Photon Voice

v2.24

Generated by Doxygen 1.8.17

Chapter 1

Main Page

Photon Voice 2 has three key classes:

- Photon.Voice.Unity.VoiceConnection (extends `Photon.Realtime.LoadBalancingClient`)
- Photon.Voice.Unity.Recorder
- Photon.Voice.Unity.Speaker

If you also use the integration with PUN 2, we added two components for ease-of-use and more convenience:

- Photon.Voice.PUN.PhotonVoiceNetwork
- Photon.Voice.PUN.PhotonVoiceView

Photon Voice 2 also comes with a WebRTC based DSP (Photon.Voice.Unity.WebRtcAudioDsp using Photon.Voice.WebRTCAudioProcessor).

Read more in the official documentation `here`.
You can download Photon Voice 2 `here`.

Chapter 2

NOTE:

This license does not apply to the `CSCore.Ffmpeg` project!

2.1 Microsoft Public License (Ms-PL)

Microsoft Public License (Ms-PL)

This license governs use of the accompanying software. If you use the software, you accept this license. If you do not accept the license, do not use the software.

2.1.0.1 1. Definitions

The terms "reproduce," "reproduction," "derivative works," and "distribution" have the same meaning here as under U.S. copyright law.

A "contribution" is the original software, or any additions or changes to the software.

A "contributor" is any person that distributes its contribution under this license.

"Licensed patents" are a contributor's patent claims that read directly on its contribution.

2.1.0.2 2. Grant of Rights

(A) Copyright Grant- Subject to the terms of this license, including the license conditions and limitations in section 3, each contributor grants you a non-exclusive, worldwide, royalty-free copyright license to reproduce its contribution, prepare derivative works of its contribution, and distribute its contribution or any derivative works that you create.

(B) Patent Grant- Subject to the terms of this license, including the license conditions and limitations in section 3, each contributor grants you a non-exclusive, worldwide, royalty-free license under its licensed patents to make, have made, use, sell, offer for sale, import, and/or otherwise dispose of its contribution in the software or derivative works of the contribution in the software.

2.1.0.3 3. Conditions and Limitations

(A) No Trademark License- This license does not grant you rights to use any contributors' name, logo, or trademarks.

(B) If you bring a patent claim against any contributor over patents that you claim are infringed by the software, your patent license from such contributor to the software ends automatically.

(C) If you distribute any portion of the software, you must retain all copyright, patent, trademark, and attribution notices that are present in the software.

(D) If you distribute any portion of the software in source code form, you may do so only under this license by including a complete copy of this license with your distribution. If you distribute any portion of the software in compiled or object code form, you may only do so under a license that complies with this license.

(E) The software is licensed "as-is." You bear the risk of using it. The contributors give no express warranties, guarantees or conditions. You may have additional consumer rights under your local laws which this license cannot change. To the extent permitted under your local laws, the contributors exclude the implied warranties of merchantability, fitness for a particular purpose and non-infringement.

Chapter 3

Namespace Documentation

3.1 CSCore Namespace Reference

Classes

- class AudioSubTypes

  Defines AudioSubTypes and provides methods to convert between AudioEncoding-values and AudioSubTypes-values.
- class Extensions

  Provides a few basic extensions.
- interface IAudioSource

  Defines the base for all audio streams.
- interface IReadableAudioSource

  Defines a generic base for all readable audio streams.
- interface IWaveSource

  Defines the base for all audio streams which provide raw byte data.
- interface IWriteable

  Provides the Write method.
- class WaveFormat

  Defines the format of waveform-audio data.
- class WaveFormatExtensible

  Defines the format of waveform-audio data for formats having more than two channels or higher sample resolutions than allowed by WaveFormat. Can be used to define any format that can be defined by WaveFormat. For more information see and .

Enumerations

- enum AudioEncoding : short

  Defines all known encoding types. Primary used in the WaveFormat class. See WaveFormat.WaveFormatTag.
- enum ChannelMask

  Channelmask used by WaveFormatExtensible. For more information see `http://msdn.microsoft.com/en-us/library/windows/desktop/dd757714(v=vs.85).aspx`

3.1.1 Enumeration Type Documentation

3.1.1.1 AudioEncoding

```
enum AudioEncoding : short [strong]
```

Defines all known encoding types. Primary used in the WaveFormat class. See WaveFormat.WaveFormatTag.

