToSpeechEvent(TextToSpeechStatus.TTSMessageAdded)]
private void OnTTSMessageAdded(ITTSMessageQueueEventArgs ttsArgs)
{
    Debug.Log($"TTS Message Has Been Added : {ttsArgs.Message.Text}");
}
```

Check out the Docs Text To Speech Events

Inheritance

System.Object

TextToSpeechEventAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class TextToSpeechEventAttribute : Attribute
```

Constructors

TextToSpeechEventAttribute(TextToSpeechStatus)

Declaration

```
public TextToSpeechEventAttribute(TextToSpeechStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
TextToSpeechStatus	options	

Properties

Options

Declaration

```
public TextToSpeechStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
TextToSpeechStatus	

Enum TextToSpeechStatus

The type of **Vivox Text-to-Speech Event** that you want this method to be subscribed to. Works for TextToSpeechEventAttribute and TextToSpeechEventAsyncAttribute.

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public enum TextToSpeechStatus
```

Fields

NAME	DESCRIPTION
TTSMessageAdded	Event is invoked/fired when a Text-To-Speech message is added to the queue and is spoken/played Method must contain only **1** parameter of type VivoxUnity.ITTSMessageQueueEventArgs
TTSMessageRemoved	Event is invoked/fired when a Text-To-Speech message is removed from the queue and is disposed of or canceled Method must contain only **1** parameter of type VivoxUnity.ITTSMessageQueueEventArgs
TTSMessageUpdated	Event is invoked/fired when a Text-To-Speech message is removed from queue and begins to play Method must contain only **1** parameter of type VivoxUnity.ITTSMessageQueueEventArgs

Class UserEventsAsyncAttribute

Place this on a async void or async Task method to subscribe to Vivox User Events asynchronously

Do not modify any GameObjects, UI, or anything that relies/runs on Unity's main thread

Method will be called when chosen UserStatus event happens

```
[UserEventsAsync(UserStatus.LocalUserMuted)]
private async void OnLocalUserMutedAsync()
{
Debug.Log("Local User is Muted");
await SavePlayerData();
}
```

Check out the Docs Dynamic Async Events - User Events

Inheritance

System.Object
UserEventsAsyncAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class UserEventsAsyncAttribute : Attribute
```

Constructors

UserEventsAsyncAttribute(UserStatus)

EasyCode uses Reflection to find methods that contain UserEventsAsyncAttribute and invokes these methods dynamically with the specified options.

Declaration

```
public UserEventsAsyncAttribute(UserStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
UserStatus	options	The status for the User event.

Properties

Options

Gets or sets the status for the User event.

Declaration

```
public UserStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
UserStatus	

Class UserEventsAttribute

Place this on a method to subscribe to Vivox User Events

Online Docs (Dynamic Events - User Participant Events)

Method will be called when chosen UserStatus event happens

Example Method

```
[UserEvents(UserStatus.UserMuted)]
private void OnUserMuted(IParticipant participant)
{
    Debug.Log($"{participant.Account.DisplayName} Is Muted : (Muted For All : {participant.IsMutedForAll})");
}
```

Check out the Docs User Events

Inheritance

System.Object
UserEventsAttribute

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public class UserEventsAttribute : Attribute
```

Constructors

UserEventsAttribute(UserStatus)

Declaration

```
public UserEventsAttribute(UserStatus options)
```

Parameters

TYPE	NAME	DESCRIPTION
UserStatus	options	

Properties

Options

Declaration

```
public UserStatus Options { get; set; }
```

Property Value

TYPE	DESCRIPTION
UserStatus	

Enum UserStatus

The type of **Vivox User Event** that you want this method to be subscribed to. Works for UserEventsAttribute and UserEventsAsyncAttribute.

Namespace: EasyCodeForVivox.Events

Assembly: cs.temp.dll.dll

Syntax

