Photon Voice

v2.9

Generated by Doxygen 1.8.10

Fri Aug 16 2019 15:45:51

Contents

Chapter 1

Photon Voice Doxygen Readme

Offline Docs

Manual Generation

To manually generate doxygen offline files:

"doxygen .\Docs\doxygen\voice-doxygen-offline.config"

Automatic Generation

Simply run "Docs\generate_offline.bat". Open the file and edit DOXYGEN_PATH accordingly. Also you need a LaTeX distrubution installed and some packages/dependencies.

This script will also copy the offline files to their respective locations and then clean up.

Files

HTML

It is not possible to disable HTML files generation. So those are just ignored or cleaned up after generation.

CHM

"PhotonVoice-Documentation.chm" should be copied

to "Assets\Photon\PhotonVoice-Documentation.chm"

from "Docs\TempOutputDocs\VOICE_OFFLINE_HTML\PhotonVoice-Documentation.chm".

PDF

"PhotonVoice-Documentation.pdf" should be copied

to "Assets\Photon\PhotonVoice-Documentation.pdf"

from "Docs\TempOutputDocs\latex\refman.pdf".

Online Docs

To manually generate doxygen online files:

"doxygen .\Docs\doxygen\voice-doxygen-online.config"

Chapter 2

Namespace Documentation

2.1 Photon Namespace Reference

Namespaces

- namespace Voice

2.2 Photon.Voice Namespace Reference

Namespaces

- namespace IOS
- namespace PUN
- namespace Unity

Classes

- class AudioDesc
- class AudioInEnumerator
- class AudioStreamPlayer
- class AudioUtil

  Collection of Audio Utility functions and classes.
- class BufferReaderPushAdapter

  Simple BufferReaderPushAdapterBase implementation using a single buffer, using synchronous LocalVoice.PushData
- class BufferReaderPushAdapterAsyncPool

  BufferReaderPushAdapter implementation using asynchronous LocalVoice.PushDataAsync.
- class BufferReaderPushAdapterAsyncPoolCopy

  BufferReaderPushAdapter implementation using asynchronous LocalVoice.PushDataAsync and data copy.
- class BufferReaderPushAdapterAsyncPoolFloatToShort

  BufferReaderPushAdapter implementation using asynchronous LocalVoice.PushDataAsync, converting float samples to short.
- class BufferReaderPushAdapterBase

  Adapter base class to move data by reading from IDataReader.Read and pushing to LocalVoice.
- class FactoryPrimitiveArrayPool

  PrimitiveArrayPool<T> as wrapped in object factory interface.
- class FactoryReusableArray

Array factory returnig the same array instance as long as it requested with the same array length. If length changes, new array instance created.

- class Framer

  Utility class to re-frame audio packets.
- interface IAudioDesc

  Audio Source interface.
- interface IAudioOut
- interface IAudioPusher

  Audio Pusher interface.
- interface IAudioReader

  Audio Reader interface.
- interface IDataReader

  Interface for pulling data, in case this is more appropriate than pushing it.
- interface IDecoder

  Generic decoder interface.
- interface IDecoderQueuedOutputImageNative
- interface IEncoder

  Generic encoder interface.
- interface IEncoderDirect

  Interface for an encoder which consumes input data via explicit call.
- interface ILocalVoiceAudio

  Interface for an outgoing audio stream.
- interface ILogger
- class ImageBufferInfo
- class ImageBufferNative
- class ImageBufferNativeAlloc
- class ImageBufferNativeGCHandleSinglePlane
- class ImageBufferNativePool
- struct ImageInputBuf
- struct ImageOutputBuf
- interface IProcessor

  Audio Processor interface.
- interface IServiceable

  Interface for classes that want their Service() function to be called regularly in the context of a LocalVoice.
- interface ISyncAudioOut
- interface IVoiceTransport
- class LoadBalancingFrontend
- class LoadBalancingTransport

  Extends LoadBalancingClient with audio streaming functionality.
- class LocalVoice

  Represents outgoing data stream.
- class LocalVoiceAudio

  Outgoing audio stream.
- class LocalVoiceAudioDummy

  Dummy LocalVoiceAudio
- class LocalVoiceAudioFloat

  Specialization of LocalVoiceAudio for float audio
- class LocalVoiceAudioShort

  Specialization of LocalVoiceAudio for short audio
- class LocalVoiceFramed

  Typed re-framing LocalVoice
- class LocalVoiceFramedBase

Typed re-framing LocalVoice

- interface ObjectFactory

  Uniform interface to ObjectPool<TType, TInfo> and single reusable object.
- class ObjectPool

  Generic Pool to re-use objects of a certain type (TType) that optionally match a certain property or set of properties (TInfo).
- class OpusCodec
- class **PhotonTransportProtocol**
- class PrimitiveArrayPool

  Pool of Arrays with components of type T, with ObjectPool info being the array's size.
- class **RemoteVoice**
- class RemoteVoiceInfo

  Information about a remote voice (incoming stream).
- struct RemoteVoiceOptions

  Event Actions and other options for a remote voice (incoming stream).
- class UnsupportedCodecException

  Exception thrown if an unsupported codec is encountered.
- class UnsupportedSampleTypeException

  Exception thrown if an unsupported audio sample type is encountered.
- class VoiceClient

  Voice client interact with other clients on network via IVoiceTransport.
- class VoiceEvent
- struct VoiceInfo

  Describes stream properties.
- class WebRTCAudioLib
- class WebRTCAudioProcessor

Enumerations

- enum Codec

  Enum for Media Codecs supported by PhotonVoice.
- enum **ImageFormat**
- enum **Rotation**
- enum **Flip**

2.2.1 Enumeration Type Documentation

2.2.1.1 enum Codec [strong]

Enum for Media Codecs supported by PhotonVoice.

Transmitted in VoiceInfo. Do not change the values of this Enum!

Enumerator

AudioOpus OPUS audio

2.3 Photon.Voice.IOS Namespace Reference

Classes

- struct AudioSessionParameters
- class AudioSessionParametersPresets

Enumerations

- enum AudioSessionCategory
- enum AudioSessionMode
- enum AudioSessionCategoryOption

2.3.1 Enumeration Type Documentation

2.3.1.1 enum AudioSessionCategory [strong]

Enumerator

Ambient Use this category for background sounds such as rain, car engine noise, etc. Mixes with other music. API_AVAILABLE(ios(3.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);

SoloAmbient Use this category for background sounds. Other music will stop playing. API_AVAILABL↩ E(ios(3.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);

Playback Use this category for music tracks. API_AVAILABLE(ios(3.0), watchos(2.0), tvos(9.0)) API_UNA↩ VAILABLE(macos);

Record Use this category when recording audio. API_AVAILABLE(ios(3.0), watchos(2.0), tvos(9.0)) API_↩ UNAVAILABLE(macos);

PlayAndRecord Use this category when recording and playing back audio. API_AVAILABLE(ios(3.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);

AudioProcessing Use this category when using a hardware codec or signal processor while not playing or recording audio. API_DEPRECATED("No longer supported", ios(3.0, 10.0)) API_UNAVAILABL↩ E(watchos, tvos) API_UNAVAILABLE(macos);

MultiRoute Use this category to customize the usage of available audio accessories and built-in audio hardware. For example, this category provides an application with the ability to use an available USB output and headphone output simultaneously for separate, distinct streams of audio data. Use of this category by an application requires a more detailed knowledge of, and interaction with, the capabilities of the available audio routes. May be used for input, output, or both. Note that not all output types and output combinations are eligible for multi-route. Input is limited to the last-in input port. Eligible inputs consist of the following↩ : AVAudioSessionPortUSBAudio, AVAudioSessionPortHeadsetMic, and AVAudioSessionPortBuiltInMic. Eligible outputs consist of the following: AVAudioSessionPortUSBAudio, AVAudioSessionPortLineOut, AVAudioSessionPortHeadphones, AVAudioSessionPortHDMI, and AVAudioSessionPortBuiltInSpeaker. Note that AVAudioSessionPortBuiltInSpeaker is only allowed to be used when there are no other eligible outputs connected. API_AVAILABLE(ios(6.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);

2.3.1.2 enum AudioSessionCategoryOption [strong]

Enumerator

MixWithOthers This allows an application to set whether or not other active audio apps will be interrupted or mixed with when your app's audio session goes active. The typical cases are: (1) AVAudioSession↩ CategoryPlayAndRecord or AVAudioSessionCategoryMultiRoute this will default to false, but can be set to true. This would allow other applications to play in the background while an app had both audio input and output enabled (2) AVAudioSessionCategoryPlayback this will default to false, but can be set to true. This would allow other applications to play in the background, but an app will still be able to play regardless of the setting of the ringer switch (3) Other categories this defaults to false and cannot be changed (that is, the mix with others setting of these categories cannot be overridden. An application must be prepared for setting this property to fail as behaviour may change in future releases. If an application changes their category, they should reassert the option (it is not sticky across category changes). MixWithOthers is only valid with AVAudioSessionCategoryPlayAndRecord, AVAudioSessionCategoryPlayback, and AVAudio↩ SessionCategoryMultiRoute

DuckOthers This allows an application to set whether or not other active audio apps will be ducked when when your app's audio session goes active. An example of this is the Nike app, which provides periodic

updates to its user (it reduces the volume of any music currently being played while it provides its status). This defaults to off. Note that the other audio will be ducked for as long as the current session is active. You will need to deactivate your audio session when you want full volume playback of the other audio. If your category is AVAudioSessionCategoryPlayback, AVAudioSessionCategoryPlayAndRecord, or AVAudio↩ SessionCategoryMultiRoute, by default the audio session will be non-mixable and non-ducking. Setting this option will also make your category mixable with others (AVAudioSessionCategoryOptionMixWith↩ Others will be set). DuckOthers is only valid with AVAudioSessionCategoryAmbient, AVAudioSession↩ CategoryPlayAndRecord, AVAudioSessionCategoryPlayback, and AVAudioSessionCategoryMultiRoute

AllowBluetooth This allows an application to change the default behaviour of some audio session categories with regards to showing bluetooth Hands-Free Profile (HFP) devices as available routes. The current category behavior is: (1) AVAudioSessionCategoryPlayAndRecord this will default to false, but can be set to true. This will allow a paired bluetooth HFP device to show up as an available route for input, while playing through the category-appropriate output (2) AVAudioSessionCategoryRecord this will default to false, but can be set to true. This will allow a paired bluetooth HFP device to show up as an available route for input (3) Other categories this defaults to false and cannot be changed (that is, enabling bluetooth for input in these categories is not allowed) An application must be prepared for setting this option to fail as behaviour may change in future releases. If an application changes their category or mode, they should reassert the override (it is not sticky across category and mode changes). AllowBluetooth is only valid with AVAudioSessionCategoryRecord and AVAudioSessionCategoryPlayAndRecord

DefaultToSpeaker This allows an application to change the default behaviour of some audio session categories with regards to the audio route. The current category behavior is: (1) AVAudioSessionCategory↩ PlayAndRecord category this will default to false, but can be set to true. this will route to Speaker (instead of Receiver) when no other audio route is connected. (2) Other categories this defaults to false and cannot be changed (that is, the default to speaker setting of these categories cannot be overridden An application must be prepared for setting this property to fail as behaviour may change in future releases. If an application changes their category, they should reassert the override (it is not sticky across category and mode changes). DefaultToSpeaker is only valid with AVAudioSessionCategoryPlayAndRecord

2.3.1.3 enum AudioSessionMode [strong]

Enumerator

Default Modes modify the audio category in order to introduce behavior that is tailored to the specific use of audio within an application. Available in iOS 5.0 and greater. The default mode API_AVAILABLE(ios(5.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);

VoiceChat Only valid with AVAudioSessionCategoryPlayAndRecord. Appropriate for Voice over IP (VoIP) applications. Reduces the number of allowable audio routes to be only those that are appropriate for VoIP applications and may engage appropriate system-supplied signal processing. Has the side effect of setting AVAudioSessionCategoryOptionAllowBluetooth API_AVAILABLE(ios(5.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);

VideoRecording Only valid with AVAudioSessionCategoryPlayAndRecord or AVAudioSessionCategory↩ Record. Modifies the audio routing options and may engage appropriate system-supplied signal processing. API_AVAILABLE(ios(5.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);

Measurement Appropriate for applications that wish to minimize the effect of system-supplied signal processing for input and/or output audio signals. API_AVAILABLE(ios(5.0), watchos(2.0), tvos(9.0)) API_UNA↩ VAILABLE(macos);

MoviePlayback Engages appropriate output signal processing for movie playback scenarios. Currently only applied during playback over built-in speaker. API_AVAILABLE(ios(6.0), watchos(2.0), tvos(9.0)) API_↩ UNAVAILABLE(macos);

VideoChat Only valid with kAudioSessionCategory_PlayAndRecord. Reduces the number of allowable audio routes to be only those that are appropriate for video chat applications. May engage appropriate system-supplied signal processing. Has the side effect of setting AVAudioSessionCategory↩ OptionAllowBluetooth and AVAudioSessionCategoryOptionDefaultToSpeaker. API_AVAILABLE(ios(7.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);

2.4 Photon.Voice.PUN Namespace Reference

Classes

- class PhotonVoiceNetwork

  This class can be used to automatically sync client states between PUN and Voice. It also sets a custom PUN Speaker factory to find the Speaker component for a character's voice. For this to work attach a PhotonVoiceView next to the PhotonView of your player's prefab.
- class PhotonVoiceView

  Component that should be attached to a networked PUN prefab that has PhotonView. It will bind remote Recorder with local Speaker of the same networked prefab. This component makes automatic voice stream routing easy for players' characters/avatars.

2.5 Photon.Voice.Unity Namespace Reference

Namespaces

- namespace UtilityScripts

Classes

- class AudioClipWrapper
- class AudioOutCapture
- interface ILoggable
- class IOSAudioForceToSpeaker
- class Logger
- class MicWrapper
- class PhotonVoiceCreatedParams
- class Recorder

  Component representing outgoing audio stream in scene.
- class RemoteVoiceLink
- class Speaker

  Component representing remote audio stream in local scene.
- class UnityAndroidAudioInAEC
- class UnityAudioOut
- class VoiceComponent
- class VoiceConnection

  Component that represents a client voice connection to Photon Servers.
- class VoiceLogger
- class WebRtcAudioDsp

2.6 Photon.Voice.Unity.UtilityScripts Namespace Reference

Classes

- class ConnectAndJoin
- class MicAmplifier
- class MicAmplifierFloat
- class MicAmplifierShort
- class PhotonVoiceLagSimulationGui
- class PhotonVoiceStatsGui

Basic GUI to show traffic and health statistics of the connection to Photon, toggled by shift+tab.

- class TestTone
- class ToneAudioReader

2.7 POpusCodec Namespace Reference

Namespaces

- namespace Enums

Classes

- class OpusDecoder
- class OpusEncoder
- class OpusException
- class **Wrapper**

2.8 POpusCodec.Enums Namespace Reference

Enumerations

- enum Bandwidth : int
- enum Channels : int
- enum **Complexity** : int
- enum Delay

  Using a duration of less than 10 ms will prevent the encoder from using the LPC or hybrid modes.
- enum **ForceChannels** : int
- enum OpusApplicationType : int
- enum **OpusStatusCode** : int
- enum **SamplingRate** : int
- enum SignalHint : int

2.8.1 Enumeration Type Documentation

2.8.1.1 enum Bandwidth : int [strong]

Enumerator

Narrowband Up to 4Khz

Mediumband Up to 6Khz

Wideband Up to 8Khz

SuperWideband Up to 12Khz

Fullband Up to 20Khz (High Definition)

2.8.1.2 enum Channels : int [strong]

Enumerator

Mono 1 Channel

Stereo 2 Channels

2.8.1.3 enum Delay [strong]

Using a duration of less than 10 ms will prevent the encoder from using the LPC or hybrid modes.

Enumerator

Delay2dot5ms 2.5ms

Delay5ms 5ms

Delay10ms 10ms

Delay20ms 20ms

Delay40ms 40ms

Delay60ms 60ms

2.8.1.4 enum OpusApplicationType : int [strong]

Enumerator

Voip Gives best quality at a given bitrate for voice signals. It enhances the input signal by high-pass filtering and emphasizing formants and harmonics. Optionally it includes in-band forward error correction to protect against packet loss. Use this mode for typical VoIP applications. Because of the enhancement, even at high bitrates the output may sound different from the input.

Audio Gives best quality at a given bitrate for most non-voice signals like music. Use this mode for music and mixed (music/voice) content, broadcast, and applications requiring less than 15 ms of coding delay.

RestrictedLowDelay Configures low-delay mode that disables the speech-optimized mode in exchange for slightly reduced delay.

2.8.1.5 enum SignalHint : int [strong]

Enumerator

Auto (default)

Voice Bias thresholds towards choosing LPC or Hybrid modes

Music Bias thresholds towards choosing MDCT modes.

Chapter 3

Class Documentation

3.1 AudioClipWrapper Class Reference

Inherits IAudioReader< float >.