Enumerator

Unknown	WAVE_FORMAT_UNKNOWN, Microsoft Corporation
Pcm	WAVE_FORMAT_PCM Microsoft Corporation
Adpcm	WAVE_FORMAT_ADPCM Microsoft Corporation
IeeeFloat	WAVE_FORMAT_IEEE_FLOAT Microsoft Corporation
Vselp	WAVE_FORMAT_VSELP Compaq Computer Corp.
IbmCvsd	WAVE_FORMAT_IBM_CVSD IBM Corporation
ALaw	WAVE_FORMAT_ALAW Microsoft Corporation
MuLaw	WAVE_FORMAT_MULAW Microsoft Corporation
Dts	WAVE_FORMAT_DTS Microsoft Corporation
Drm	WAVE_FORMAT_DRM Microsoft Corporation
WmaVoice9	WAVE_FORMAT_WMAVOICE9
OkiAdpcm	WAVE_FORMAT_OKI_ADPCM OKI
DviAdpcm	WAVE_FORMAT_DVI_ADPCM Intel Corporation
ImaAdpcm	WAVE_FORMAT_IMA_ADPCM Intel Corporation
MediaspaceAdpcm	WAVE_FORMAT_MEDIASPACE_ADPCM Videologic
SierraAdpcm	WAVE_FORMAT_SIERRA_ADPCM Sierra Semiconductor Corp
G723Adpcm	WAVE_FORMAT_G723_ADPCM Antex Electronics Corporation
DigiStd	WAVE_FORMAT_DIGISTD DSP Solutions, Inc.
DigiFix	WAVE_FORMAT_DIGIFIX DSP Solutions, Inc.
DialogicOkiAdpcm	WAVE_FORMAT_DIALOGIC_OKI_ADPCM Dialogic Corporation
MediaVisionAdpcm	WAVE_FORMAT_MEDIAVISION_ADPCM Media Vision, Inc.
CUCodec	WAVE_FORMAT_CU_CODEC Hewlett-Packard Company
YamahaAdpcm	WAVE_FORMAT_YAMAHA_ADPCM Yamaha Corporation of America
SonarC	WAVE_FORMAT_SONARC Speech Compression
DspGroupTrueSpeech	WAVE_FORMAT_DSPGROUP_TRUESPEECH DSP Group, Inc
EchoSpeechCorporation1	WAVE_FORMAT_ECHOSC1 Echo Speech Corporation
AudioFileAf36	WAVE_FORMAT_AUDIOFILE_AF36, Virtual Music, Inc.
Aptx	WAVE_FORMAT_APTX Audio Processing Technology
AudioFileAf10	WAVE_FORMAT_AUDIOFILE_AF10, Virtual Music, Inc.
Prosody1612	WAVE_FORMAT_PROSODY_1612, Aculab plc
Lrc	WAVE_FORMAT_LRC, Merging Technologies S.A.
DolbyAc2	WAVE_FORMAT_DOLBY_AC2, Dolby Laboratories