```
public enum UserStatus
```

Fields

NAME	DESCRIPTION
LocalUserMuted	Event is invoked/fired when the local player mutes themselves in a Vivox Channel Method must have **0** parameters
LocalUserUnmuted	Event is invoked/fired when the local player unmutes themselves in a Vivox Channel Method must have **0** parameters
UserCrossMuted	Event is invoked/fired when a player gets cross muted in a Vivox Channel Method must contain only **1** parameter of type VivoxUnity.AccountId
UserCrossUnmuted	Event is invoked/fired when a player gets cross unmuted in a Vivox Channel Method must contain only **1** parameter of type VivoxUnity.AccountId
UserJoinedChannel	Event is invoked/fired when a player joins a Vivox Channel Method must contain only **1** parameter of type VivoxUnity.IParticipant
UserLeftChannel	Event is invoked/fired when a player leaves a Vivox Channel Method must contain only **1** parameter of type VivoxUnity.IParticipant
UserMuted	Event is invoked/fired when a player gets muted in a Vivox Channel Method must contain only **1** parameter of type VivoxUnity.IParticipant
UserNotSpeaking	Event is invoked/fired when a player stops speaking in a Vivox Channel Method must contain only **1** parameter of type VivoxUnity.IParticipant

NAME	DESCRIPTION
UserSpeaking	Event is invoked/fired when a player is speaking in a Vivox Channel Method must contain only **1** parameter of type VivoxUnity.IParticipant
UserUnmuted	Event is invoked/fired when a player gets unmuted in a Vivox Channel Method must contain only **1** parameter of type VivoxUnity.IParticipant
UserValuesUpdated	Event is invoked/fired when a players values get updated in a Vivox Channel such as being muted/unmuted Method must contain only **1** parameter of type VivoxUnity.IParticipant

Namespace EasyCodeForVivox.Events.Internal

Classes

EasyEventsAsync

Class EasyEventsAsync

Inheritance

System.Object
EasyEventsAsync

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox.Events.Internal

Assembly: cs.temp.dll.dll

Syntax

```
public class EasyEventsAsync
```

Constructors

EasyEventsAsync(EasySettingsSO)

Declaration

```
public EasyEventsAsync(EasySettingsSO settings)
```

Parameters

TYPE	NAME	DESCRIPTION
EasySettingsSO	settings	

Methods

OnAudioChannelConnectedAsync(IChannelSession)

Declaration

```
public Task OnAudioChannelConnectedAsync(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
Task	

OnAudioChannelConnectedAsync<T>(IChannelSession, T)

Declaration

```
public Task OnAudioChannelConnectedAsync<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnAudioChannelConnectingAsync(IChannelSession)

Declaration

```
public Task OnAudioChannelConnectingAsync(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
Task	

OnAudioChannelConnectingAsync<T>(IChannelSession, T)

Declaration

```
public Task OnAudioChannelConnectingAsync<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnAudioChannelDisconnectedAsync(IChannelSession)

Declaration

```
public Task OnAudioChannelDisconnectedAsync(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
Task	

OnAudioChannelDisconnectedAsync<T>(IChannelSession, T)

Declaration

```
public Task OnAudioChannelDisconnectedAsync<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnAudioChannelDisconnectingAsync(IChannelSession)

Declaration

```
public Task OnAudioChannelDisconnectingAsync(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
Task	

OnAudioChannelDisconnectingAsync<T>(IChannelSession, T)

Declaration

```
public Task OnAudioChannelDisconnectingAsync<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnAudioInputDeviceAddedAsync(IAudioDevice)

Declaration

```
public void OnAudioInputDeviceAddedAsync(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioInputDeviceRemovedAsync(IAudioDevice)

Declaration

```
public void OnAudioInputDeviceRemovedAsync(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioInputDeviceUpdatedAsync(IAudioDevice)

Declaration

```
public void OnAudioInputDeviceUpdatedAsync(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioOutputDeviceAdded(IAudioDevice)

Declaration

```
public void OnAudioOutputDeviceAdded(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioOutputDeviceRemoved(IAudioDevice)

Declaration