Public Member Functions

- **AudioClipWrapper** (AudioClip audioClip)
- bool **Read** (float[ ] buffer)
- void **Dispose** ()

Properties

- bool **Loop** `[get, set]`
- int **SamplingRate** `[get]`
- int **Channels** `[get]`
- string **Error** `[get]`

3.2 AudioDesc Class Reference

Inherits IAudioDesc.

Public Member Functions

- **AudioDesc** (int samplingRate, int channels, string error)
- void **Dispose** ()

Properties

- int **SamplingRate** `[get]`
- int **Channels** `[get]`
- string **Error** `[get]`

3.3 AudioInEnumerator Class Reference

Inherits IDisposable.

Public Member Functions

- **AudioInEnumerator** (ILogger logger)
- void **Refresh** ()
- string **NameAtIndex** (int i)
- int **IDAtIndex** (int i)
- bool **IDIsValid** (int id)
- void **Dispose** ()

Public Attributes

- readonly bool **IsSupported** = false

Properties

- string **Error** `[get]`
- int **Count** `[get]`

3.4 AudioOutCapture Class Reference

Inherits MonoBehaviour.

Events

- Action< float[ ], int > **OnAudioFrame**

3.5 AudioSessionParameters Struct Reference

Public Member Functions

- int **CategoryOptionsToInt** ()
- override string **ToString** ()

Public Attributes

- AudioSessionCategory **Category**
- AudioSessionMode **Mode**
- AudioSessionCategoryOption[ ] **CategoryOptions**

3.6 AudioSessionParametersPresets Class Reference

Static Public Attributes

- static AudioSessionParameters **Game**
- static AudioSessionParameters **VoIP**

3.6.1 Member Data Documentation

3.6.1.1 AudioSessionParameters Game [static]

Initial value:

```
= new AudioSessionParameters()
        {
            Category = AudioSessionCategory.PlayAndRecord,
            Mode = AudioSessionMode.Default,
            CategoryOptions = new AudioSessionCategoryOption[] {
      AudioSessionCategoryOption.DefaultToSpeaker,
      AudioSessionCategoryOption.AllowBluetooth }
        }
```

3.6.1.2 AudioSessionParameters VoIP [static]

Initial value:

```
= new AudioSessionParameters()
        {
            Category = AudioSessionCategory.PlayAndRecord,
            Mode = AudioSessionMode.VoiceChat,


            CategoryOptions = new AudioSessionCategoryOption[] {
      AudioSessionCategoryOption.AllowBluetooth }
         }
```

3.7 AudioStreamPlayer< T > Class Template Reference

Inherits IAudioOut< T >.

Public Member Functions

- **AudioStreamPlayer** (ILogger logger, ISyncAudioOut< T > audioOut, string logPrefix, bool debugInfo)
- void **Start** (int frequency, int channels, int frameSamples, int playDelayMs)
- void **Service** ()
- void **Push** (T[] frame)
- void **Stop** ()

Properties

- int **Lag** [get]
- bool **IsPlaying** [get]

3.8 AudioUtil Class Reference

Collection of Audio Utility functions and classes.

Classes

- interface ILevelMeter

  Audio Level Metering interface.
- interface IVoiceDetector

Voice Activity Detector interface.

- class LevelMeter

  Audio Level Meter.

- class LevelMeterDummy

  Dummy Audio Level Meter that doesn't actually do anything.

- class LevelMeterFloat

  LevelMeter specialization for float audio.

- class LevelMeterShort

  LevelMeter specialization for short audio.

- class Resampler

  Sample-rate conversion Audio Processor.

- class ToneAudioPusher

  IAudioPusher that provides a constant tone signal.

- class ToneAudioReader

  IAudioReader that provides a constant tone signal.

- class VoiceDetector

  Simple voice activity detector triggered by signal level.

- class VoiceDetectorCalibration

  Calibration Utility for Voice Detector

- class VoiceDetectorDummy

  Dummy VoiceDetector that doesn't actually do anything.

- class VoiceDetectorFloat

  VoiceDetector specialization for float audio.

- class VoiceDetectorShort

  VoiceDetector specialization for float audio.

- class VoiceLevelDetectCalibrate

  Utility Audio Processor Voice Detection Calibration.

Static Public Member Functions

- static void Resample< T > (T[ ] src, T[ ] dst, int dstCount, int channels)

  Resample audio data so that the complete src buffer fits into dstCount samples in the dst buffer.

- static void ResampleAndConvert (short[ ] src, float[ ] dst, int dstCount, int channels)

  Resample audio data so that the complete src buffer fits into dstCount samples in the dst buffer, and convert short to float samples along the way.

- static void ResampleAndConvert (float[ ] src, short[ ] dst, int dstCount, int channels)

  Resample audio data so that the complete src buffer fits into dstCount samples in the dst buffer, and convert float to short samples along the way.

- static void Convert (float[ ] src, short[ ] dst, int dstCount)

  Convert audio buffer from float to short samples.

- static void Convert (short[ ] src, float[ ] dst, int dstCount)

  Convert audio buffer from short to float samples.

- static void ForceToStereo< T > (T[ ] src, T[ ] dst, int srcChannels)

  Convert audio buffer with arbitrary number of channels to stereo.

3.8.1 Detailed Description

Collection of Audio Utility functions and classes.

3.8.2 Member Function Documentation

3.8.2.1 static void Convert (float[ ] *src,* short[ ] *dst,* int *dstCount*) [static]

Convert audio buffer from float to short samples.

Parameters

src	Source buffer.
dst	Destination buffer.
dstCount	Size of destination buffer (in total samples), source buffer must be of same length or longer.

3.8.2.2 static void Convert (short[ ] *src,* float[ ] *dst,* int *dstCount*) [static]

Convert audio buffer from short to float samples.

Parameters

src	Source buffer.
dst	Destination buffer.
dstCount	Size of destination buffer (in total samples), source buffer must be of same length or longer.

3.8.2.3 static void ForceToStereo< T > (T[ ] *src,* T[ ] *dst,* int *srcChannels*) [static]

Convert audio buffer with arbitrary number of channels to stereo.

For mono sources (srcChannels==1), the signal will be copied to both Left and Right stereo channels. For all others, the first two available channels will be used, any other channels will be discarded.

Parameters

src	Source buffer.
dst	Destination buffer.
srcChannels	Number of (interleaved) channels in src.

3.8.2.4 static void Resample< T > (T[ ] *src,* T[ ] *dst,* int *dstCount,* int *channels*) [static]

Resample audio data so that the complete src buffer fits into dstCount samples in the dst buffer.

This implements a primitive nearest-neighbor resampling algorithm for an arbitrary number of channels.

Parameters

src	Source buffer.
dst	Destination buffer.
dstCount	Target size of destination buffer (in samples per channel).
channels	Number of channels in the signal (1=mono, 2=stereo). Must be > 0.

3.8.2.5 static void ResampleAndConvert (short[ ] *src,* float[ ] *dst,* int *dstCount,* int *channels*) [static]

Resample audio data so that the complete src buffer fits into dstCount samples in the dst buffer, and convert short to float samples along the way.

This implements a primitive nearest-neighbor resampling algorithm for an arbitrary number of channels.

Parameters

src	Source buffer.
dst	Destination buffer.

dstCount	Target size of destination buffer (in samples per channel).
channels	Number of channels in the signal (1=mono, 2=stereo). Must be > 0.

3.8.2.6 static void ResampleAndConvert (float[ ] *src,* short[ ] *dst,* int *dstCount,* int *channels*) [static]

Resample audio data so that the complete src buffer fits into dstCount samples in the dst buffer, and convert float to short samples along the way.

This implements a primitive nearest-neighbor resampling algorithm for an arbitrary number of channels.

Parameters

src	Source buffer.
dst	Destination buffer.
dstCount	Target size of destination buffer (in samples per channel).
channels	Number of channels in the signal (1=mono, 2=stereo). Must be > 0.

3.9 BufferReaderPushAdapter< T > Class Template Reference

Simple BufferReaderPushAdapterBase implementation using a single buffer, using synchronous LocalVoice.Push↩ Data

Inherits BufferReaderPushAdapterBase< T >.

Public Member Functions

- BufferReaderPushAdapter (LocalVoice localVoice, IDataReader< T > reader)

  Create a new BufferReaderPushAdapter instance
- override void Service (LocalVoice localVoice)

  Do the actual data read/push.

Protected Attributes

- T[ ] **buffer**

3.9.1 Detailed Description

Simple BufferReaderPushAdapterBase implementation using a single buffer, using synchronous LocalVoice.Push↩ Data

3.9.2 Constructor & Destructor Documentation

3.9.2.1 BufferReaderPushAdapter (LocalVoice *localVoice,* IDataReader< T > *reader*)

Create a new BufferReaderPushAdapter instance

Parameters

localVoice	LocalVoice instance to push data to.

reader	DataReader to read from.

3.9.3 Member Function Documentation

3.9.3.1 override void Service (LocalVoice *localVoice*) [virtual]

Do the actual data read/push.

Parameters

localVoice	LocalVoice instance to push data to.

Implements BufferReaderPushAdapterBase< T >.

3.10 BufferReaderPushAdapterAsyncPool< T > Class Template Reference

BufferReaderPushAdapter implementation using asynchronous LocalVoice.PushDataAsync.

Inherits BufferReaderPushAdapterBase< T >.

Public Member Functions

- BufferReaderPushAdapterAsyncPool (LocalVoice localVoice, IDataReader< T > reader)
  Create a new BufferReaderPushAdapter instance
- override void Service (LocalVoice localVoice)
  Do the actual data read/push.

Additional Inherited Members

3.10.1 Detailed Description

BufferReaderPushAdapter implementation using asynchronous LocalVoice.PushDataAsync.

Acquires a buffer from pool before each Read, releases buffer after last Read (brings Acquire/Release overhead).

Expects localVoice to be a LocalVoiceFramed<T> of same T.

3.10.2 Constructor & Destructor Documentation

3.10.2.1 BufferReaderPushAdapterAsyncPool (LocalVoice *localVoice,* IDataReader< T > *reader*)

Create a new BufferReaderPushAdapter instance

Parameters

localVoice	LocalVoice instance to push data to.
reader	DataReader to read from.

3.10.3 Member Function Documentation

3.10.3.1 override void Service (LocalVoice *localVoice*) [virtual]

Do the actual data read/push.

Parameters

localVoice	LocalVoice instance to push data to. Must be a LocalVoiceFramed<T> of same T.

Implements BufferReaderPushAdapterBase< T >.

3.11 BufferReaderPushAdapterAsyncPoolCopy< T > Class Template Reference

BufferReaderPushAdapter implementation using asynchronous LocalVoice.PushDataAsync and data copy.

Inherits BufferReaderPushAdapterBase< T >.

Public Member Functions

- BufferReaderPushAdapterAsyncPoolCopy (LocalVoice localVoice, IDataReader< T > reader)

  Create a new BufferReaderPushAdapter instance
- override void Service (LocalVoice localVoice)

  Do the actual data read/push.

Protected Attributes

- T[ ] **buffer**

3.11.1 Detailed Description

BufferReaderPushAdapter implementation using asynchronous LocalVoice.PushDataAsync and data copy.

Reads data to preallocated buffer, copies it to buffer from pool before pushing. Compared with , this avoids one pool Acquire/Release cycle at the cost of a buffer copy. Expects localVoice to be a LocalVoiceFramed<T> of same T.

3.11.2 Constructor & Destructor Documentation

3.11.2.1 BufferReaderPushAdapterAsyncPoolCopy (LocalVoice *localVoice,* IDataReader< T > *reader*)

Create a new BufferReaderPushAdapter instance

Parameters

localVoice	LocalVoice instance to push data to.
reader	DataReader to read from.

3.11.3 Member Function Documentation

3.11.3.1 override void Service (LocalVoice *localVoice*) [virtual]

Do the actual data read/push.

Parameters

localVoice	LocalVoice instance to push data to. Must be a LocalVoiceFramed<T> of same T.

Implements BufferReaderPushAdapterBase< T >.

3.12 BufferReaderPushAdapterAsyncPoolFloatToShort Class Reference

BufferReaderPushAdapter implementation using asynchronous LocalVoice.PushDataAsync, converting float samples to short.

Inherits BufferReaderPushAdapterBase< float >.

Public Member Functions

- BufferReaderPushAdapterAsyncPoolFloatToShort (LocalVoice localVoice, IDataReader< float > reader)

  Create a new BufferReaderPushAdapter instance
- override void Service (LocalVoice localVoice)

  Do the actual data read/push.

Additional Inherited Members

3.12.1 Detailed Description

BufferReaderPushAdapter implementation using asynchronous LocalVoice.PushDataAsync, converting float samples to short.

This adapter works exactly like BufferReaderPushAdapterAsyncPool, but it converts float samples to short. Acquires a buffer from pool before each Read, releases buffer after last Read.

Expects localVoice to be a LocalVoiceFramed<T> of same T.

3.12.2 Constructor & Destructor Documentation

3.12.2.1 BufferReaderPushAdapterAsyncPoolFloatToShort (LocalVoice *localVoice,* IDataReader< float > *reader*)

Create a new BufferReaderPushAdapter instance

Parameters

localVoice	LocalVoice instance to push data to.
reader	DataReader to read from.

3.12.3 Member Function Documentation

3.12.3.1 override void Service (LocalVoice *localVoice*) [virtual]

Do the actual data read/push.

Parameters

localVoice	LocalVoice instance to push data to. Must be a LocalVoiceFramed<T> of same T.

Implements BufferReaderPushAdapterBase< float >.

3.13 BufferReaderPushAdapterBase< T > Class Template Reference

Adapter base class to move data by reading from IDataReader.Read and pushing to LocalVoice.

Inherits IServiceable.

Inherited by BufferReaderPushAdapter< T >, BufferReaderPushAdapterAsyncPool< T >, and BufferReaderPushAdapterAsyncPoolCopy< T >.

Public Member Functions

- abstract void Service (LocalVoice localVoice)

  Do the actual data read/push.
- BufferReaderPushAdapterBase (IDataReader< T > reader)

  Create a new BufferReaderPushAdapterBase instance
- void Dispose ()

  Release resources associated with this instance.

Protected Attributes

- IDataReader< T > **reader**

3.13.1 Detailed Description

Adapter base class to move data by reading from IDataReader.Read and pushing to LocalVoice.

Use this with a LocalVoice of same T type.

3.13.2 Constructor & Destructor Documentation

3.13.2.1 BufferReaderPushAdapterBase (IDataReader< T > *reader*)

Create a new BufferReaderPushAdapterBase instance

Parameters

reader	DataReader to read from.

3.13.3 Member Function Documentation

3.13.3.1 void Dispose ( )

Release resources associated with this instance.

3.13.3.2 abstract void Service (LocalVoice *localVoice*) [pure virtual]

Do the actual data read/push.

Parameters

localVoice	LocalVoice instance to push data to.

Implements IServiceable.

Implemented in BufferReaderPushAdapterAsyncPoolFloatToShort, BufferReaderPushAdapterAsyncPoolCopy< T >, BufferReaderPushAdapterAsyncPool< T >, and BufferReaderPushAdapter< T >.

3.14 WebRTCAudioLib.ConfigParam Struct Reference

Public Attributes

- const int **AEC_DELAY_AGNOSTIC** = 12
- const int **AEC_EXTENDED_FILTER** = 13
- const int **AGC_EXPERIMENTAL** = 53
- const int **AGC_EXPERIMENTAL_STARTUP_MIN_VOLUME** = 54
- const int **AGC_EXPERIMENTAL_CLIP_LEVEL_MIN** = 55

3.15 ConnectAndJoin Class Reference

Inherits MonoBehaviour, IConnectionCallbacks, and IMatchmakingCallbacks.

Public Member Functions

- void **ConnectNow** ()
- void **OnCreatedRoom** ()
- void **OnCreateRoomFailed** (short returnCode, string message)
- void **OnFriendListUpdate** (List< FriendInfo > friendList)
- void **OnJoinedRoom** ()
- void **OnJoinRandomFailed** (short returnCode, string message)
- void **OnJoinRoomFailed** (short returnCode, string message)
- void **OnLeftRoom** ()
- void **OnConnected** ()
- void **OnConnectedToMaster** ()
- void **OnDisconnected** (DisconnectCause cause)
- void **OnRegionListReceived** (RegionHandler regionHandler)
- void **OnCustomAuthenticationResponse** (Dictionary< string, object > data)
- void **OnCustomAuthenticationFailed** (string debugMessage)

Public Attributes

- bool **RandomRoom** = true
- string **RoomName**

Properties

- bool **IsConnected** `[get]`

3.16 OpusCodec.Decoder< T > Class Template Reference

Inherits IDecoder.

Public Member Functions

- **Decoder** (Action< T[ ]> output, ILogger logger)
- void Open (VoiceInfo i)

  Open (initialize) the decoder.
- void **Dispose** ()
- void Input (byte[ ] buf)

  Consumes the given encoded data.