Enumerator

Gsm610	WAVE_FORMAT_GSM610, Microsoft Corporation
MsnAudio	WAVE_FORMAT_MSNAUDIO, Microsoft Corporation
AntexAdpcme	WAVE_FORMAT_ANTEX_ADPCME, Antex Electronics Corporation
ControlResVqlpc	WAVE_FORMAT_CONTROL_RES_VQLPC, Control Resources Limited
DigiReal	WAVE_FORMAT_DIGIREAL, DSP Solutions, Inc.
DigiAdpcm	WAVE_FORMAT_DIGIADPCM, DSP Solutions, Inc.
ControlResCr10	WAVE_FORMAT_CONTROL_RES_CR10, Control Resources Limited
WAVE_FORMAT_NMS_VBXADPCM	WAVE_FORMAT_NMS_VBXADPCM
WAVE_FORMAT_CS_IMAADPCM	WAVE_FORMAT_CS_IMAADPCM
WAVE_FORMAT_ECHOSC3	WAVE_FORMAT_ECHOSC3
WAVE_FORMAT_ROCKWELL_ADPCM	WAVE_FORMAT_ROCKWELL_ADPCM
WAVE_FORMAT_ROCKWELL_DIGITALK	WAVE_FORMAT_ROCKWELL_DIGITALK
WAVE_FORMAT_XEBEC	WAVE_FORMAT_XEBEC
WAVE_FORMAT_G721_ADPCM	WAVE_FORMAT_G721_ADPCM
WAVE_FORMAT_G728_CELP	WAVE_FORMAT_G728_CELP
WAVE_FORMAT_MSG723	WAVE_FORMAT_MSG723
Mpeg	WAVE_FORMAT_MPEG, Microsoft Corporation
WAVE_FORMAT_RT24	WAVE_FORMAT_RT24
WAVE_FORMAT_PAC	WAVE_FORMAT_PAC
MpegLayer3	WAVE_FORMAT_MPEGLAYER3, ISO/MPEG Layer3 Format Tag
WAVE_FORMAT_LUCENT_G723	WAVE_FORMAT_LUCENT_G723
WAVE_FORMAT_CIRRUS	WAVE_FORMAT_CIRRUS
WAVE_FORMAT_ESPCM	WAVE_FORMAT_ESPCM
WAVE_FORMAT_VOXWARE	WAVE_FORMAT_VOXWARE
WAVE_FORMAT_CANOPUS_ATRAC	WAVE_FORMAT_CANOPUS_ATRAC
WAVE_FORMAT_G726_ADPCM	WAVE_FORMAT_G726_ADPCM
WAVE_FORMAT_G722_ADPCM	WAVE_FORMAT_G722_ADPCM
WAVE_FORMAT_DSAT_DISPLAY	WAVE_FORMAT_DSAT_DISPLAY
WAVE_FORMAT_VOXWARE_BYTE_ALIGNED	WAVE_FORMAT_VOXWARE_BYTE_ALIGNED
WAVE_FORMAT_VOXWARE_AC8	WAVE_FORMAT_VOXWARE_AC8
WAVE_FORMAT_VOXWARE_AC10	WAVE_FORMAT_VOXWARE_AC10
WAVE_FORMAT_VOXWARE_AC16	WAVE_FORMAT_VOXWARE_AC16
WAVE_FORMAT_VOXWARE_AC20	WAVE_FORMAT_VOXWARE_AC20
WAVE_FORMAT_VOXWARE_RT24	WAVE_FORMAT_VOXWARE_RT24
WAVE_FORMAT_VOXWARE_RT29	WAVE_FORMAT_VOXWARE_RT29
WAVE_FORMAT_VOXWARE_RT29HW	WAVE_FORMAT_VOXWARE_RT29HW
WAVE_FORMAT_VOXWARE_VR12	WAVE_FORMAT_VOXWARE_VR12
WAVE_FORMAT_VOXWARE_VR18	WAVE_FORMAT_VOXWARE_VR18
WAVE_FORMAT_VOXWARE_TQ40	WAVE_FORMAT_VOXWARE_TQ40
WAVE_FORMAT_SOFTSOUND	WAVE_FORMAT_SOFTSOUND
WAVE_FORMAT_VOXWARE_TQ60	WAVE_FORMAT_VOXWARE_TQ60
WAVE_FORMAT_MSRT24	WAVE_FORMAT_MSRT24
WAVE_FORMAT_G729A	WAVE_FORMAT_G729A
WAVE_FORMAT_MVI_MVI2	WAVE_FORMAT_MVI_MVI2
WAVE_FORMAT_DF_G726	WAVE_FORMAT_DF_G726