```
public void OnAudioOutputDeviceRemoved(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnAudioOutputDeviceUpdated(IAudioDevice)

Declaration

```
public void OnAudioOutputDeviceUpdated(IAudioDevice audioDevice)
```

Parameters

TYPE	NAME	DESCRIPTION
IAudioDevice	audioDevice	

OnChannelConnectedAsync(IChannelSession)

Declaration

```
public Task OnChannelConnectedAsync(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
Task	

OnChannelConnectedAsync<T>(IChannelSession, T)

Declaration

```
public Task OnChannelConnectedAsync<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnChannelConnectingAsync(IChannelSession)

Declaration

```
public Task OnChannelConnectingAsync(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
Task	

OnChannelConnectingAsync<T>(IChannelSession, T)

Declaration

```
public Task OnChannelConnectingAsync<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnChannelDisconnectedAsync(IChannelSession)

Declaration

```
public Task OnChannelDisconnectedAsync(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
Task	

OnChannelDisconnectedAsync<T>(IChannelSession, T)

Declaration

```
public Task OnChannelDisconnectedAsync<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnChannelDisconnectingAsync(IChannelSession)

Declaration

```
public Task OnChannelDisconnectingAsync(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
Task	

OnChannelDisconnectingAsync<T>(IChannelSession, T)

Declaration

```
public Task OnChannelDisconnectingAsync<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnChannelMessageRecievedAsync(IChannelTextMessage)

Declaration

```
public Task OnChannelMessageRecievedAsync(IChannelTextMessage channelTextMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelTextMessage	channelTextMessage	

Returns

TYPE	DESCRIPTION
Task	

OnChannelMessageRecievedAsync<T>(IChannelTextMessage, T)

Declaration

```
public Task OnChannelMessageRecievedAsync<T>(IChannelTextMessage channelTextMessage, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelTextMessage	channelTextMessage	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnChannelMessageSentAsync()

Declaration

```
public Task OnChannelMessageSentAsync()
```

Returns

TYPE	DESCRIPTION
Task	

OnChannelMessageSentAsync<T>(T)

Declaration

```
public Task OnChannelMessageSentAsync<T>(T value)
```

Parameters

TYPE	NAME	DESCRIPTION
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnDirectMessageFailedAsync(IFailedDirectedTextMessage)

Declaration

```
public Task OnDirectMessageFailedAsync(IFailedDirectedTextMessage failedMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
IFailedDirectedTextMessage	failedMessage	

Returns

TYPE	DESCRIPTION
Task	

OnDirectMessageFailedAsync<T>(IFailedDirectedTextMessage, T)

Declaration

```
public Task OnDirectMessageFailedAsync<T>(IFailedDirectedTextMessage failedMessage, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IFailedDirectedTextMessage	failedMessage	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnDirectMessageRecievedAsync(IDirectedTextMessage)

Declaration

```
public Task OnDirectMessageRecievedAsync(IDirectedTextMessage message)
```

Parameters

TYPE	NAME	DESCRIPTION
IDirectedTextMessage	message	

Returns

TYPE	DESCRIPTION
Task	

OnDirectMessageRecievedAsync<T>(IDirectedTextMessage, T)

Declaration

```
public Task OnDirectMessageRecievedAsync<T>(IDirectedTextMessage message, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IDirectedTextMessage	message	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnDirectMessageSentAsync()

Declaration

```
public Task OnDirectMessageSentAsync()
```

Returns

TYPE	DESCRIPTION
Task	

OnDirectMessageSentAsync<T>(T)

Declaration

```
public Task OnDirectMessageSentAsync<T>(T value)
```

Parameters

TYPE	NAME	DESCRIPTION
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnEventMessageRecievedAsync(IChannelTextMessage)

Declaration

```
public Task OnEventMessageRecievedAsync(IChannelTextMessage channelTextMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelTextMessage	channelTextMessage	

Returns

TYPE	DESCRIPTION
Task	

OnEventMessageRecievedAsync<T>(IChannelTextMessage, T)