Protected Member Functions

- abstract T[ ] **decodeTyped** (byte[ ] buf)

Protected Attributes

- OpusDecoder **decoder**

Properties

- string **Error** `[get]`

3.16.1 Member Function Documentation

3.16.1.1 void Input (byte[ ] *buf*)

Consumes the given encoded data.

Implements IDecoder.

3.16.1.2 void Open (VoiceInfo *info*)

Open (initialize) the decoder.

Parameters

info	Properties of the data stream to decode.

Implements IDecoder.

3.17 OpusCodec.DecoderFactory Class Reference

Static Public Member Functions

- static IEncoder **Create**< **T** > (VoiceInfo i, ILogger logger)

3.18 OpusCodec.DecoderFloat Class Reference

Inherits OpusCodec.Decoder< float >.

Public Member Functions

- **DecoderFloat** (Action< float[ ]> output, ILogger logger)

Protected Member Functions

- override float[ ] **decodeTyped** (byte[ ] buf)

Additional Inherited Members

3.19 OpusCodec.DecoderShort Class Reference

Inherits OpusCodec.Decoder< short >.

Public Member Functions

- **DecoderShort** (Action< short[ ]> output, ILogger logger)

Protected Member Functions

- override short[ ] **decodeTyped** (byte[ ] buf)

Additional Inherited Members

3.20 OpusCodec.Encoder< T > Class Template Reference

Inherits IEncoderDirect< T[ ]>.

Public Member Functions

- void **Input** (T[ ] buf)
- ArraySegment< byte > **DequeueOutput** ()
- void **Dispose** ()

Protected Member Functions

- **Encoder** (VoiceInfo i, ILogger logger)
- abstract ArraySegment< byte > **encodeTyped** (T[ ] buf)

Protected Attributes

- OpusEncoder **encoder**
- bool **disposed**

Properties

- string **Error** `[get]`
- Action< ArraySegment< byte > > **Output** `[get, set]`

3.21 OpusCodec.EncoderFloat Class Reference

Inherits OpusCodec.Encoder< float >.

Protected Member Functions

- override ArraySegment< byte > **encodeTyped** (float[ ] buf)

Additional Inherited Members

3.22 OpusCodec.EncoderShort Class Reference

Inherits OpusCodec.Encoder< short >.

Protected Member Functions

- override ArraySegment< byte > **encodeTyped** (short[ ] buf)

Additional Inherited Members

3.23 OpusCodec.Factory Class Reference

Static Public Member Functions

- static IEncoder **CreateEncoder**< **B** > (VoiceInfo i, ILogger logger)

3.24 FactoryPrimitiveArrayPool< T > Class Template Reference

PrimitiveArrayPool<T> as wrapped in object factory interface.

Inherits ObjectFactory< T[ ], int >.

Public Member Functions

- **FactoryPrimitiveArrayPool** (int capacity, string name)
- **FactoryPrimitiveArrayPool** (int capacity, string name, int info)
- T[ ] **New** ()
- T[ ] **New** (int size)
- void **Free** (T[ ] obj)
- void **Free** (T[ ] obj, int info)
- void **Dispose** ()

Properties

- int **Info** `[get]`

3.24.1 Detailed Description

PrimitiveArrayPool<T> as wrapped in object factory interface.

Template Parameters

T	Array element type.

3.25 FactoryReusableArray< T > Class Template Reference

Array factory returnig the same array instance as long as it requested with the same array length. If length changes, new array instance created.

Inherits ObjectFactory< T[], int >.

Public Member Functions

- **FactoryReusableArray** (int size)
- T[] **New** ()
- T[] **New** (int size)
- void **Free** (T[] obj)
- void **Free** (T[] obj, int info)
- void **Dispose** ()

Properties

- int **Info** `[get]`

3.25.1 Detailed Description

Array factory returnig the same array instance as long as it requested with the same array length. If length changes, new array instance created.

Template Parameters

T	Array element type.

3.26 Framer< T > Class Template Reference

Utility class to re-frame audio packets.

Public Member Functions

- Framer (int frameSize)

  Create new Framer instance.
- int Count (int bufLen)

  Get the number of frames available after adding bufLen samples.
- IEnumerable< T[]> Frame (T[] buf)

  Append arbitrary-sized buffer and return available full frames.

3.26.1 Detailed Description

Utility class to re-frame audio packets.

3.26.2 Constructor & Destructor Documentation

3.26.2.1 Framer (int *frameSize*)

Create new Framer instance.

3.26.3 Member Function Documentation

3.26.3.1 int Count (int *bufLen*)

Get the number of frames available after adding bufLen samples.

Parameters

bufLen	Number of samples that would be added.

Returns

Number of full frames available when adding bufLen samples.

3.26.3.2 IEnumerable<T[]> Frame (T[] *buf*)

Append arbitrary-sized buffer and return available full frames.

Parameters

buf	Array of samples to add.

Returns

Enumerator of full frames (might be none).

3.27 IAudioDesc Interface Reference

Audio Source interface.

Inherits IDisposable.

Inherited by AudioDesc, IAudioPusher< T >, and IAudioReader< T >.

Properties

- int SamplingRate `[get]`

  Sampling rate of the audio signal (in Hz).
- int Channels `[get]`

  Number of channels in the audio signal.
- string Error `[get]`

  If not null, audio object is in invalid state.

3.27.1 Detailed Description

Audio Source interface.

3.27.2 Property Documentation

3.27.2.1 int Channels [get]

Number of channels in the audio signal.

3.27.2.2 string Error [get]

If not null, audio object is in invalid state.

3.27.2.3 int SamplingRate [get]

Sampling rate of the audio signal (in Hz).

3.28 IAudioOut< T > Interface Template Reference

Inherited by AudioStreamPlayer< T >, and ISyncAudioOut< T >.

Public Member Functions

- void **Start** (int frequency, int channels, int frameSamplesPerChannel, int playDelayMs)
- void **Stop** ()
- void **Push** (T[] frame)
- void **Service** ()

Properties

- bool **IsPlaying** [get]
- int **Lag** [get]

3.29 IAudioPusher< T > Interface Template Reference

Audio Pusher interface.

Inherits IAudioDesc.

Inherited by AudioUtil.ToneAudioPusher< T >.

Public Member Functions

- void SetCallback (Action< T[]> callback, ObjectFactory< T[], int > bufferFactory)

  Set the callback function used for pushing data.

Additional Inherited Members

3.29.1 Detailed Description

Audio Pusher interface.

Opposed to an IAudioReader (which will deliver audio data when it is "pulled"), an IAudioPusher will push its audio data whenever it is ready,

3.29.2 Member Function Documentation

3.29.2.1 void SetCallback (Action< T[ ]> *callback,* ObjectFactory< T[ ], int > *bufferFactory*)

Set the callback function used for pushing data.

Parameters

callback	Callback function to use.
localVoice	Outgoing audio stream, for context.

Implemented in AudioUtil.ToneAudioPusher< T >.

3.30 IAudioReader< T > Interface Template Reference

Audio Reader interface.

Inherits IDataReader< T >, and IAudioDesc.

Inherited by AudioUtil.ToneAudioReader< T >.

Additional Inherited Members

3.30.1 Detailed Description

Audio Reader interface.

Opposed to an IAudioPusher (which will push its audio data whenever it is ready), an IAudioReader will deliver audio data when it is "pulled" (it's Read function is called).

3.31 IDataReader< T > Interface Template Reference

Interface for pulling data, in case this is more appropriate than pushing it.

Inherits IDisposable.

Inherited by IAudioReader< T >.

Public Member Functions

- bool Read (T[ ] buffer)

  Fill full given frame buffer with source uncompressed data or return false if not enough such data.

3.31.1 Detailed Description

Interface for pulling data, in case this is more appropriate than pushing it.

3.31.2 Member Function Documentation

3.31.2.1 bool Read (T[ ] *buffer*)

Fill full given frame buffer with source uncompressed data or return false if not enough such data.

Parameters

buffer	Buffer to fill.

Returns

True if buffer was filled successfully, false otherwise.

Implemented in AudioUtil.ToneAudioReader< T >.

3.32 IDecoder Interface Reference

Generic decoder interface.

Inherits IDisposable.

Inherited by IDecoderQueuedOutputImageNative, and OpusCodec.Decoder< T >.

Public Member Functions

- void Open (VoiceInfo info)

  Open (initialize) the decoder.
- void Input (byte[ ] buf)

  Consumes the given encoded data.

Properties

- string Error `[get]`

  If not null, the object is in invalid state.

3.32.1 Detailed Description

Generic decoder interface.

3.32.2 Member Function Documentation

3.32.2.1 void Input (byte[ ] *buf*)

Consumes the given encoded data.

Implemented in OpusCodec.Decoder< T >.

3.32.2.2 void Open (VoiceInfo *info*)

Open (initialize) the decoder.

Parameters

info	Properties of the data stream to decode.

Implemented in OpusCodec.Decoder< T >.

3.32.3 Property Documentation

3.32.3.1 string Error [get]

If not null, the object is in invalid state.

3.33 IDecoderQueuedOutputImageNative Interface Reference

Inherits IDecoder.

Properties

- ImageFormat **OutputImageFormat** [get, set]
- Flip **OutputImageFlip** [get, set]
- Func< int, int, IntPtr > **OutputImageBufferGetter** [get, set]

Additional Inherited Members

3.34 IEncoder Interface Reference

Generic encoder interface.

Inherits IDisposable.

Inherited by IEncoderDirect< B >.

Public Member Functions

- ArraySegment< byte > DequeueOutput ()

  Returns next encoded data frame (if such output supported).

Properties

- string Error [get]

  If not null, the object is in invalid state.
- Action< ArraySegment< byte > > Output [get, set]

  Set callback encoder calls on each encoded data frame (if such output supported).

3.34.1 Detailed Description

Generic encoder interface.

Depending on implementation, encoder should either call Output on eaach data frame or return next data frame in DequeueOutput() call.

3.34.2 Member Function Documentation

3.34.2.1 ArraySegment<byte> DequeueOutput ( )

Returns next encoded data frame (if such output supported).

3.34.3 Property Documentation

3.34.3.1 string Error [get]

If not null, the object is in invalid state.

3.34.3.2 Action<ArraySegment<byte> > Output [get], [set]

Set callback encoder calls on each encoded data frame (if such output supported).

3.35 IEncoderDirect< B > Interface Template Reference

Interface for an encoder which consumes input data via explicit call.

Inherits IEncoder.

Public Member Functions

- void Input (B buf)

  Consumes the given raw data.

Additional Inherited Members

3.35.1 Detailed Description

Interface for an encoder which consumes input data via explicit call.

3.35.2 Member Function Documentation

3.35.2.1 void Input (B *buf*)

Consumes the given raw data.

Parameters

buf	Array containing raw data (e.g. audio samples).

3.36 AudioUtil.ILevelMeter Interface Reference

Audio Level Metering interface.

Inherited by AudioUtil.LevelMeter< T >, and AudioUtil.LevelMeterDummy.

Public Member Functions

- void ResetAccumAvgPeakAmp ()

  Reset AccumAvgPeakAmp.

Properties

- float CurrentAvgAmp [get]

  Average amplitude value over last half second.
- float CurrentPeakAmp [get]

  Maximum amplitude value over last half second sec.
- float AccumAvgPeakAmp [get]

  Average of CurrentPeakAmps since last reset.

3.36.1 Detailed Description

Audio Level Metering interface.

3.36.2 Member Function Documentation

3.36.2.1 void ResetAccumAvgPeakAmp ( )

Reset AccumAvgPeakAmp.

Implemented in AudioUtil.LevelMeter< T >, and AudioUtil.LevelMeterDummy.

3.36.3 Property Documentation

3.36.3.1 float AccumAvgPeakAmp [get]

Average of CurrentPeakAmps since last reset.

3.36.3.2 float CurrentAvgAmp [get]

Average amplitude value over last half second.

3.36.3.3 float CurrentPeakAmp [get]

Maximum amplitude value over last half second sec.

3.37 ILocalVoiceAudio Interface Reference

Interface for an outgoing audio stream.

Inherited by LocalVoiceAudio< T >, and LocalVoiceAudioDummy.

Public Member Functions

- void VoiceDetectorCalibrate (int durationMs, Action< float > onCalibrated=null)

  Trigger voice detector calibration process.

Properties

- AudioUtil.IVoiceDetector VoiceDetector [get]

  The VoiceDetector in use.
- AudioUtil.ILevelMeter LevelMeter [get]

  The LevelMeter utility in use.
- bool VoiceDetectorCalibrating [get]

  If true, voice detector calibration is in progress.

3.37.1 Detailed Description

Interface for an outgoing audio stream.

A LocalVoice always brings a LevelMeter and a VoiceDetector, which you can access using this interface.

3.37.2 Member Function Documentation

3.37.2.1 void VoiceDetectorCalibrate (int *durationMs*, Action< float > *onCalibrated* = null)

Trigger voice detector calibration process.

While calibrating, keep silence. Voice detector sets threshold based on measured backgroud noise level.

Parameters

durationMs	Duration of calibration (in milliseconds).
onCalibrated	Called when calibration is complete. Parameter is new threshold value.

Implemented in LocalVoiceAudioDummy, and LocalVoiceAudio< T >.

3.37.3 Property Documentation

3.37.3.1 AudioUtil.ILevelMeter LevelMeter [get]

The LevelMeter utility in use.

3.37.3.2 AudioUtil.IVoiceDetector VoiceDetector [get]

The VoiceDetector in use.

Use it to enable or disable voice detector and set its parameters.

3.37.3.3 bool VoiceDetectorCalibrating [get]

If true, voice detector calibration is in progress.

3.38 ILoggable Interface Reference

Inherited by VoiceComponent, and VoiceConnection.

Properties

- DebugLevel **LogLevel** [get, set]
- VoiceLogger **Logger** [get]

3.39 ILogger Interface Reference

Inherited by IVoiceTransport, Logger, and VoiceLogger.

Public Member Functions

- void **LogError** (string fmt, params object[ ] args)
- void **LogWarning** (string fmt, params object[ ] args)
- void **LogInfo** (string fmt, params object[ ] args)
- void **LogDebug** (string fmt, params object[ ] args)

3.40 ImageBufferInfo Class Reference

Public Member Functions

- **ImageBufferInfo** (int width, int height, int[ ] stride, ImageFormat format)

Properties

- int **Width** `[get]`
- int **Height** `[get]`
- int[ ] **Stride** `[get]`
- ImageFormat **Format** `[get]`
- Rotation **Rotation** `[get, set]`
- Flip **Flip** `[get, set]`

3.41 ImageBufferNative Class Reference

Inherited by ImageBufferNativeAlloc, and ImageBufferNativeGCHandleSinglePlane.

Public Member Functions

- **ImageBufferNative** (ImageBufferInfo info)
- virtual void **Release** ()
- virtual void **Dispose** ()

Properties

- ImageBufferInfo **Info** `[get, protected set]`
- IntPtr[ ] **Planes** `[get, protected set]`

3.42 ImageBufferNativeAlloc Class Reference

Inherits ImageBufferNative, and IDisposable.

Public Member Functions

- **ImageBufferNativeAlloc** (ImageBufferNativePool< ImageBufferNativeAlloc > pool, ImageBufferInfo info)
- override void **Release** ()
- override void **Dispose** ()

Additional Inherited Members

3.43 ImageBufferNativeGCHandleSinglePlane Class Reference

Inherits ImageBufferNative, and IDisposable.

Public Member Functions

- **ImageBufferNativeGCHandleSinglePlane** (ImageBufferNativePool< ImageBufferNativeGCHandleSingle↩ Plane > pool, ImageBufferInfo info)
- void **PinPlane** (byte[ ] plane)
- override void **Release** ()
- override void **Dispose** ()

Additional Inherited Members

3.44 ImageBufferNativePool< T > Class Template Reference

Inherits ObjectPool< T, ImageBufferInfo >.

Public Member Functions

- delegate T **Factory** (ImageBufferNativePool< T > pool, ImageBufferInfo info)
- **ImageBufferNativePool** (int capacity, Factory factory, string name)
- **ImageBufferNativePool** (int capacity, Factory factory, string name, ImageBufferInfo info)

Protected Member Functions

- override T **createObject** (ImageBufferInfo info)
- override void **destroyObject** (T obj)
- override bool **infosMatch** (ImageBufferInfo i0, ImageBufferInfo i1)

Additional Inherited Members

3.45 ImageInputBuf Struct Reference

Public Attributes

- IntPtr[ ] **Buf**
- int **Width**
- int **Height**
- int[ ] **Stride**
- ImageFormat **ImageFormat**
- Rotation **Rotation**
- Flip **Flip**

3.46 ImageOutputBuf Struct Reference

Public Attributes

- IntPtr **Buf**
- int **Width**
- int **Height**
- int **Stride**

3.47 IOSAudioForceToSpeaker Class Reference

Inherits MonoBehaviour.

3.48 IProcessor< T > Interface Template Reference

Audio Processor interface.

Inherits IDisposable.

Inherited by AudioUtil.LevelMeter< T >, AudioUtil.Resampler< T >, AudioUtil.VoiceDetector< T >, AudioUtil.↩ VoiceDetectorCalibration< T >, and AudioUtil.VoiceLevelDetectCalibrate< T >.