Declaration

```
public Task OnEventMessageRecievedAsync<T>(IChannelTextMessage channelTextMessage, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelTextMessage	channelTextMessage	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnLocalUserMutedAsync()

Declaration

```
public Task OnLocalUserMutedAsync()
```

Returns

TYPE	DESCRIPTION
Task	

OnLocalUserMutedAsync<T>(T)

Declaration

```
public Task OnLocalUserMutedAsync<T>(T value)
```

Parameters

TYPE	NAME	DESCRIPTION
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnLocalUserUnmutedAsync()

Declaration

```
public Task OnLocalUserUnmutedAsync()
```

Returns

TYPE	DESCRIPTION
Task	

OnLocalUserUnmutedAsync<T>(T)

Declaration

```
public Task OnLocalUserUnmutedAsync<T>(T value)
```

Parameters

TYPE	NAME	DESCRIPTION
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnLoggedInAsync(ILoginSession)

Declaration

```
public Task OnLoggedInAsync(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

Returns

TYPE	DESCRIPTION
Task	

OnLoggedInAsync<T>(ILoginSession, T)

Declaration

```
public Task OnLoggedInAsync<T>(ILoginSession loginSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnLoggedOutAsync(ILoginSession)

Declaration

```
public Task OnLoggedOutAsync(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

Returns

TYPE	DESCRIPTION
Task	

OnLoggedOutAsync<T>(ILoginSession, T)

Declaration

```
public Task OnLoggedOutAsync<T>(ILoginSession loginSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnLoggingInAsync(ILoginSession)

Declaration

```
public Task OnLoggingInAsync(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

Returns

TYPE	DESCRIPTION
Task	

OnLoggingInAsync<T>(ILoginSession, T)

Declaration

```
public Task OnLoggingInAsync<T>(ILoginSession loginSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnLoggingOutAsync(ILoginSession)

Declaration

```
public Task OnLoggingOutAsync(ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

Returns

TYPE	DESCRIPTION
Task	

OnLoggingOutAsync<T>(ILoginSession, T)

Declaration

```
public Task OnLoggingOutAsync<T>(ILoginSession loginSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnTextChannelConnectedAsync(IChannelSession)

Declaration

```
public Task OnTextChannelConnectedAsync(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
Task	

OnTextChannelConnectedAsync<T>(IChannelSession, T)

Declaration

```
public Task OnTextChannelConnectedAsync<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnTextChannelConnectingAsync(IChannelSession)

Declaration

```
public Task OnTextChannelConnectingAsync(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
Task	

OnTextChannelConnectingAsync<T>(IChannelSession, T)

Declaration

```
public Task OnTextChannelConnectingAsync<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnTextChannelDisconnectedAsync(IChannelSession)

Declaration

```
public Task OnTextChannelDisconnectedAsync(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
Task	

OnTextChannelDisconnectedAsync<T>(IChannelSession, T)

Declaration

```
public Task OnTextChannelDisconnectedAsync<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnTextChannelDisconnectingAsync(IChannelSession)

Declaration

```
public Task OnTextChannelDisconnectingAsync(IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
Task	

OnTextChannelDisconnectingAsync<T>(IChannelSession, T)

Declaration

```
public Task OnTextChannelDisconnectingAsync<T>(IChannelSession channelSession, T value)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	
T	value	

Returns

TYPE	DESCRIPTION
Task	

Type Parameters

NAME	DESCRIPTION
T	

OnTTSMessageAddedAsync(ITTSMessageQueueEventArgs)

Declaration

```
public Task OnTTSMessageAddedAsync(ITTSMessageQueueEventArgs ttsArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
ITTSMessageQueueEventArgs	ttsArgs	

Returns

TYPE	DESCRIPTION
Task	

OnTTSMessageRemovedAsync(ITTSMessageQueueEventArgs)

Declaration

```
public Task OnTTSMessageRemovedAsync(ITTSMessageQueueEventArgs ttsArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
ITTSMessageQueueEventArgs	ttsArgs	

Returns

TYPE	DESCRIPTION
Task	

OnTTSMessageUpdatedAsync(ITTSMessageQueueEventArgs)

Declaration

```
public Task OnTTSMessageUpdatedAsync(ITTSMessageQueueEventArgs ttsArgs)
```

Parameters

TYPE	NAME	DESCRIPTION
ITTSMessageQueueEventArgs	ttsArgs	

Returns

TYPE	DESCRIPTION
Task	

OnUserCrossMutedAsync(AccountId)

Declaration

```
public Task OnUserCrossMutedAsync(AccountId accountId)
```

Parameters

TYPE	NAME	DESCRIPTION
AccountId	accountId	

Returns

TYPE	DESCRIPTION
Task	

OnUserCrossUnmutedAsync(AccountId)

Declaration

```
public Task OnUserCrossUnmutedAsync(AccountId accountId)
```

Parameters

TYPE	NAME	DESCRIPTION
AccountId	accountId	

Returns

TYPE	DESCRIPTION
Task	

OnUserJoinedChannelAsync(IParticipant)

Declaration

```
public Task OnUserJoinedChannelAsync(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

Returns

TYPE	DESCRIPTION
Task	

OnUserLeftChannelAsync(IParticipant)

Declaration

```
public Task OnUserLeftChannelAsync(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

Returns

TYPE	DESCRIPTION
Task	

OnUserMutedAsync(IParticipant)

Declaration

```
public Task OnUserMutedAsync(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

Returns

TYPE	DESCRIPTION
Task	

OnUserNotSpeakingAsync(IParticipant)

Declaration

```
public Task OnUserNotSpeakingAsync(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

Returns

TYPE	DESCRIPTION
Task	

OnUserSpeakingAsync(IParticipant)

Declaration

```
public Task OnUserSpeakingAsync(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

Returns

TYPE	DESCRIPTION
Task	

OnUserUnmutedAsync(IParticipant)

Declaration

```
public Task OnUserUnmutedAsync(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

Returns

TYPE	DESCRIPTION
Task	

OnUserValuesUpdatedAsync(IParticipant)

Declaration

```
public Task OnUserValuesUpdatedAsync(IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

Returns

TYPE	DESCRIPTION
Task	

Namespace EasyCodeForVivox.Extensions

Classes

EasyDebug

EasySIPExtensions

GameObjectExtensions

TTSMessageExtensions

UIExtensions

VivoxExtensions

Class EasyDebug

Inheritance

System.Object
EasyDebug

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox.Extensions

Assembly: cs.temp.dll.dll

Syntax

```
public static class EasyDebug
```

Fields

Aqua

Declaration

```
public const string Aqua = "#00ffffff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Blue

Declaration

```
public const string Blue = "#0000ffff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Brown

Declaration

```
public const string Brown = "#a52a2aff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Cyan

Declaration

```
public const string Cyan = "#00ffffff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Darkblue

Declaration

```
public const string Darkblue = "#0000a0ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Fuchsia

Declaration

```
public const string Fuchsia = "#ff00ffff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Green

Declaration

```
public const string Green = "#008000ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Grey

Declaration

```
public const string Grey = "#808080ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Lightblue

Declaration

```
public const string Lightblue = "#add8e6ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Lime

Declaration

```
public const string Lime = "#00ff00ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Magenta

Declaration

```
public const string Magenta = "#ff00ffff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Maroon

Declaration

```
public const string Maroon = "#800000ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Navy

Declaration

```
public const string Navy = "#000080ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Olive

Declaration

```
public const string Olive = "#808000ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Orange

Declaration

```
public const string Orange = "#ffa500ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Purple

Declaration

```
public const string Purple = "#800080ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Red

Declaration

```
public const string Red = "#ff0000ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Silver

Declaration

```
public const string Silver = "#c0c0c0ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Teal

Declaration

```
public const string Teal = "#008080ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

White

Declaration

```
public const string White = "#ffffffff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Yellow