Public Member Functions

- T[ ] Process (T[ ] buf)

  Process a frame of audio data.

3.48.1 Detailed Description

Audio Processor interface.

3.48.2 Member Function Documentation

3.48.2.1 T [ ] Process (T[ ] *buf*)

Process a frame of audio data.

Parameters

buf	Buffer containing input audio data

Returns

Buffer containing output audio data

Implemented in AudioUtil.VoiceLevelDetectCalibrate< T >, AudioUtil.VoiceDetector< T >, AudioUtil.Voice↩ DetectorCalibration< T >, AudioUtil.LevelMeter< T >, and AudioUtil.Resampler< T >.

3.49 IServiceable Interface Reference

Interface for classes that want their Service() function to be called regularly in the context of a LocalVoice.

Inherited by BufferReaderPushAdapterBase< T >.

Public Member Functions

- void Service (LocalVoice localVoice)

  Service function that should be called regularly.

3.49.1 Detailed Description

Interface for classes that want their Service() function to be called regularly in the context of a LocalVoice.

3.49.2 Member Function Documentation

3.49.2.1 void Service (LocalVoice *localVoice*)

Service function that should be called regularly.

Implemented in BufferReaderPushAdapterAsyncPoolCopy< T >, BufferReaderPushAdapterAsyncPool< T >, BufferReaderPushAdapter< T >, and BufferReaderPushAdapterBase< T >.

3.50 ISyncAudioOut< T > Interface Template Reference

Inherits IAudioOut< T >.

Public Member Functions

- void **Pause** ()
- void **UnPause** ()

Properties

- int **PlaySamplePos** `[get, set]`

3.51 AudioUtil.IVoiceDetector Interface Reference

Voice Activity Detector interface.

Inherited by AudioUtil.VoiceDetector< T >, and AudioUtil.VoiceDetectorDummy.

Properties

- bool On `[get, set]`

  If true, voice detection enabled.
- float Threshold `[get, set]`

  Voice detected as soon as signal level exceeds threshold.
- bool Detected `[get]`

  If true, voice detected.
- DateTime DetectedTime `[get]`

  Last time when switched to detected state.
- int ActivityDelayMs `[get, set]`

  Keep detected state during this time after signal level dropped below threshold.

Events

- Action OnDetected

  Called when switched to detected state.

3.51.1 Detailed Description

Voice Activity Detector interface.

3.51.2 Property Documentation

3.51.2.1 int ActivityDelayMs [get], [set]

Keep detected state during this time after signal level dropped below threshold.

3.51.2.2 bool Detected [get]

If true, voice detected.

3.51.2.3 DateTime DetectedTime [get]

Last time when switched to detected state.

3.51.2.4 bool On [get], [set]

If true, voice detection enabled.

3.51.2.5 float Threshold [get], [set]

Voice detected as soon as signal level exceeds threshold.

3.51.3 Event Documentation

3.51.3.1 Action OnDetected

Called when switched to detected state.

3.52 IVoiceTransport Interface Reference

Inherits ILogger.

Inherited by LoadBalancingTransport.

Public Member Functions

- bool **IsChannelJoined** (int channelId)
- void **SendVoicesInfo** (IEnumerable< LocalVoice > voices, int channelId, int targetPlayerId)
- void **SendVoiceRemove** (LocalVoice voice, int channelId, int targetPlayerId)
- void **SendFrame** (ArraySegment< byte > data, byte evNumber, byte voiceId, int channelId, LocalVoice localVoice)

- string **ChannelIdStr** (int channelId)
- string **PlayerIdStr** (int playerId)
- void **SetDebugEchoMode** (LocalVoice v)

3.53 AudioUtil.LevelMeter< T > Class Template Reference

Audio Level Meter.

Inherits IProcessor< T >, and AudioUtil.ILevelMeter.

Public Member Functions

- void ResetAccumAvgPeakAmp ()

  Reset AccumAvgPeakAmp.
- abstract T[ ] Process (T[ ] buf)

  Process a frame of audio data.
- void **Dispose** ()

Protected Attributes

- float **ampSum**
- float **ampPeak**
- int **bufferSize**
- float[ ] **prevValues**
- int **prevValuesHead**
- float **accumAvgPeakAmpSum**
- int **accumAvgPeakAmpCount**

Properties

- float **CurrentAvgAmp** `[get]`
- float **CurrentPeakAmp** `[get, protected set]`
- float **AccumAvgPeakAmp** `[get]`

3.53.1 Detailed Description

Audio Level Meter.

3.53.2 Member Function Documentation

3.53.2.1 abstract T [ ] Process (T[ ] *buf*) `[pure virtual]`

Process a frame of audio data.

Parameters

buf	Buffer containing input audio data

Returns

Buffer containing output audio data

Implements IProcessor< T >.

3.53.2.2 void ResetAccumAvgPeakAmp ( )

Reset AccumAvgPeakAmp.

Implements AudioUtil.ILevelMeter.

3.54 AudioUtil.LevelMeterDummy Class Reference

Dummy Audio Level Meter that doesn't actually do anything.

Inherits AudioUtil.ILevelMeter.

Public Member Functions

- void ResetAccumAvgPeakAmp ()
  Reset AccumAvgPeakAmp.

Properties

- float **CurrentAvgAmp** `[get]`
- float **CurrentPeakAmp** `[get]`
- float **AccumAvgPeakAmp** `[get]`

3.54.1 Detailed Description

Dummy Audio Level Meter that doesn't actually do anything.

3.54.2 Member Function Documentation

3.54.2.1 void ResetAccumAvgPeakAmp ( )

Reset AccumAvgPeakAmp.

Implements AudioUtil.ILevelMeter.

3.55 AudioUtil.LevelMeterFloat Class Reference

LevelMeter specialization for float audio.

Inherits AudioUtil.LevelMeter< float >.

Public Member Functions

- LevelMeterFloat (int samplingRate, int numChannels)
  Create new LevelMeterFloat instance.
- override float[ ] **Process** (float[ ] buf)

Additional Inherited Members

3.55.1 Detailed Description

LevelMeter specialization for float audio.

3.55.2 Constructor & Destructor Documentation

3.55.2.1 LevelMeterFloat (int *samplingRate,* int *numChannels*)

Create new LevelMeterFloat instance.

Parameters

samplingRate	Sampling rate of the audio signal (in Hz).
numChannels	Number of channels in the audio signal.

3.56 AudioUtil.LevelMeterShort Class Reference

LevelMeter specialization for short audio.

Inherits AudioUtil.LevelMeter< short >.

Public Member Functions

- LevelMeterShort (int samplingRate, int numChannels)

  Create new LevelMeterShort instance.
- override short[ ] **Process** (short[ ] buf)

Additional Inherited Members

3.56.1 Detailed Description

LevelMeter specialization for short audio.

3.56.2 Constructor & Destructor Documentation

3.56.2.1 LevelMeterShort (int *samplingRate,* int *numChannels*)

Create new LevelMeterShort instance.

Parameters

samplingRate	Sampling rate of the audio signal (in Hz).
numChannels	Number of channels in the audio signal.

3.57 LoadBalancingFrontend Class Reference

Inherits LoadBalancingTransport.

Additional Inherited Members

3.58 LoadBalancingTransport Class Reference

Extends LoadBalancingClient with audio streaming functionality.

Inherits LoadBalancingClient, IVoiceTransport, and IDisposable.

Inherited by LoadBalancingFrontend.

Public Member Functions

- void **LogError** (string fmt, params object[ ] args)
- void **LogWarning** (string fmt, params object[ ] args)
- void **LogInfo** (string fmt, params object[ ] args)
- void **LogDebug** (string fmt, params object[ ] args)
- bool **IsChannelJoined** (int channelId)
- void **SetDebugEchoMode** (LocalVoice v)
- LoadBalancingTransport (ConnectionProtocol connectionProtocol=ConnectionProtocol.Udp)

  Initializes a new LoadBalancingTransport.
- new void Service ()

  This method dispatches all available incoming commands and then sends this client's outgoing commands. Call this method regularly (2 to 20 times a second).
- virtual bool **ChangeAudioGroups** (byte[ ] groupsToRemove, byte[ ] groupsToAdd)
- void **SendVoicesInfo** (IEnumerable< LocalVoice > voices, int channelId, int targetPlayerId)
- void SendDebugEchoVoicesInfo (int channelId)

  Send VoicesInfo events to the local player for all voices that have DebugEcho enabled.
- void **SendVoiceRemove** (LocalVoice voice, int channelId, int targetPlayerId)
- void **SendFrame** (ArraySegment< byte > data, byte evNumber, byte voiceId, int channelId, LocalVoice localVoice)
- string **ChannelIdStr** (int channelId)
- string **PlayerIdStr** (int playerId)
- void Dispose ()

  Releases all resources used by the LoadBalancingTransport instance.

Protected Attributes

- VoiceClient **voiceClient**

Properties

- VoiceClient VoiceClient `[get]`

  The VoiceClient implementation associated with this LoadBalancingTransport.
- byte **GlobalAudioGroup** `[get, set]`
- byte GlobalInterestGroup `[get, set]`

  Set global audio group for this client. This call sets InterestGroup for existing local voices and for created later to given value. Client set as listening to this group only until LoadBalancingPeer.OpChangeGroups() called. This method can be called any time.

3.58.1 Detailed Description

Extends LoadBalancingClient with audio streaming functionality.

Use your normal LoadBalancing workflow to join a Voice room. All standard LoadBalancing features are available.

To work with audio:

- Create outgoing audio streams with Client.CreateLocalVoice.
- Handle new incoming audio streams info with OnRemoteVoiceInfoAction .
- Handle incoming audio streams data with OnAudioFrameAction .
- Handle closing of incoming audio streams with .

3.58.2 Constructor & Destructor Documentation

3.58.2.1 LoadBalancingTransport (ConnectionProtocol *connectionProtocol* = `ConnectionProtocol.Udp`)

Initializes a new LoadBalancingTransport.

Parameters

connection↩ Protocol	Connection protocol (UDP or TCP). ConnectionProtocol

3.58.3 Member Function Documentation

3.58.3.1 void Dispose ( )

Releases all resources used by the LoadBalancingTransport instance.

3.58.3.2 void SendDebugEchoVoicesInfo (int *channelId*)

Send VoicesInfo events to the local player for all voices that have DebugEcho enabled.

This function will call SendVoicesInfo for all local voices of our VoiceClient that have DebugEchoMode set to true, with the given channel ID, and the local Player's ActorNumber as target.

Parameters

channelId	Transport Channel ID

3.58.3.3 new void Service ( )

This method dispatches all available incoming commands and then sends this client's outgoing commands. Call this method regularly (2 to 20 times a second).

3.58.4 Property Documentation

3.58.4.1 byte GlobalInterestGroup `[get],[set]`

Set global audio group for this client. This call sets InterestGroup for existing local voices and for created later to given value. Client set as listening to this group only until LoadBalancingPeer.OpChangeGroups() called. This method can be called any time.

LocalVoice.InterestGroup LoadBalancingPeer.OpChangeGroups(byte[], byte[])

3.58.4.2 VoiceClient VoiceClient `[get]`

The VoiceClient implementation associated with this LoadBalancingTransport.

3.59 LocalVoice Class Reference

Represents outgoing data stream.

Inherits IDisposable.

Inherited by LocalVoiceAudioDummy, and LocalVoiceFramedBase.

Public Member Functions

- virtual IEncoder **CreateDefaultEncoder** (VoiceInfo info)
- void RemoveSelf ()

  Remove this voice from it's VoiceClient (using VoiceClient.RemoveLocalVoice
- virtual void **Dispose** ()

Public Attributes

- const int **DATA_POOL_CAPACITY** = 50

Protected Member Functions

- void **resetNoTransmitCnt** ()

Protected Attributes

- IEncoder **encoder**
- VoiceClient **voiceClient**
- volatile bool **disposed**
- object **disposeLock** = new object()

Properties

- byte **Group** `[get, set]`
- byte InterestGroup `[get, set]`

  If InterestGroup != 0, voice's data is sent only to clients listening to this group (if supported by transport).
- VoiceInfo Info `[get]`

  Returns Info structure assigned on local voice cration.
- bool TransmitEnabled `[get, set]`

  If true, stream data broadcasted.
- bool IsCurrentlyTransmitting `[get, protected set]`

  Returns true if stream broadcasts.
- int FramesSent `[get]`

  Sent frames counter.
- int FramesSentBytes `[get]`

  Sent frames bytes counter.
- bool Reliable `[get, set]`

  Send data reliable.
- bool Encrypt `[get, set]`

  Send data encrypted.
- IServiceable LocalUserServiceable `[get, set]`

  Optional user object attached to LocalVoice. its Service() will be called at each VoiceClient.Service() call.
- bool DebugEchoMode `[get, set]`

  If true, outgoing stream routed back to client via server same way as for remote client's streams. Can be swithed any time. OnRemoteVoiceInfoAction and OnRemoteVoiceRemoveAction are triggered if required. This functionality availability depends on transport.

3.59.1 Detailed Description

Represents outgoing data stream.

3.59.2 Member Function Documentation

3.59.2.1 void RemoveSelf ( )

Remove this voice from it's VoiceClient (using VoiceClient.RemoveLocalVoice

.

3.59.3 Property Documentation

3.59.3.1 bool DebugEchoMode [get], [set]

If true, outgoing stream routed back to client via server same way as for remote client's streams. Can be swithed any time. OnRemoteVoiceInfoAction and OnRemoteVoiceRemoveAction are triggered if required. This functionality availability depends on transport.

3.59.3.2 bool Encrypt [get], [set]

Send data encrypted.

3.59.3.3 int FramesSent [get]

Sent frames counter.

3.59.3.4 int FramesSentBytes [get]

Sent frames bytes counter.

3.59.3.5 VoiceInfo Info [get]

Returns Info structure assigned on local voice cration.

3.59.3.6 byte InterestGroup [get], [set]

If InterestGroup != 0, voice's data is sent only to clients listening to this group (if supported by transport).

3.59.3.7 bool IsCurrentlyTransmitting [get], [protected set]

Returns true if stream broadcasts.

3.59.3.8 IServiceable LocalUserServiceable [get], [set]

Optional user object attached to LocalVoice. its Service() will be called at each VoiceClient.Service() call.

3.59.3.9 bool Reliable [get], [set]

Send data reliable.

3.59.3.10 bool TransmitEnabled [get], [set]

If true, stream data broadcasted.

3.60 LocalVoiceAudio< T > Class Template Reference

Outgoing audio stream.

Inherits LocalVoiceFramed< T >, and ILocalVoiceAudio.

Public Member Functions

- override IEncoder **CreateDefaultEncoder** (VoiceInfo info)
- void VoiceDetectorCalibrate (int durationMs, Action< float > onCalibrated=null)

  Trigger voice detector calibration process.

Static Public Member Functions

- static LocalVoiceAudio< T > Create (VoiceClient voiceClient, byte voiceId, IEncoder encoder, VoiceInfo voiceInfo, IAudioDesc audioSourceDesc, int channelId)

  Create a new LocalVoiceAudio<T> instance.

Protected Member Functions

- void **initBuiltinProcessors** ()

Protected Attributes

- AudioUtil.VoiceDetector< T > **voiceDetector**
- AudioUtil.VoiceDetectorCalibration< T > **voiceDetectorCalibration**
- AudioUtil.LevelMeter< T > **levelMeter**
- int **channels**
- bool **resampleSource**

Properties

- virtual AudioUtil.IVoiceDetector **VoiceDetector** `[get]`
- virtual AudioUtil.ILevelMeter **LevelMeter** `[get]`
- bool VoiceDetectorCalibrating `[get]`

  True if the VoiceDetector is currently calibrating.

Additional Inherited Members

3.60.1 Detailed Description

Outgoing audio stream.

3.60.2 Member Function Documentation

3.60.2.1 static LocalVoiceAudio<T> Create (VoiceClient *voiceClient,* byte *voiceId,* IEncoder *encoder,* VoiceInfo *voiceInfo,* IAudioDesc *audioSourceDesc,* int *channelId*) `[static]`

Create a new LocalVoiceAudio<T> instance.

Parameters

voiceClient	The VoiceClient to use for this outgoing stream.
voiceId	Numeric ID for this voice.
encoder	Encoder to use for this voice.
channelId	Voice transport channel ID to use for this voice.

Returns

The new LocalVoiceAudio<T> instance.

3.60.2.2 void VoiceDetectorCalibrate (int *durationMs*, Action< float > *onCalibrated* = `null`)

Trigger voice detector calibration process.

While calibrating, keep silence. Voice detector sets threshold basing on measured backgroud noise level.

Parameters

durationMs	Duration of calibration in milliseconds.
onCalibrated	Called when calibration is complete. Parameter is new threshold value.

Implements ILocalVoiceAudio.

3.60.3 Property Documentation

3.60.3.1 bool VoiceDetectorCalibrating `[get]`

True if the VoiceDetector is currently calibrating.

3.61 LocalVoiceAudioDummy Class Reference

Dummy LocalVoiceAudio

Inherits LocalVoice, and ILocalVoiceAudio.

Public Member Functions

- void VoiceDetectorCalibrate (int durationMs, Action< float > onCalibrated=null)

  Trigger voice detector calibration process.

Static Public Attributes

- static LocalVoiceAudioDummy Dummy = new LocalVoiceAudioDummy()

  A Dummy LocalVoiceAudio instance.

Properties

- AudioUtil.IVoiceDetector **VoiceDetector** `[get]`
- AudioUtil.ILevelMeter **LevelMeter** `[get]`
- bool **VoiceDetectorCalibrating** `[get]`

Additional Inherited Members

3.61.1 Detailed Description

Dummy LocalVoiceAudio

For testing, this LocalVoiceAudio implementation features a AudioUtil.VoiceDetectorDummy and a AudioUtil.Level↩ MeterDummy

3.61.2 Member Function Documentation

3.61.2.1 void VoiceDetectorCalibrate (int *durationMs*, Action< float > *onCalibrated* = `null`)

Trigger voice detector calibration process.

While calibrating, keep silence. Voice detector sets threshold based on measured backgroud noise level.

Parameters

durationMs	Duration of calibration (in milliseconds).
onCalibrated	Called when calibration is complete. Parameter is new threshold value.

Implements ILocalVoiceAudio.

3.61.3 Member Data Documentation

3.61.3.1 LocalVoiceAudioDummy Dummy = new LocalVoiceAudioDummy() `[static]`

A Dummy LocalVoiceAudio instance.

3.62 LocalVoiceAudioFloat Class Reference

Specialization of LocalVoiceAudio for float audio

Inherits LocalVoiceAudio< float >.

Additional Inherited Members

3.62.1 Detailed Description

Specialization of LocalVoiceAudio for float audio

3.63 LocalVoiceAudioShort Class Reference

Specialization of LocalVoiceAudio for short audio

Inherits LocalVoiceAudio< short >.

Additional Inherited Members

3.63.1 Detailed Description

Specialization of LocalVoiceAudio for short audio

3.64 LocalVoiceFramed< T > Class Template Reference

Typed re-framing LocalVoice

Inherits LocalVoiceFramedBase.

Inherited by LocalVoiceAudio< T >.

Public Member Functions

- void AddPostProcessor (params IProcessor< T >[ ] processors)

  Adds processors after any built-in processors and everything added with AddPreProcessor.
- void AddPreProcessor (params IProcessor< T >[ ] processors)

  Adds processors before built-in processors and everything added with AddPostProcessor.
- void ClearProcessors ()

  Clears all processors in pipeline including built-in resampling. User should add at least resampler processor after call.
- void PushDataAsync (T[ ] buf)

  Asynchronously push data into this stream.
- void PushData (T[ ] buf)

  Synchronously push data into this stream.
- override void Dispose ()

  Releases resources used by the VoiceFramed instance. Buffers used for asynchronous push will be disposed in encoder thread's 'finally'.

Properties

- FactoryPrimitiveArrayPool< T > **BufferFactory** `[get]`
- bool PushDataAsyncReady `[get]`

  Wether this LocalVoiceFramed has capacity for more data buffers to be pushed asynchronously.

Additional Inherited Members

3.64.1 Detailed Description

Typed re-framing LocalVoice

Consumes data in array buffers of arbitrary length. Repacks them in frames of constant length for further processing and encoding.

Parameters

voiceInfo	Outgoing stream parameters. Set applicable fields to read them by encoder and by receiving client when voice created.
channelId	Transport channel specific to transport.
encoder	Encoder producing the stream.

Returns

Outgoing stream handler.

3.64.2 Member Function Documentation

3.64.2.1 void AddPostProcessor (params IProcessor< T >[ ] *processors*)

Adds processors after any built-in processors and everything added with AddPreProcessor.

Parameters

processors	

3.64.2.2 void AddPreProcessor (params IProcessor< T >[ ] *processors*)

Adds processors before built-in processors and everything added with AddPostProcessor.

Parameters

processors	

3.64.2.3 void ClearProcessors ( )

Clears all processors in pipeline including built-in resampling. User should add at least resampler processor after call.

3.64.2.4 override void Dispose ( ) [virtual]

Releases resources used by the VoiceFramed instance. Buffers used for asynchronous push will be disposed in encoder thread's 'finally'.

Reimplemented from LocalVoice.

3.64.2.5 void PushData (T[ ] *buf*)

Synchronously push data into this stream.

3.64.2.6 void PushDataAsync (T[ ] *buf*)

Asynchronously push data into this stream.

3.64.3 Property Documentation

3.64.3.1 bool PushDataAsyncReady [get]

Wether this LocalVoiceFramed has capacity for more data buffers to be pushed asynchronously.

3.65 LocalVoiceFramedBase Class Reference

Typed re-framing LocalVoice

Inherits LocalVoice.

Inherited by LocalVoiceFramed< T >.

Properties

- int FrameSize [get]

  Data flow will be repacked to frames of this size. May differ from input voiceInfo.FrameSize. Processors should resample in this case.

Additional Inherited Members

3.65.1 Detailed Description

Typed re-framing LocalVoice

Base class for typed re-framing LocalVoice implementation (LocalVoiceFramedBase<T>)

3.65.2 Property Documentation

3.65.2.1 int FrameSize [get]

Data flow will be repacked to frames of this size. May differ from input voiceInfo.FrameSize. Processors should resample in this case.

3.66 Logger Class Reference

Inherits ILogger.

Public Member Functions

- void **LogError** (string fmt, params object[ ] args)
- void **LogWarning** (string fmt, params object[ ] args)
- void **LogInfo** (string fmt, params object[ ] args)
- void **LogDebug** (string fmt, params object[ ] args)

3.67 MicAmplifier Class Reference

Inherits VoiceComponent.

Properties

- float **AmplificationFactor** [get, set]
- float **BoostValue** [get, set]

Additional Inherited Members

3.68 MicAmplifierFloat Class Reference

Inherits IProcessor< float >.

Public Member Functions

- **MicAmplifierFloat** (float amplificationFactor, float boostValue)
- float[ ] **Process** (float[ ] buf)
- void **Dispose** ()

Properties

- float **AmplificationFactor** `[get, set]`
- float **BoostValue** `[get, set]`
- float **MaxBefore** `[get]`
- float **MaxAfter** `[get]`
- bool **Disabled** `[get, set]`

3.69 MicAmplifierShort Class Reference

Inherits IProcessor< short >.

Public Member Functions

- **MicAmplifierShort** (short amplificationFactor, short boostValue)
- short[ ] **Process** (short[ ] buf)
- void **Dispose** ()

Properties

- short **AmplificationFactor** `[get, set]`
- short **BoostValue** `[get, set]`
- short **MaxBefore** `[get]`
- short **MaxAfter** `[get]`
- bool **Disabled** `[get, set]`

3.70 MicWrapper Class Reference

Inherits IAudioReader< float >.

Public Member Functions

- **MicWrapper** (string device, int suggestedFrequency, Voice.ILogger logger)
- void **Dispose** ()
- bool **Read** (float[ ] buffer)

Properties

- int **SamplingRate** `[get]`
- int **Channels** `[get]`
- string **Error** `[get]`

3.71 ObjectFactory< TType, TInfo > Interface Template Reference

Uniform interface to ObjectPool<TType, TInfo> and single reusable object.

Inherits IDisposable.

Public Member Functions

- TType **New** ()
- TType **New** (TInfo info)
- void **Free** (TType obj)
- void **Free** (TType obj, TInfo info)

Properties

- TInfo **Info** `[get]`

3.71.1 Detailed Description

Uniform interface to ObjectPool<TType, TInfo> and single reusable object.

Template Parameters

TType	Object type.
TInfo	Type of property used to check 2 objects identity (like integral length of array).

3.72 ObjectPool< TType, TInfo > Class Template Reference

Generic Pool to re-use objects of a certain type (TType) that optionally match a certain property or set of properties (TInfo).

Inherits IDisposable.

Public Member Functions

- ObjectPool (int capacity, string name)

  Create a new ObjectPool instance. Does not call Init().
- ObjectPool (int capacity, string name, TInfo info)

  Create a new ObjectPool instance with the given info structure. Calls Init().
- void Init (TInfo info)

  (Re-)Initializes this ObjectPool.
- TType AcquireOrCreate ()

  Acquire an existing object, or create a new one if none are available.
- TType AcquireOrCreate (TInfo info)

  Acquire an existing object (if info matches), or create a new one from the passed info.
- virtual bool Release (TType obj, TInfo objInfo)

  Returns object to pool.
- virtual bool Release (TType obj)

  Returns object to pool, or destroys it if the pool is full.
- void Dispose ()

  Free resources assoicated with this ObjectPool

Protected Member Functions

- abstract TType **createObject** (TInfo info)
- abstract void **destroyObject** (TType obj)
- abstract bool **infosMatch** (TInfo i0, TInfo i1)

Protected Attributes

- int **capacity**
- TInfo **info**
- int **pos**
- string **name**

Properties

- TInfo Info `[get]`

  The property (info) that objects in this Pool must match.

3.72.1 Detailed Description

Generic Pool to re-use objects of a certain type (TType) that optionally match a certain property or set of properties (TInfo).

Template Parameters

TType	Object type.
TInfo	Type of parameter used to check 2 objects identity (like integral length of array).

3.72.2 Constructor & Destructor Documentation

3.72.2.1 ObjectPool (int *capacity*, string *name*)

Create a new ObjectPool instance. Does not call Init().

Parameters

capacity	Capacity (size) of the object pool.
name	Name of the object pool.

3.72.2.2 ObjectPool (int *capacity*, string *name*, TInfo *info*)

Create a new ObjectPool instance with the given info structure. Calls Init().

Parameters

capacity	Capacity (size) of the object pool.
name	Name of the object pool.
info	Info about this Pool's objects.

3.72.3 Member Function Documentation

3.72.3.1 TType AcquireOrCreate ( )

Acquire an existing object, or create a new one if none are available.

If it fails to get one from the pool, this will create from the info given in this pool's constructor.

3.72.3.2 TType AcquireOrCreate (TInfo *info*)

Acquire an existing object (if info matches), or create a new one from the passed info.

Parameters

info	Info structure to match, or create a new object with.

3.72.3.3 void Dispose ( )

Free resources assoicated with this ObjectPool

3.72.3.4 void Init (TInfo *info*)

(Re-)Initializes this ObjectPool.

If there are objects available in this Pool, they will be destroyed. Allocates (Capacity) new Objects.

Parameters

info	Info about this Pool's objects.

3.72.3.5 virtual bool Release (TType *obj,* TInfo *objInfo*) [virtual]

Returns object to pool.

Parameters

obj	The object to return to the pool.
objInfo	The info structure about obj.

obj is returned to the pool only if objInfo matches this pool's info. Else, it is destroyed.

3.72.3.6 virtual bool Release (TType *obj*) [virtual]

Returns object to pool, or destroys it if the pool is full.

Parameters

obj	The object to return to the pool.

3.72.4 Property Documentation

3.72.4.1 TInfo Info [get]

The property (info) that objects in this Pool must match.

3.73 OpusCodec Class Reference

Classes

- class Decoder
- class DecoderFactory
- class DecoderFloat
- class DecoderShort
- class Encoder
- class EncoderFloat
- class EncoderShort

- class Factory
- class Util

Public Types

- enum **FrameDuration**

3.74 OpusDecoder Class Reference

Inherits IDisposable.

Public Member Functions

- **OpusDecoder** (SamplingRate outputSamplingRateHz, Channels numChannels)
- float[ ] **DecodePacketFloat** (byte[ ] packetData)
- short[ ] **DecodePacketShort** (byte[ ] packetData)
- void **Dispose** ()

Properties

- string **Version** `[get]`
- Bandwidth **PreviousPacketBandwidth** `[get]`

3.75 OpusEncoder Class Reference

Inherits IDisposable.

Public Member Functions

- **OpusEncoder** (SamplingRate inputSamplingRateHz, Channels numChannels, int bitrate, OpusApplication↩ Type applicationType, Delay encoderDelay)
- ArraySegment< byte > **Encode** (float[ ] pcmSamples)
- ArraySegment< byte > **Encode** (short[ ] pcmSamples)
- void **Dispose** ()

Public Attributes

- const int **BitrateMax** = -1

Properties

- SamplingRate **InputSamplingRate** `[get]`
- Channels **InputChannels** `[get]`
- string **Version** `[get]`
- Delay EncoderDelay `[get, set]`

  Using a duration of less than 10 ms will prevent the encoder from using the LPC or hybrid modes.
- int **FrameSizePerChannel** `[get]`
- int **Bitrate** `[get, set]`
- Bandwidth **MaxBandwidth** `[get, set]`

- Complexity **Complexity** `[get, set]`
- int **ExpectedPacketLossPercentage** `[get, set]`
- SignalHint **SignalHint** `[get, set]`
- ForceChannels **ForceChannels** `[get, set]`
- bool **UseInbandFEC** `[get, set]`
- bool **UseUnconstrainedVBR** `[get, set]`
- bool **DtxEnabled** `[get, set]`

3.75.1 Property Documentation

3.75.1.1 Delay EncoderDelay `[get],[set]`

Using a duration of less than 10 ms will prevent the encoder from using the LPC or hybrid modes.

3.76 OpusException Class Reference

Inherits Exception.

Public Member Functions

- **OpusException** (OpusStatusCode statusCode, string message)

Properties

- OpusStatusCode **StatusCode** `[get]`

3.77 WebRTCAudioLib.Param Struct Reference

Public Attributes

- const int **REVERSE_STREAM_DELAY_MS** = 1
- const int **AEC** = 10
- const int **AEC_SUPPRESSION_LEVEL** = 11
- const int **AECM** = 20
- const int **AECM_ROUTING_MODE** = 21
- const int **AECM_COMFORT_NOISE** = 22
- const int **HIGH_PASS_FILTER** = 31
- const int **NS** = 41
- const int **NS_LEVEL** = 42
- const int **AGC** = 51
- const int **AGC_MODE** = 52
- const int **AGC_COMPRESSION_GAIN** = 56
- const int **AGC_LIMITER** = 57
- const int **VAD** = 61
- const int **VAD_FRAME_SIZE_MS** = 62
- const int **VAD_LIKEHOOD** = 63

3.78 PhotonVoiceCreatedParams Class Reference

Inherited by Recorder.PhotonVoiceCreatedParams.

Properties

- Voice.LocalVoice **Voice** `[get, set]`
- Voice.IAudioDesc **AudioDesc** `[get, set]`

3.79 Recorder.PhotonVoiceCreatedParams Class Reference

Inherits PhotonVoiceCreatedParams.

Additional Inherited Members

3.80 PhotonVoiceLagSimulationGui Class Reference

Inherits MonoBehaviour.

Public Member Functions

- void **OnEnable** ()

3.81 PhotonVoiceNetwork Class Reference

This class can be used to automatically sync client states between PUN and Voice. It also sets a custom PUN Speaker factory to find the Speaker component for a character's voice. For this to work attach a PhotonVoiceView next to the PhotonView of your player's prefab.

Inherits VoiceConnection.

Public Member Functions

- bool ConnectAndJoinRoom ()

  Connect voice client to Photon servers and join a Voice room
- void Disconnect ()

  Disconnect voice client from all Photon servers

Public Attributes

- const string VoiceRoomNameSuffix = "_voice_"

  Suffix for voice room names appended to PUN room names.
- bool AutoConnectAndJoin = true

  Auto connect voice client and join a voice room when PUN client is joined to a PUN room
- bool AutoLeaveAndDisconnect = true

  Auto disconnect voice client when PUN client is not joined to a PUN room
- bool AutoCreateSpeakerIfNotFound = true

  Auto instantiate a GameObject and attach a Speaker component to link to a remote audio stream if no candidate could be found

Protected Member Functions

- override void **Awake** ()
- override void **OnDisable** ()
- override void **OnDestroy** ()
- override void **OnVoiceStateChanged** (ClientState fromState, ClientState toState)
- override Speaker **SimpleSpeakerFactory** (int playerId, byte voiceId, object userData)

Properties

- static PhotonVoiceNetwork Instance `[get, set]`

  Singleton instance for PhotonVoiceNetwork

Additional Inherited Members

3.81.1 Detailed Description

This class can be used to automatically sync client states between PUN and Voice. It also sets a custom PUN Speaker factory to find the Speaker component for a character's voice. For this to work attach a PhotonVoiceView next to the PhotonView of your player's prefab.

3.81.2 Member Function Documentation

3.81.2.1 bool ConnectAndJoinRoom ( )

Connect voice client to Photon servers and join a Voice room

Returns

If true, connection command send from client

3.81.2.2 void Disconnect ( )

Disconnect voice client from all Photon servers

3.81.3 Member Data Documentation

3.81.3.1 bool AutoConnectAndJoin = true

Auto connect voice client and join a voice room when PUN client is joined to a PUN room

3.81.3.2 bool AutoCreateSpeakerIfNotFound = true

Auto instantiate a GameObject and attach a Speaker component to link to a remote audio stream if no candidate could be found

3.81.3.3 bool AutoLeaveAndDisconnect = true

Auto disconnect voice client when PUN client is not joined to a PUN room

3.81.3.4 const string VoiceRoomNameSuffix = "_voice_"

Suffix for voice room names appended to PUN room names.