Declaration

```
public const string Yellow = "#ffff00ff"
```

Field Value

TYPE	DESCRIPTION
System.String	

Methods

Bold(String)

Declaration

```
public static string Bold(this string msg)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	msg	

Returns

TYPE	DESCRIPTION
System.String	

Color(String, String)

Declaration

```
public static string Color(this string msg, string color)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	msg	

TYPE	NAME	DESCRIPTION
System.String	color	

Returns

TYPE	DESCRIPTION
System.String	

Italic(String)

Declaration

```
public static string Italic(this string msg)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	msg	

Returns

TYPE	DESCRIPTION
System.String	

Class EasySIPExtensions

Inheritance

System.Object
EasySIPExtensions

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox.Extensions

Assembly: cs.temp.dll.dll

Syntax

```
public static class EasySIPExtensions
```

Methods

GetSIP(IChannelSession)

Gets the valid Vivox SIP address from this IChannelSession

Declaration

```
public static string GetSIP(this IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
System.String	

GetSIP(ILoginSession)

Gets the valid Vivox SIP address from this ILoginSession

Declaration

```
public static string GetSIP(this ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	

Returns

TYPE	DESCRIPTION
System.String	

GetSIP(IParticipant)

Gets the valid Vivox SIP address from this IParticipant

Declaration

```
public static string GetSIP(this IParticipant participant)
```

Parameters

TYPE	NAME	DESCRIPTION
IParticipant	participant	

Returns

TYPE	DESCRIPTION
System.String	

IsSelf(IChannelSession)

Checks if this IchannelSession is the current logged in user

Declaration

```
public static bool IsSelf(this IChannelSession channelSession)
```

Parameters

TYPE	NAME	DESCRIPTION
IChannelSession	channelSession	

Returns

TYPE	DESCRIPTION
System.Boolean	

Class GameObjectExtensions

Inheritance

System.Object
GameObjectExtensions

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox.Extensions

Assembly: cs.temp.dll.dll

Syntax

```
public static class GameObjectExtensions
```

Methods

SwitchTo(GameObject, GameObject)

Deactivates this Gameobject and activates another Gameobject

Declaration

```
public static void SwitchTo(this GameObject toDeactivate, GameObject toActivate)
```

Parameters

TYPE	NAME	DESCRIPTION
GameObject	toDeactivate	Gameobject to Deactivate
GameObject	toActivate	Gameobject to Activate

Class TTSMessageExtensions

Inheritance

System.Object
TTSMessageExtensions

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox.Extensions

Assembly: cs.temp.dll.dll

Syntax

```
public static class TTSMessageExtensions
```

Methods

TTSMsgLocalPlayOverCurrent(String, ILoginSession)

Play this message locally and override current playing TTS message

Declaration

```
public static void TTSMsgLocalPlayOverCurrent(this string message, ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	message	
ILoginSession	loginSession	

TTSMsgLocalRemotePlayOverCurrent(String, ILoginSession)

Play this message locally and remotely and ovverride current playing TTS message

Declaration

```
public static void TTSMsgLocalRemotePlayOverCurrent(this string message, ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	message	
ILoginSession	loginSession	

TTSMsgLocalReplaceCurrentMessagePlaying(String, ILoginSession)

Replace current playing TTS message with this message locally

Declaration

```
public static void TTSMsgLocalReplaceCurrentMessagePlaying(this string message, ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	message	
ILoginSession	loginSession	

TTSMsgQueueLocal(String, ILoginSession)

Play TTS message locally, adds to current queue if a message is already playing

Declaration

```
public static void TTSMsgQueueLocal(this string message, ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	message	
ILoginSession	loginSession	

TTSMsgQueueRemote(String, ILoginSession)

Play TTS message remotely, adds to current queue if a message is already playing

Declaration

```
public static void TTSMsgQueueRemote(this string message, ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	message	
ILoginSession	loginSession	

TTSMsgQueueRemoteLocal(String, ILoginSession)