3.81.4 Property Documentation

3.81.4.1 PhotonVoiceNetwork Instance [static],[get],[set]

Singleton instance for PhotonVoiceNetwork

3.82 PhotonVoiceStatsGui Class Reference

Basic GUI to show traffic and health statistics of the connection to Photon, toggled by shift+tab.

Inherits MonoBehaviour.

3.82.1 Detailed Description

Basic GUI to show traffic and health statistics of the connection to Photon, toggled by shift+tab.

The shown health values can help identify problems with connection losses or performance. Example: If the time delta between two consecutive SendOutgoingCommands calls is a second or more, chances rise for a disconnect being caused by this (because acknowledgments to the server need to be sent in due time).

3.83 PhotonVoiceView Class Reference

Component that should be attached to a networked PUN prefab that has PhotonView. It will bind remote Recorder with local Speaker of the same networked prefab. This component makes automatic voice stream routing easy for players' characters/avatars.

Inherits VoiceComponent.

Public Attributes

- bool AutoCreateRecorderIfNotFound

  If true, a Recorder component will be added to the same GameObject if not found already.
- bool UsePrimaryRecorder

  If true, PhotonVoiceNetwork.PrimaryRecorder will be used by this PhotonVoiceView
- bool SetupDebugSpeaker

  If true, a Speaker component will be setup to be used for the DebugEcho mode

Protected Member Functions

- override void **Awake** ()

Properties

- Recorder RecorderInUse [get, set]

  The Recorder component currently used by this PhotonVoiceView
- Speaker SpeakerInUse [get, set]

  The Speaker component currently used by this PhotonVoiceView

- bool IsSetup `[get, protected set]`

  If true, this PhotonVoiceView is setup and ready to be used
- bool IsSpeaker `[get, protected set]`

  If true, this PhotonVoiceView has a Speaker setup for playback of received audio frames from remote audio source
- bool IsSpeaking `[get]`

  If true, this PhotonVoiceView has a Speaker that is currently playing received audio frames from remote audio source
- bool IsRecorder `[get, protected set]`

  If true, this PhotonVoiceView has a Recorder setup for transmission of audio stream from local audio source
- bool IsRecording `[get]`

  If true, this PhotonVoiceView has a Recorder that is currently transmitting audio stream from local audio source

Additional Inherited Members

3.83.1 Detailed Description

Component that should be attached to a networked PUN prefab that has PhotonView. It will bind remote Recorder with local Speaker of the same networked prefab. This component makes automatic voice stream routing easy for players' characters/avatars.

3.83.2 Member Data Documentation

3.83.2.1 bool AutoCreateRecorderIfNotFound

If true, a Recorder component will be added to the same GameObject if not found already.

3.83.2.2 bool SetupDebugSpeaker

If true, a Speaker component will be setup to be used for the DebugEcho mode

3.83.2.3 bool UsePrimaryRecorder

If true, PhotonVoiceNetwork.PrimaryRecorder will be used by this PhotonVoiceView

3.83.3 Property Documentation

3.83.3.1 bool IsRecorder `[get], [protected set]`

If true, this PhotonVoiceView has a Recorder setup for transmission of audio stream from local audio source

3.83.3.2 bool IsRecording `[get]`

If true, this PhotonVoiceView has a Recorder that is currently transmitting audio stream from local audio source

3.83.3.3 bool IsSetup `[get], [protected set]`

If true, this PhotonVoiceView is setup and ready to be used

3.83.3.4 bool IsSpeaker `[get], [protected set]`

If true, this PhotonVoiceView has a Speaker setup for playback of received audio frames from remote audio source

3.83.3.5 bool IsSpeaking [get]

If true, this PhotonVoiceView has a Speaker that is currently playing received audio frames from remote audio source

3.83.3.6 Recorder RecorderInUse [get], [set]

The Recorder component currently used by this PhotonVoiceView

3.83.3.7 Speaker SpeakerInUse [get], [set]

The Speaker component currently used by this PhotonVoiceView

3.84 PrimitiveArrayPool< T > Class Template Reference

Pool of Arrays with components of type T, with ObjectPool info being the array's size.

Inherits ObjectPool< T[], int >.

Public Member Functions

- **PrimitiveArrayPool** (int capacity, string name)
- **PrimitiveArrayPool** (int capacity, string name, int info)

Protected Member Functions

- override T[] **createObject** (int info)
- override void **destroyObject** (T[] obj)
- override bool **infosMatch** (int i0, int i1)

Additional Inherited Members

3.84.1 Detailed Description

Pool of Arrays with components of type T, with ObjectPool info being the array's size.

Template Parameters

T	Array element type.

3.85 Recorder Class Reference

Component representing outgoing audio stream in scene.

Inherits VoiceComponent.

Classes

- class PhotonVoiceCreatedParams

Public Types

- enum **InputSourceType**
- enum **MicType**
- enum **SampleTypeConv**

Public Member Functions

- void Init (VoiceClient voiceClient, object customObj=null)

  Initializes the Recorder component to be able to transmit audio.
- void **ReInit** ()
- void RestartRecording ()

  Restarts recording if something has changed that requires this.
- void VoiceDetectorCalibrate (int durationMs, Action< float > detectionEndedCallback=null)

  Trigger voice detector calibration process. While calibrating, keep silence. Voice detector sets threshold basing on measured backgroud noise level.
- void StartRecording ()

  Starts recording.
- void StopRecording ()

  Stops recording.

Protected Member Functions

- virtual void **SendPhotonVoiceCreatedMessage** ()

Properties

- static AudioInEnumerator PhotonMicrophoneEnumerator `[get]`

  Enumerator for the available microphone devices gathered by the Photon plugin.
- bool IsInitialized `[get]`

  If true, this Recorder has been initialized and is ready to transmit to remote clients. Otherwise call Init.
- bool **RequiresInit** `[get]`
- bool RequiresRestart `[get, protected set]`

  Returns true if something has changed in the Recorder while recording that won't take effect unless recording is restarted using RestartRecording.
- bool TransmitEnabled `[get, set]`

  If true, audio transmission is enabled.
- bool Encrypt `[get, set]`

  If true, voice stream is sent encrypted.
- bool DebugEchoMode `[get, set]`

  If true, outgoing stream routed back to client via server same way as for remote client's streams.
- bool ReliableMode `[get, set]`

  If true, stream data sent in reliable mode.
- bool VoiceDetection `[get, set]`

  If true, voice detection enabled.
- float VoiceDetectionThreshold `[get, set]`

  Voice detection threshold (0..1, where 1 is full amplitude).
- int VoiceDetectionDelayMs `[get, set]`

  Keep detected state during this time after signal level dropped below threshold. Default is 500ms
- object UserData `[get, set]`

  Custom user object to be sent in the voice stream info event.

- Func< IAudioDesc > InputFactory [get, set]

  Set the method returning new Voice.IAudioDesc instance to be assigned to a new voice created with Source set to Factory
- AudioUtil.IVoiceDetector VoiceDetector [get]

  Returns voice activity detector for recorder's audio stream.
- string UnityMicrophoneDevice [get, set]

  Set or get Unity microphone device used for streaming.
- int PhotonMicrophoneDeviceId [get, set]

  Set or get photon microphone device used for streaming.
- byte AudioGroup [get, set]

  Target interest group that will receive transmitted audio.
- byte InterestGroup [get, set]

  Target interest group that will receive transmitted audio.
- bool IsCurrentlyTransmitting [get]

  Returns true if audio stream broadcasts.
- AudioUtil.ILevelMeter LevelMeter [get]

  Level meter utility.
- bool VoiceDetectorCalibrating [get]

  If true, voice detector calibration is in progress.
- ILocalVoiceAudio **voiceAudio** [get]
- InputSourceType SourceType [get, set]

  Audio data source.
- MicType MicrophoneType [get, set]

  Which microphone API to use when the Source is set to Microphone.
- SampleTypeConv TypeConvert [get, set]

  Force creation of 'short' pipeline and convert audio data to short for 'float' audio sources.
- AudioClip AudioClip [get, set]

  Source audio clip.
- bool LoopAudioClip [get, set]

  Loop playback for audio clip sources.
- POpusCodec.Enums.SamplingRate SamplingRate [get, set]

  Outgoing audio stream sampling rate.
- OpusCodec.FrameDuration FrameDuration [get, set]

  Outgoing audio stream encoder delay.
- int Bitrate [get, set]

  Outgoing audio stream bitrate.
- bool IsRecording [get, set]

  Gets or sets whether this Recorder is actively recording audio to be transmitted.
- bool **ReactOnSystemChanges** [get, set]
- bool AutoStart [get, set]

  If true, automatically start recording when initialized.

Additional Inherited Members

3.85.1 Detailed Description

Component representing outgoing audio stream in scene.

3.85.2 Member Function Documentation

3.85.2.1 void Init (VoiceClient *voiceClient*, object *customObj* = null)

Initializes the Recorder component to be able to transmit audio.

Parameters

voiceClient	The VoiceClient to be used with this Recorder.
customObj	Optional user data object to be transmitted with the voice stream info

3.85.2.2 void RestartRecording ( )

Restarts recording if something has changed that requires this.

3.85.2.3 void StartRecording ( )

Starts recording.

3.85.2.4 void StopRecording ( )

Stops recording.

3.85.2.5 void VoiceDetectorCalibrate (int *durationMs*, Action< float > *detectionEndedCallback* = `null`)

Trigger voice detector calibration process. While calibrating, keep silence. Voice detector sets threshold basing on measured backgroud noise level.

Parameters

durationMs	Duration of calibration in milliseconds.

3.85.3 Property Documentation

3.85.3.1 AudioClip AudioClip [get], [set]

Source audio clip.

3.85.3.2 byte AudioGroup [get], [set]

Target interest group that will receive transmitted audio.

If AudioGroup != 0, recorder's audio data is sent only to clients listening to this group.

3.85.3.3 bool AutoStart [get], [set]

If true, automatically start recording when initialized.

3.85.3.4 int Bitrate [get], [set]

Outgoing audio stream bitrate.

3.85.3.5 bool DebugEchoMode [get], [set]

If true, outgoing stream routed back to client via server same way as for remote client's streams.

3.85.3.6 bool Encrypt [get], [set]

If true, voice stream is sent encrypted.

3.85.3.7 OpusCodec.FrameDuration FrameDuration [get], [set]

Outgoing audio stream encoder delay.

3.85.3.8 Func<IAudioDesc> InputFactory [get], [set]

Set the method returning new Voice.IAudioDesc instance to be assigned to a new voice created with Source set to Factory

3.85.3.9 byte InterestGroup [get], [set]

Target interest group that will receive transmitted audio.

If InterestGroup != 0, recorder's audio data is sent only to clients listening to this group.

3.85.3.10 bool IsCurrentlyTransmitting [get]

Returns true if audio stream broadcasts.

3.85.3.11 bool IsInitialized [get]

If true, this Recorder has been initialized and is ready to transmit to remote clients. Otherwise call Init.

3.85.3.12 bool IsRecording [get], [set]

Gets or sets whether this Recorder is actively recording audio to be transmitted.

3.85.3.13 AudioUtil.ILevelMeter LevelMeter [get]

Level meter utility.

3.85.3.14 bool LoopAudioClip [get], [set]

Loop playback for audio clip sources.

3.85.3.15 MicType MicrophoneType [get], [set]

Which microphone API to use when the Source is set to Microphone.

3.85.3.16 int PhotonMicrophoneDeviceId [get], [set]

Set or get photon microphone device used for streaming.

3.85.3.17 AudioInEnumerator PhotonMicrophoneEnumerator [static], [get]

Enumerator for the available microphone devices gathered by the Photon plugin.

3.85.3.18 bool ReliableMode [get], [set]

If true, stream data sent in reliable mode.

3.85.3.19 bool RequiresRestart [get], [protected set]

Returns true if something has changed in the Recorder while recording that won't take effect unless recording is restarted using RestartRecording.

Think of this as a "isDirty" flag.

3.85.3.20 POpusCodec.Enums.SamplingRate SamplingRate [get], [set]

Outgoing audio stream sampling rate.

3.85.3.21 InputSourceType SourceType [get], [set]

Audio data source.

3.85.3.22 bool TransmitEnabled [get], [set]

If true, audio transmission is enabled.

3.85.3.23 SampleTypeConv TypeConvert [get], [set]

Force creation of 'short' pipeline and convert audio data to short for 'float' audio sources.

3.85.3.24 string UnityMicrophoneDevice [get], [set]

Set or get Unity microphone device used for streaming.

3.85.3.25 object UserData [get], [set]

Custom user object to be sent in the voice stream info event.

3.85.3.26 bool VoiceDetection [get], [set]

If true, voice detection enabled.

3.85.3.27 int VoiceDetectionDelayMs [get], [set]

Keep detected state during this time after signal level dropped below threshold. Default is 500ms

3.85.3.28 float VoiceDetectionThreshold [get], [set]

Voice detection threshold (0..1, where 1 is full amplitude).

3.85.3.29 AudioUtil.IVoiceDetector VoiceDetector [get]

Returns voice activity detector for recorder's audio stream.

3.85.3.30 bool VoiceDetectorCalibrating [get]

If true, voice detector calibration is in progress.

3.86 RemoteVoiceInfo Class Reference

Information about a remote voice (incoming stream).

Properties

- VoiceInfo Info [get]

  Remote voice info.
- int ChannelId [get]

  ID of channel used for transmission.
- int PlayerId [get]

  Player ID of voice owner.
- byte VoiceId [get]

  Voice ID (unique in the room).

3.86.1 Detailed Description

Information about a remote voice (incoming stream).

3.86.2 Property Documentation

3.86.2.1 int ChannelId [get]

ID of channel used for transmission.

3.86.2.2 VoiceInfo Info [get]

Remote voice info.

3.86.2.3 int PlayerId [get]

Player ID of voice owner.

3.86.2.4 byte VoiceId [get]

Voice ID (unique in the room).

3.87 RemoteVoiceLink Class Reference

Public Member Functions

- **RemoteVoiceLink** (VoiceInfo info, int playerId, int voiceId, int channelId, ref RemoteVoiceOptions options)

Properties

- VoiceInfo **Info** `[get]`
- int **PlayerId** `[get]`
- int **VoiceId** `[get]`
- int **ChannelId** `[get]`

Events

- Action< float[ ]> **FloatFrameDecoded**
- Action **RemoteVoiceRemoved**

3.88 RemoteVoiceOptions Struct Reference

Event Actions and other options for a remote voice (incoming stream).

Public Member Functions

- void SetOutput (Action< float[ ]> output)

  Register a method to be called when new data frame received..
- void **SetOutput** (Action< short[ ]> output)
- void **SetOutput** (Action< ImageOutputBuf > output)

Properties

- Action OnRemoteVoiceRemoveAction `[get, set]`

  Register a method to be called when the remote voice is removed.
- IDecoder Decoder `[get, set]`

  Remote voice data decoder. Use to set decoder options or override it with user decoder.
- ImageFormat **OutputImageFormat** `[get, set]`
- Flip **OutputImageFlip** `[get, set]`

3.88.1 Detailed Description

Event Actions and other options for a remote voice (incoming stream).

3.88.2 Member Function Documentation

3.88.2.1 void SetOutput (Action< float[ ]> *output*)

Register a method to be called when new data frame received..

3.88.3 Property Documentation

3.88.3.1 IDecoder Decoder `[get],[set]`

Remote voice data decoder. Use to set decoder options or override it with user decoder.

3.88.3.2 Action OnRemoteVoiceRemoveAction [get],[set]

Register a method to be called when the remote voice is removed.

3.89 AudioUtil.Resampler< T > Class Template Reference

Sample-rate conversion Audio Processor.

Inherits IProcessor< T >.

Public Member Functions

- Resampler (int dstSize, int channels)
  Create a new Resampler instance.
- T[ ] Process (T[ ] buf)
  Process a frame of audio data.
- void **Dispose** ()

Protected Attributes

- T[ ] **frameResampled**

3.89.1 Detailed Description

Sample-rate conversion Audio Processor.

This processor converts the sample-rate of the source stream. Internally, it uses AudioUtil.Resample.

3.89.2 Constructor & Destructor Documentation

3.89.2.1 Resampler (int *dstSize*, int *channels*)

Create a new Resampler instance.

Parameters

dstSize	Frame size of a destination frame. Determins output rate.
channels	Number of audio channels expected in both in- and output.

3.89.3 Member Function Documentation

3.89.3.1 T [ ] Process (T[ ] *buf*)

Process a frame of audio data.

Parameters

buf	Buffer containing input audio data

Returns

Buffer containing output audio data

Implements IProcessor< T >.

3.90 Speaker Class Reference

Component representing remote audio stream in local scene.