Play TTS message remotely and locally, adds to current queue if a message is already playing

Declaration

```
public static void TTSMsgQueueRemoteLocal(this string message, ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	message	
ILoginSession	loginSession	

TTSMsgRemotePlayOverCurrent(String, ILoginSession)

Play this message remotely and override current playing TTS message

Declaration

```
public static void TTSMsgRemotePlayOverCurrent(this string message, ILoginSession loginSession)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	message	
ILoginSession	loginSession	

WaitForMessage(ILoginSession, TTSMessage)

Declaration

```
public static IEnumerator WaitForMessage(ILoginSession loginSession, TTSMessage ttsMessage)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
TTSMessage	ttsMessage	

Returns

TYPE	DESCRIPTION
IEnumerator	

Class UIExtensions

Inheritance

System.Object
UIExtensions

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox.Extensions

Assembly: cs.temp.dll.dll

Syntax

```
public static class UIExtensions
```

Methods

AddValue(TMP_Dropdown, String)

Declaration

```
public static void AddValue(this TMP_Dropdown dropdown, string valueToAdd)
```

Parameters

TYPE	NAME	DESCRIPTION
TMP_Dropdown	dropdown	
System.String	valueToAdd	

GetSelected(TMP_Dropdown)

Declaration

```
public static string GetSelected(this TMP_Dropdown dropdown)
```

Parameters

TYPE	NAME	DESCRIPTION
TMP_Dropdown	dropdown	

Returns

TYPE	DESCRIPTION
System.String	

RemoveValue(TMP_Dropdown, String)

Declaration

```
public static void RemoveValue(this TMP_Dropdown dropdown, string valueToRemove)
```

Parameters

TYPE	NAME	DESCRIPTION
TMP_Dropdown	dropdown	
System.String	valueToRemove	

TurnOff(Toggle)

Declaration

```
public static void TurnOff(this Toggle toggle)
```

Parameters

TYPE	NAME	DESCRIPTION
Toggle	toggle	

TurnOn(Toggle)

Declaration

```
public static void TurnOn(this Toggle toggle)
```

Parameters

TYPE	NAME	DESCRIPTION
Toggle	toggle	

Class VivoxExtensions

Inheritance

System.Object
VivoxExtensions

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox.Extensions

Assembly: cs.temp.dll.dll

Syntax

```
public static class VivoxExtensions
```

Methods

GetChannelId(ILoginSession, String)

Declaration

```
public static ChannelId GetChannelId(this ILoginSession loginSession, string channelName)
```

Parameters

TYPE	NAME	DESCRIPTION
ILoginSession	loginSession	
System.String	channelName	

Returns

TYPE	DESCRIPTION
ChannelId	

GetMD5Hash(String)

Declaration

```
public static string GetMD5Hash(this string valueToHash)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	valueToHash	

Returns

TYPE	DESCRIPTION
System.String	

Namespace EasyCodeForVivox.Utilities

Classes

EasyVivoxUtilities

Class EasyVivoxUtilities

Inheritance

System.Object
EasyVivoxUtilities

Inherited Members

System.Object.ToString()
System.Object.Equals(System.Object)
System.Object.Equals(System.Object, System.Object)
System.Object.ReferenceEquals(System.Object, System.Object)
System.Object.GetHashCode()
System.Object.GetType()
System.Object.MemberwiseClone()

Namespace: EasyCodeForVivox.Utilities

Assembly: cs.temp.dll.dll

Syntax

```
public static class EasyVivoxUtilities
```

Methods

FilterChannelAndUserName(String)

Declaration

```
public static bool FilterChannelAndUserName(string nameToFilter)
```

Parameters

TYPE	NAME	DESCRIPTION
System.String	nameToFilter	

Returns

TYPE	DESCRIPTION
System.Boolean	

RequestAndroidMicPermission()

Declaration

```
public static void RequestAndroidMicPermission()
```

RequestIOSMicrophoneAccess()

Declaration

```
public static void RequestIOSMicrophoneAccess()
```