Inherits VoiceComponent.

Public Attributes

- int **PlayDelayMs** = 200

Protected Member Functions

- override void **Awake** ()

Properties

- bool IsPlaying `[get]`

  Is the speaker playing right now.
- int Lag `[get]`

  Smoothed difference between (jittering) stream and (clock-driven) audioOutput.
- Action< Speaker > OnRemoteVoiceRemoveAction `[get, set]`

  Register a method to be called when remote voice removed.
- Realtime.Player Actor `[get, set]`

  Per room, the connected users/players are represented with a Realtime.Player, also known as Actor.
- bool IsLinked `[get]`

  Whether or not this Speaker has been linked to a remote voice stream.

Additional Inherited Members

3.90.1 Detailed Description

Component representing remote audio stream in local scene.

3.90.2 Property Documentation

3.90.2.1 Realtime.Player Actor `[get]`, `[set]`

Per room, the connected users/players are represented with a Realtime.Player, also known as Actor.

Photon Voice calls this Actor, to avoid a name-clash with the Player class in Voice.

3.90.2.2 bool IsLinked `[get]`

Whether or not this Speaker has been linked to a remote voice stream.

3.90.2.3 bool IsPlaying `[get]`

Is the speaker playing right now.

3.90.2.4 int Lag [get]

Smoothed difference between (jittering) stream and (clock-driven) audioOutput.

3.90.2.5 Action<Speaker> OnRemoteVoiceRemoveAction [get], [set]

Register a method to be called when remote voice removed.

3.91 TestTone Class Reference

Inherits MonoBehaviour.

3.92 AudioUtil.ToneAudioPusher< T > Class Template Reference

IAudioPusher that provides a constant tone signal.

Inherits IAudioPusher< T >.

Public Member Functions

- ToneAudioPusher (int frequency=440, int bufSizeMs=100, int samplingRate=441000, int channels=2)
  Create a new ToneAudioReader instance
- void SetCallback (Action< T[]> callback, ObjectFactory< T[], int > bufferFactory)
  Set the callback function used for pushing data
- void **Dispose** ()

Properties

- int **Channels** [get]
- int **SamplingRate** [get]
- string **Error** [get]

3.92.1 Detailed Description

IAudioPusher that provides a constant tone signal.

3.92.2 Constructor & Destructor Documentation

3.92.2.1 ToneAudioPusher (int *frequency* = 440, int *bufSizeMs* = 100, int *samplingRate* = 441000, int *channels* = 2)

Create a new ToneAudioReader instance

Parameters

frequency	Frequency of the generated tone (in Hz).
bufSizeMs	Size of buffers to push (in milliseconds).
samplingRate	Sampling rate of the audio signal (in Hz).

channels	Number of channels in the audio signal.

3.92.3 Member Function Documentation

3.92.3.1 void SetCallback (Action< T[ ]> *callback,* ObjectFactory< T[ ], int > *bufferFactory*)

Set the callback function used for pushing data

Parameters

callback	Callback function to use
localVoice	Outgoing audio stream, for context

Implements IAudioPusher< T >.

3.93 ToneAudioReader Class Reference

Inherits IAudioReader< float >.

Public Member Functions

- void **Dispose** ()
- bool **Read** (float[ ] buf)

Properties

- int **Channels** `[get]`
- int **SamplingRate** `[get]`
- string **Error** `[get]`

3.94 AudioUtil.ToneAudioReader< T > Class Template Reference

IAudioReader that provides a constant tone signal.

Inherits IAudioReader< T >.

Public Member Functions

- ToneAudioReader (Func< double > clockSec=null, double frequency=440, int samplingRate=441000, int channels=2)

  Create a new ToneAudioReader instance
- void **Dispose** ()
- bool Read (T[ ] buf)

  Fill full given frame buffer with source uncompressed data or return false if not enough such data.

Properties

- int Channels `[get]`

  Number of channels in the audio signal.
- int SamplingRate `[get]`

Sampling rate of the audio signal (in Hz).

- string Error [get]

  If not null, audio object is in invalid state.

3.94.1 Detailed Description

IAudioReader that provides a constant tone signal.

See also MicWrapper and AudioClipWrapper Because of current resampling algorithm, the tone is distorted if SamplingRate does not equal encoder sampling rate.

3.94.2 Constructor & Destructor Documentation

3.94.2.1 ToneAudioReader (Func< double > *clockSec* = null, double *frequency* = 440, int *samplingRate* = 441000, int *channels* = 2)

Create a new ToneAudioReader instance

Parameters

clockSec	Function to get current time in seconds. In Unity, pass in '() => AudioSettings.dspTime' for better results.
frequency	Frequency of the generated tone (in Hz).
samplingRate	Sampling rate of the audio signal (in Hz).
channels	Number of channels in the audio signal.

3.94.3 Member Function Documentation

3.94.3.1 bool Read (T[ ] *buffer*)

Fill full given frame buffer with source uncompressed data or return false if not enough such data.

Parameters

buffer	Buffer to fill.

Returns

True if buffer was filled successfully, false otherwise.

Implements IDataReader< T >.

3.94.4 Property Documentation

3.94.4.1 int Channels [get]

Number of channels in the audio signal.

3.94.4.2 string Error [get]

If not null, audio object is in invalid state.

3.94.4.3 int SamplingRate [get]

Sampling rate of the audio signal (in Hz).

3.95 UnityAndroidAudioInAEC Class Reference

Inherits IAudioPusher< short >.

Public Member Functions

- **UnityAndroidAudioInAEC** (Voice.ILogger logger)
- void **SetCallback** (Action< short[ ]> callback, ObjectFactory< short[ ], int > bufferFactory)
- void **Dispose** ()

Properties

- int **Channels** `[get]`
- int **SamplingRate** `[get]`
- string **Error** `[get]`

3.96 UnityAudioOut Class Reference

Inherits ISyncAudioOut< float >.

Public Member Functions

- **UnityAudioOut** (AudioSource audioSource)
- void **Start** (int frequency, int channels, int frameSamples, int playDelayMs)
- void **Service** ()
- void **Push** (float[ ] frame)
- void **Stop** ()
- void **Pause** ()
- void **UnPause** ()

Public Attributes

- const int **FRAME_POOL_CAPACITY** = 50

Properties

- int **Lag** `[get]`
- int **PlaySamplePos** `[get, set]`
- bool **IsPlaying** `[get]`

3.97 UnsupportedCodecException Class Reference

Exception thrown if an unsupported codec is encountered.

Inherits Exception.

Public Member Functions

- UnsupportedCodecException (string info, Codec codec, ILogger logger)

  Create a new UnsupportedCodecException.

3.97.1 Detailed Description

Exception thrown if an unsupported codec is encountered.

PhotonVoice currently only supports one Codec, Codec.AudioOpus.

3.97.2 Constructor & Destructor Documentation

3.97.2.1 UnsupportedCodecException (string *info*, Codec *codec*, ILogger *logger*)

Create a new UnsupportedCodecException.

Parameters

info	The info prepending standard message.
codec	The codec actually encountered.
logger	Loogger.

3.98 UnsupportedSampleTypeException Class Reference

Exception thrown if an unsupported audio sample type is encountered.

Inherits Exception.

Public Member Functions

- UnsupportedSampleTypeException (Type t)
  Create a new UnsupportedSampleTypeException.

3.98.1 Detailed Description

Exception thrown if an unsupported audio sample type is encountered.

PhotonVoice generally supports 32-bit floating point ("float") or 16-bit signed integer ("short") audio, but it usually won't be converted automatically due to the high CPU overhead (and potential loss of precision) involved.

3.98.2 Constructor & Destructor Documentation

3.98.2.1 UnsupportedSampleTypeException (Type *t*)

Create a new UnsupportedSampleTypeException.

Parameters

t	The sample type actually encountered.

3.99 OpusCodec.Util Class Reference

3.100 VoiceClient Class Reference

Voice client interact with other clients on network via IVoiceTransport.

Inherits IDisposable.

Public Member Functions

- delegate void RemoteVoiceInfoDelegate (int channelId, int playerId, byte voiceId, VoiceInfo voiceInfo, ref RemoteVoiceOptions options)

  Remote voice info event delegate.
- IEnumerable< LocalVoice > LocalVoicesInChannel (int channelId)

  Iterates through copy of all local voices list of given channel.
- void Service ()

  This method dispatches all available incoming commands and then sends this client's outgoing commands. Call this method regularly (2..20 times a second).
- LocalVoice CreateLocalVoice (VoiceInfo voiceInfo, int channelId=0, IEncoder encoder=null)

  Creates basic outgoing stream w/o data processing support. Provided encoder should generate output data stream.
- LocalVoiceFramed< T > CreateLocalVoiceFramed< T > (VoiceInfo voiceInfo, int frameSize, int channelId=0, IEncoder encoder=null)

  Creates outgoing stream consuming sequence of values passed in array buffers of arbitrary length which repacked in frames of constant length for further processing and encoding.
- LocalVoiceAudio< T > CreateLocalVoiceAudio< T > (VoiceInfo voiceInfo, IAudioDesc audioSourceDesc, int channelId=0, IEncoder encoder=null)

  Creates outgoing audio stream. Adds audio specific features (e.g. resampling, level meter) to processing pipeline and to returning stream handler.
- LocalVoice CreateLocalVoiceAudioFromSource (VoiceInfo voiceInfo, IAudioDesc source, bool forceShort=false, int channelId=0, IEncoder encoder=null)

  Creates outgoing audio stream of type automatically assigned and adds procedures (callback or serviceable) for consuming given audio source data. Adds audio specific features (e.g. resampling, level meter) to processing pipeline and to returning stream handler.
- void RemoveLocalVoice (LocalVoice voice)

  Removes local voice (outgoing data stream).
  Parameters

voice	*Handler of outgoing stream to be removed.*
- void **Dispose** ()

Properties

- int FramesLost `[get, set]`

  Lost frames counter.
- int FramesReceived `[get]`

  Received frames counter.
- int FramesSent `[get]`

  Sent frames counter.
- int FramesSentBytes `[get]`

  Sent frames bytes counter.
- int RoundTripTime `[get]`

  Average time required voice packet to return to sender.
- int RoundTripTimeVariance `[get]`

  Average round trip time variation.
- bool SuppressInfoDuplicateWarning `[get, set]`

  Do not log warning when duplicate info received.
- RemoteVoiceInfoDelegate OnRemoteVoiceInfoAction `[get, set]`

  Register a method to be called when remote voice info arrived (after join or new new remote voice creation). Metod parameters: (int channelId, int playerId, byte voiceId, VoiceInfo voiceInfo, ref RemoteVoiceOptions options);
- int DebugLostPercent `[get, set]`

  Lost frames simulation ratio.

- IEnumerable< LocalVoice > LocalVoices [get]

  Iterates through copy of all local voices list.
- IEnumerable< RemoteVoiceInfo > RemoteVoiceInfos [get]

  Iterates through all remote voices infos.

3.100.1 Detailed Description

Voice client interact with other clients on network via IVoiceTransport.

3.100.2 Member Function Documentation

3.100.2.1 LocalVoice CreateLocalVoice (VoiceInfo *voiceInfo*, int *channelId* = 0, IEncoder *encoder* = null)

Creates basic outgoing stream w/o data processing support. Provided encoder should generate output data stream.

Parameters

voiceInfo	Outgoing stream parameters. Set applicable fields to read them by encoder and by receiving client when voice created.
channelId	Transport channel specific to transport.
encoder	Encoder producing the stream.

Returns

Outgoing stream handler.

3.100.2.2 LocalVoiceAudio<T> CreateLocalVoiceAudio< T > (VoiceInfo *voiceInfo*, IAudioDesc *audioSourceDesc*, int *channelId* = 0, IEncoder *encoder* = null)

Creates outgoing audio stream. Adds audio specific features (e.g. resampling, level meter) to processing pipeline and to returning stream handler.

Template Parameters

T	Element type of audio array buffers.

Parameters

voiceInfo	Outgoing audio stream parameters. Set applicable fields to read them by encoder and by receiving client when voice created.
channelId	Transport channel specific to transport.
encoder	Audio encoder. Set to null to use default Opus encoder.

Returns

Outgoing stream handler.

audioSourceDesc.SamplingRate and voiceInfo.SamplingRate may do not match. Automatic resampling will occur in this case.

3.100.2.3 LocalVoice CreateLocalVoiceAudioFromSource (VoiceInfo *voiceInfo*, IAudioDesc *source*, bool *forceShort* = false, int *channelId* = 0, IEncoder *encoder* = null)

Creates outgoing audio stream of type automatically assigned and adds procedures (callback or serviceable) for consuming given audio source data. Adds audio specific features (e.g. resampling, level meter) to processing pipeline and to returning stream handler.

Parameters

voiceInfo	Outgoing audio stream parameters. Set applicable fields to read them by encoder and by receiving client when voice created.
source	Streaming audio source.
forceShort	For audio sources producing buffers of 'float' type, creates stream of 'short' type and adds converter.
channelId	Transport channel specific to transport.
encoder	Audio encoder. Set to null to use default Opus encoder.

Returns

Outgoing stream handler.

audioSourceDesc.SamplingRate and voiceInfo.SamplingRate may do not match. Automatic resampling will occur in this case.

3.100.2.4 LocalVoiceFramed<T> CreateLocalVoiceFramed< T > (VoiceInfo *voiceInfo,* int *frameSize,* int *channelId* = 0, IEncoder *encoder* = null)

Creates outgoing stream consuming sequence of values passed in array buffers of arbitrary length which repacked in frames of constant length for further processing and encoding.

Template Parameters

T	Type of data consumed by outgoing stream (element type of array buffers).

Parameters

voiceInfo	Outgoing stream parameters. Set applicable fields to read them by encoder and by receiving client when voice created.
frameSize	Size of buffer LocalVoiceFramed repacks input data stream to.
channelId	Transport channel specific to transport.
encoder	Encoder compressing data stream in pipeline.

Returns

Outgoing stream handler.

3.100.2.5 IEnumerable<LocalVoice> LocalVoicesInChannel (int *channelId*)

Iterates through copy of all local voices list of given channel.

3.100.2.6 delegate void RemoteVoiceInfoDelegate (int *channelId,* int *playerId,* byte *voiceId,* VoiceInfo *voiceInfo,* ref RemoteVoiceOptions *options*)

Remote voice info event delegate.

3.100.2.7 void RemoveLocalVoice (LocalVoice *voice*)

Removes local voice (outgoing data stream).

Parameters

voice	Handler of outgoing stream to be removed.

3.100.2.8 void Service ( )

This method dispatches all available incoming commands and then sends this client's outgoing commands. Call this method regularly (2..20 times a second).

3.100.3 Property Documentation

3.100.3.1 int DebugLostPercent [get], [set]

Lost frames simulation ratio.

3.100.3.2 int FramesLost [get], [set]

Lost frames counter.

3.100.3.3 int FramesReceived [get]

Received frames counter.

3.100.3.4 int FramesSent [get]

Sent frames counter.

3.100.3.5 int FramesSentBytes [get]

Sent frames bytes counter.

3.100.3.6 IEnumerable<LocalVoice> LocalVoices [get]

Iterates through copy of all local voices list.

3.100.3.7 RemoteVoiceInfoDelegate OnRemoteVoiceInfoAction [get], [set]

Register a method to be called when remote voice info arrived (after join or new new remote voice creation). Metod parameters: (int channelId, int playerId, byte voiceId, VoiceInfo voiceInfo, ref RemoteVoiceOptions options);

3.100.3.8 IEnumerable<RemoteVoiceInfo> RemoteVoiceInfos [get]

Iterates through all remote voices infos.

3.100.3.9 int RoundTripTime [get]

Average time required voice packet to return to sender.

3.100.3.10 int RoundTripTimeVariance [get]

Average round trip time variation.

3.100.3.11 bool SuppressInfoDuplicateWarning [get], [set]

Do not log warning when duplicate info received.

3.101 VoiceComponent Class Reference

Inherits MonoBehaviour, and ILoggable.

Inherited by PhotonVoiceView, Recorder, Speaker, MicAmplifier, and WebRtcAudioDsp.

Protected Member Functions

- virtual void **Awake** ()

Protected Attributes

- DebugLevel **logLevel** = DebugLevel.ERROR

Properties

- VoiceLogger **Logger** [get, protected set]
- DebugLevel **LogLevel** [get, set]

3.102 VoiceConnection Class Reference

Component that represents a client voice connection to Photon Servers.

Inherits ConnectionHandler, and ILoggable.

Inherited by PhotonVoiceNetwork.

Public Member Functions

- bool ConnectUsingSettings (AppSettings overwriteSettings=null)

  Connect to Photon server using Settings

Public Attributes

- AppSettings Settings

  Settings to be used by this voice connection
- Recorder PrimaryRecorder

  Main Recorder to be used for transmission by default
- Func< int, byte, object, Speaker > SpeakerFactory

  Special factory to link Speaker components with incoming remote audio streams

Protected Member Functions

- override void **Awake** ()
- virtual void **Update** ()
- virtual void **FixedUpdate** ()
- override void **OnDisable** ()
- virtual void **OnDestroy** ()
- virtual Speaker **SimpleSpeakerFactory** (int playerId, byte voiceId, object userData)
- virtual void **OnVoiceStateChanged** (ClientState fromState, ClientState toState)
- void **CalcStatistics** ()
- void **LinkSpeaker** (Speaker speaker, RemoteVoiceLink remoteVoice)

Protected Attributes

- List< RemoteVoiceLink > **cachedRemoteVoices** = new List<RemoteVoiceLink>()

Properties

- VoiceLogger Logger `[get, protected set]`

  Logger used by this component
- DebugLevel LogLevel `[get, set]`

  Log level for this component
- new LoadBalancingTransport **Client** `[get]`
- VoiceClient VoiceClient `[get]`

  Returns underlying Photon Voice client.
- ClientState ClientState `[get]`

  Returns Photon Voice client state.
- float FramesReceivedPerSecond `[get]`

  Number of frames received per second.
- float FramesLostPerSecond `[get]`

  Number of frames lost per second.
- float FramesLostPercent `[get]`

  Percentage of lost frames.
- GameObject SpeakerPrefab `[get, set]`

  Prefab that contains Speaker component to be instantiated when receiving a new remote audio source info

Events

- Action< Speaker > SpeakerLinked

  Fires when a speaker has been linked to a remote audio stream
- Action< RemoteVoiceLink > RemoteVoiceAdded

  Fires when a remote voice stream is added

3.102.1 Detailed Description

Component that represents a client voice connection to Photon Servers.

3.102.2 Member Function Documentation

3.102.2.1 bool ConnectUsingSettings (AppSettings *overwriteSettings* = `null`)

Connect to Photon server using Settings

Parameters

overwrite↩ Settings	Overwrites Settings before connecting

Returns

If true voice connection command was sent from client

3.102.3 Member Data Documentation

3.102.3.1 Recorder PrimaryRecorder

Main Recorder to be used for transmission by default

3.102.3.2 AppSettings Settings

Settings to be used by this voice connection

3.102.3.3 Func<int, byte, object, Speaker> SpeakerFactory

Special factory to link Speaker components with incoming remote audio streams

3.102.4 Property Documentation

3.102.4.1 ClientState ClientState [get]

Returns Photon Voice client state.

3.102.4.2 float FramesLostPercent [get]

Percentage of lost frames.

3.102.4.3 float FramesLostPerSecond [get]

Number of frames lost per second.

3.102.4.4 float FramesReceivedPerSecond [get]

Number of frames received per second.

3.102.4.5 VoiceLogger Logger [get],[protected set]

Logger used by this component

3.102.4.6 DebugLevel LogLevel [get],[set]

Log level for this component

3.102.4.7 GameObject SpeakerPrefab [get],[set]

Prefab that contains Speaker component to be instantiated when receiving a new remote audio source info

3.102.4.8 VoiceClient VoiceClient [get]

Returns underlying Photon Voice client.

3.102.5 Event Documentation

3.102.5.1 Action<RemoteVoiceLink> RemoteVoiceAdded

Fires when a remote voice stream is added

3.102.5.2 Action<Speaker> SpeakerLinked

Fires when a speaker has been linked to a remote audio stream

3.103 AudioUtil.VoiceDetector< T > Class Template Reference

Simple voice activity detector triggered by signal level.

Inherits IProcessor< T >, and AudioUtil.IVoiceDetector.

Public Member Functions

- abstract T[ ] Process (T[ ] buf)

  Process a frame of audio data.
- void **Dispose** ()

Protected Attributes

- int **activityDelay**
- int **autoSilenceCounter** = 0
- int **valuesCountPerSec**
- int **activityDelayValuesCount**

Properties

- bool On [get, set]

  If true, voice detection enabled.
- float Threshold [get, set]

  Voice detected as soon as signal level exceeds threshold.
- bool Detected [get, protected set]

  If true, voice detected.
- DateTime DetectedTime [get]

  Last time when switched to detected state.
- int ActivityDelayMs [get, set]

  Keep detected state during this time after signal level dropped below threshold.

Events

- Action OnDetected

  Called when switched to detected state.

3.103.1 Detailed Description

Simple voice activity detector triggered by signal level.

3.103.2 Member Function Documentation

3.103.2.1 abstract T [ ] Process (T[ ] *buf*) [pure virtual]

Process a frame of audio data.

Parameters

buf	Buffer containing input audio data

Returns

Buffer containing output audio data

Implements IProcessor< T >.

3.103.3 Property Documentation

3.103.3.1 int ActivityDelayMs [get], [set]

Keep detected state during this time after signal level dropped below threshold.

3.103.3.2 bool Detected [get], [protected set]

If true, voice detected.

3.103.3.3 DateTime DetectedTime [get]

Last time when switched to detected state.

3.103.3.4 bool On [get], [set]

If true, voice detection enabled.

3.103.3.5 float Threshold [get], [set]

Voice detected as soon as signal level exceeds threshold.

3.103.4 Event Documentation

3.103.4.1 Action OnDetected

Called when switched to detected state.

3.104 AudioUtil.VoiceDetectorCalibration< T > Class Template Reference

Calibration Utility for Voice Detector

Inherits IProcessor< T >.

Public Member Functions

- VoiceDetectorCalibration (IVoiceDetector voiceDetector, ILevelMeter levelMeter, int samplingRate, int channels)

  Create new VoiceDetectorCalibration instance.
- void Calibrate (int durationMs, Action< float > onCalibrated=null)

  Start calibration.
- T[ ] Process (T[ ] buf)

  Process a frame of audio data.
- void **Dispose** ()

Protected Attributes

- int **calibrateCount**

Properties

- bool **IsCalibrating** `[get]`

3.104.1 Detailed Description

Calibration Utility for Voice Detector

Using this audio processor, you can calibrate the IVoiceDetector.Threshold.

3.104.2 Constructor & Destructor Documentation

3.104.2.1 VoiceDetectorCalibration (IVoiceDetector *voiceDetector,* ILevelMeter *levelMeter,* int *samplingRate,* int *channels*)

Create new VoiceDetectorCalibration instance.

Parameters

voiceDetector	Voice Detector to calibrate.
levelMeter	Level Meter to look at for calibration.
samplingRate	Sampling rate of the audio signal (in Hz).
numChannels	Number of channels in the audio signal.

3.104.3 Member Function Documentation

3.104.3.1 void Calibrate (int *durationMs,* Action< float > *onCalibrated* = `null`)

Start calibration.

Parameters

durationMs	Duration of the calibration procedure (in milliseconds).

This activates the Calibration process. It will reset the given LevelMeter's AccumAvgPeakAmp (accumulated average peak amplitude), and when the duration has passed, use it for the VoiceDetector's detection threshold.

3.104.3.2 T [ ] Process (T[ ] *buf*)

Process a frame of audio data.

Parameters

buf	Buffer containing input audio data

Returns

Buffer containing output audio data

Implements IProcessor< T >.

3.105 AudioUtil.VoiceDetectorDummy Class Reference

Dummy VoiceDetector that doesn't actually do anything.

Inherits AudioUtil.IVoiceDetector.

Properties

- bool **On** `[get, set]`
- float **Threshold** `[get, set]`
- bool **Detected** `[get]`
- int **ActivityDelayMs** `[get, set]`
- DateTime **DetectedTime** `[get]`
- Action **OnDetected**

Additional Inherited Members

3.105.1 Detailed Description

Dummy VoiceDetector that doesn't actually do anything.

3.106 AudioUtil.VoiceDetectorFloat Class Reference

VoiceDetector specialization for float audio.

Inherits AudioUtil.VoiceDetector< float >.

Public Member Functions

- VoiceDetectorFloat (int samplingRate, int numChannels)

  Create a new VoiceDetectorFloat instance.
- override float[ ] **Process** (float[ ] buffer)

Additional Inherited Members

3.106.1 Detailed Description

VoiceDetector specialization for float audio.

3.106.2 Constructor & Destructor Documentation

3.106.2.1 VoiceDetectorFloat (int *samplingRate,* int *numChannels*)

Create a new VoiceDetectorFloat instance.

Parameters

samplingRate	Sampling rate of the audio signal (in Hz).
numChannels	Number of channels in the audio signal.

3.107 AudioUtil.VoiceDetectorShort Class Reference

VoiceDetector specialization for float audio.

Inherits AudioUtil.VoiceDetector< short >.

Public Member Functions

- VoiceDetectorShort (int samplingRate, int numChannels)
  Create a new VoiceDetectorFloat instance
- override short[ ] **Process** (short[ ] buffer)

Additional Inherited Members

3.107.1 Detailed Description

VoiceDetector specialization for float audio.

3.107.2 Constructor & Destructor Documentation

3.107.2.1 VoiceDetectorShort (int *samplingRate,* int *numChannels*)

Create a new VoiceDetectorFloat instance

Parameters

samplingRate	Sampling rate of the audio signal (in Hz).
numChannels	Number of channels in the audio signal.

3.108 VoiceEvent Class Reference

Public Attributes

- const byte Code = 202
  Single event used for voice communications.

3.108.1 Member Data Documentation

3.108.1.1 const byte Code = 202

Single event used for voice communications.

Change if it conflicts with other event codes used in the same Photon room.

3.109 VoiceInfo Struct Reference

Describes stream properties.

Public Member Functions

- override string **ToString** ()

Static Public Member Functions

- static VoiceInfo CreateAudioOpus (POpusCodec.Enums.SamplingRate samplingRate, int channels, Opus↩ Codec.FrameDuration frameDurationUs, int bitrate, object userdata=null)

  Create stream info for an Opus audio stream.

Properties

- Codec **Codec** `[get, set]`
- int SamplingRate `[get, set]`

  Audio sampling rate (frequency, in Hz).
- int Channels `[get, set]`

  Source audio sampling rate (to be resampled to SamplingRate; in Hz).
- int FrameDurationUs `[get, set]`

  Uncompressed frame (audio packet) size in microseconds.
- int Bitrate `[get, set]`

  Target bitrate (in bits/second).
- object UserData `[get, set]`

  Optional user data. Should be serializable by Photon.
- int FrameDurationSamples `[get]`

  Uncompressed frame (data packet) size in samples.
- int FrameSize `[get]`

  Uncompressed frame (data packet) array size.
- int Width `[get, set]`

  Video width (optional).
- int Height `[get, set]`

  Video height (optional)

3.109.1 Detailed Description

Describes stream properties.

3.109.2 Member Function Documentation

3.109.2.1 static VoiceInfo CreateAudioOpus (POpusCodec.Enums.SamplingRate *samplingRate,* int *channels,* OpusCodec.FrameDuration *frameDurationUs,* int *bitrate,* object *userdata* = `null`) [static]

Create stream info for an Opus audio stream.

Parameters

samplingRate	Audio sampling rate.
channels	Number of channels.
frameDurationUs	Uncompressed frame (audio packet) size in microseconds.
bitrate	Stream bitrate (in bits/second).
userdata	Optional user data. Should be serializable by Photon.

Returns

VoiceInfo instance.

3.109.3 Property Documentation

3.109.3.1 int Bitrate [get], [set]

Target bitrate (in bits/second).

3.109.3.2 int Channels [get], [set]

Source audio sampling rate (to be resampled to SamplingRate; in Hz).

3.109.3.3 int FrameDurationSamples [get]

Uncompressed frame (data packet) size in samples.

3.109.3.4 int FrameDurationUs [get], [set]

Uncompressed frame (audio packet) size in microseconds.

3.109.3.5 int FrameSize [get]

Uncompressed frame (data packet) array size.

3.109.3.6 int Height [get], [set]

Video height (optional)

3.109.3.7 int SamplingRate [get], [set]

Audio sampling rate (frequency, in Hz).

3.109.3.8 object UserData [get], [set]

Optional user data. Should be serializable by Photon.

3.109.3.9 int Width [get], [set]

Video width (optional).

3.110 AudioUtil.VoiceLevelDetectCalibrate< T > Class Template Reference

Utility Audio Processor Voice Detection Calibration.

Inherits IProcessor< T >.

Public Member Functions

- VoiceLevelDetectCalibrate (int samplingRate, int channels)
  Create new VoiceLevelDetectCalibrate instance
- void Calibrate (int durationMs, Action< float > onCalibrated=null)
  Start calibration
- T[ ] Process (T[ ] buf)
  Process a frame of audio data.
- void **Dispose** ()

Properties

- ILevelMeter LevelMeter `[get]`
  The LevelMeter in use.
- IVoiceDetector VoiceDetector `[get]`
  The VoiceDetector in use
- bool **IsCalibrating** `[get]`

3.110.1 Detailed Description

Utility Audio Processor Voice Detection Calibration.

Encapsulates level meter, voice detector and voice detector calibrator in single instance.

3.110.2 Constructor & Destructor Documentation

3.110.2.1 VoiceLevelDetectCalibrate (int *samplingRate*, int *channels*)

Create new VoiceLevelDetectCalibrate instance

Parameters

samplingRate	Sampling rate of the audio signal (in Hz).
numChannels	Number of channels in the audio signal.

3.110.3 Member Function Documentation

3.110.3.1 void Calibrate (int *durationMs*, Action< float > *onCalibrated* = `null`)

Start calibration

Parameters

durationMs	Duration of the calibration procedure (in milliseconds).

onCalibrated	Called when calibration is complete. Parameter is new threshold value.

This activates the Calibration process. It will reset the given LevelMeter's AccumAvgPeakAmp (accumulated average peak amplitude), and when the duration has passed, use it for the VoiceDetector's detection threshold.

3.110.3.2 T [ ] Process (T[ ] *buf*)

Process a frame of audio data.

Parameters

buf	Buffer containing input audio data

Returns

Buffer containing output audio data

Implements IProcessor< T >.

3.110.4 Property Documentation

3.110.4.1 ILevelMeter LevelMeter [get]

The LevelMeter in use.

3.110.4.2 IVoiceDetector VoiceDetector [get]

The VoiceDetector in use

3.111 VoiceLogger Class Reference

Inherits ILogger.

Public Member Functions

- **VoiceLogger** (Object context, string tag, DebugLevel level=DebugLevel.ERROR)
- **VoiceLogger** (string tag, DebugLevel level=DebugLevel.ERROR)
- void **LogError** (string fmt, params object[ ] args)
- void **LogWarning** (string fmt, params object[ ] args)
- void **LogInfo** (string fmt, params object[ ] args)
- void **LogDebug** (string fmt, params object[ ] args)

Properties

- string **Tag** [get, set]
- DebugLevel **LogLevel** [get, set]
- bool **IsErrorEnabled** [get]
- bool **IsWarningEnabled** [get]
- bool **IsInfoEnabled** [get]
- bool **IsDebugEnabled** [get]

3.112 WebRtcAudioDsp Class Reference

Inherits VoiceComponent.

Protected Member Functions

- override void **Awake** ()

Properties

- bool **AEC** `[get, set]`
- bool **AECMobile** `[get, set]`
- bool **AECMobileComfortNoise** `[get, set]`
- int **ReverseStreamDelayMs** `[get, set]`
- bool **NoiseSuppression** `[get, set]`
- bool **HighPass** `[get, set]`
- bool **Bypass** `[get, set]`
- bool **AGC** `[get, set]`
- bool **VAD** `[get, set]`

Additional Inherited Members

3.113 WebRTCAudioLib Class Reference

Inherited by WebRTCAudioProcessor.

Classes

- struct ConfigParam
- struct Param

Public Member Functions

- static IntPtr **webrtc_audio_processor_create** (int samplingRate, int channels, int frameSize, int rev↩ SamplingRate, int revChannels)
- static int **webrtc_audio_processor_set_config_param** (IntPtr proc, int param, int v)
- static int **webrtc_audio_processor_init** (IntPtr proc)
- static int **webrtc_audio_processor_set_param** (IntPtr proc, int param, int v)
- static int **webrtc_audio_processor_process** (IntPtr proc, short[ ] buffer, int offset, out bool voiceDetected)
- static int **webrtc_audio_processor_process_reverse** (IntPtr proc, short[ ] buffer, int bufferSize)
- static void **webrtc_audio_processor_destroy** (IntPtr proc)

3.114 WebRTCAudioProcessor Class Reference

Inherits WebRTCAudioLib, and IProcessor< short >.

Public Member Functions

- **WebRTCAudioProcessor** (ILogger logger, int frameSize, int samplingRate, int channels, int reverse↩ SamplingRate, int reverseChannels)
- short[ ] **Process** (short[ ] buf)
- void **OnAudioOutFrameFloat** (float[ ] data)
- void **Dispose** ()

Properties

- int **AECStreamDelayMs** `[set]`
- bool **AEC** `[set]`
- bool **AECMobile** `[set]`
- int **AECMRoutingMode** `[set]`
- bool **AECMComfortNoise** `[set]`
- bool **HighPass** `[set]`
- bool **NoiseSuppression** `[set]`
- bool **AGC** `[set]`
- bool **VAD** `[set]`
- bool **Bypass** `[set]`

Index