Enumerator

WAVE_FORMAT_DF_GSM610	WAVE_FORMAT_DF_GSM610
WAVE_FORMAT_ISIAUDIO	WAVE_FORMAT_ISIAUDIO
WAVE_FORMAT_ONLIVE	WAVE_FORMAT_ONLIVE
WAVE_FORMAT_SBC24	WAVE_FORMAT_SBC24
WAVE_FORMAT_DOLBY_AC3_SPDIF	WAVE_FORMAT_DOLBY_AC3_SPDIF
WAVE_FORMAT_MEDIASONIC_G723	WAVE_FORMAT_MEDIASONIC_G723
WAVE_FORMAT_PROSODY_8KBPS	WAVE_FORMAT_PROSODY_8KBPS
WAVE_FORMAT_ZYXEL_ADPCM	WAVE_FORMAT_ZYXEL_ADPCM
WAVE_FORMAT_PHILIPS_LPCBB	WAVE_FORMAT_PHILIPS_LPCBB
WAVE_FORMAT_PACKED	WAVE_FORMAT_PACKED
WAVE_FORMAT_MALDEN_PHONYTALK	WAVE_FORMAT_MALDEN_PHONYTALK
Gsm	WAVE_FORMAT_GSM
G729	WAVE_FORMAT_G729
G723	WAVE_FORMAT_G723
Acelp	WAVE_FORMAT_ACELP
RawAac	WAVE_FORMAT_RAW_AAC1
WAVE_FORMAT_RHETOREX_ADPCM	WAVE_FORMAT_RHETOREX_ADPCM
WAVE_FORMAT_IRAT	WAVE_FORMAT_IRAT
WAVE_FORMAT_VIVO_G723	WAVE_FORMAT_VIVO_G723
WAVE_FORMAT_VIVO_SIREN	WAVE_FORMAT_VIVO_SIREN
WAVE_FORMAT_DIGITAL_G723	WAVE_FORMAT_DIGITAL_G723
WAVE_FORMAT_SANYO_LD_ADPCM	WAVE_FORMAT_SANYO_LD_ADPCM
WAVE_FORMAT_SIPROLAB_ACEPLNET	WAVE_FORMAT_SIPROLAB_ACEPLNET
WAVE_FORMAT_SIPROLAB_ACELP4800	WAVE_FORMAT_SIPROLAB_ACELP4800
WAVE_FORMAT_SIPROLAB_ACELP8V3	WAVE_FORMAT_SIPROLAB_ACELP8V3
WAVE_FORMAT_SIPROLAB_G729	WAVE_FORMAT_SIPROLAB_G729
WAVE_FORMAT_SIPROLAB_G729A	WAVE_FORMAT_SIPROLAB_G729A
WAVE_FORMAT_SIPROLAB_KELVIN	WAVE_FORMAT_SIPROLAB_KELVIN
WAVE_FORMAT_G726ADPCM	WAVE_FORMAT_G726ADPCM
WAVE_FORMAT_QUALCOMM_PUREVOICE	WAVE_FORMAT_QUALCOMM_PUREVOICE
WAVE_FORMAT_QUALCOMM_HALFRATE	WAVE_FORMAT_QUALCOMM_HALFRATE
WAVE_FORMAT_TUBGSM	WAVE_FORMAT_TUBGSM
WAVE_FORMAT_MSAUDIO1	WAVE_FORMAT_MSAUDIO1
WindowsMediaAudio	Windows Media Audio, WAVE_FORMAT_WMAUDIO2, Microsoft Corporation
WindowsMediaAudioProfessional	Windows Media Audio Professional WAVE_FORMAT_WMAUDIO3, Microsoft Corporation
WindowsMediaAudioLosseless	Windows Media Audio Lossless, WAVE_FORMAT_WMAUDIO_LOSSLESS
WindowsMediaAudioSpdif	Windows Media Audio Professional over SPDIF WAVE_FORMAT_WMASPDIF (0x0164)
WAVE_FORMAT_UNISYS_NAP_ADPCM	WAVE_FORMAT_UNISYS_NAP_ADPCM
WAVE_FORMAT_UNISYS_NAP_ULAW	WAVE_FORMAT_UNISYS_NAP_ULAW
WAVE_FORMAT_UNISYS_NAP_ALAW	WAVE_FORMAT_UNISYS_NAP_ALAW
WAVE_FORMAT_UNISYS_NAP_16K	WAVE_FORMAT_UNISYS_NAP_16K
WAVE_FORMAT_CREATIVE_ADPCM	WAVE_FORMAT_CREATIVE_ADPCM
WAVE_FORMAT_CREATIVE_FASTSPEECH8	WAVE_FORMAT_CREATIVE_FASTSPEECH8
WAVE_FORMAT_CREATIVE_FASTSPEECH10	WAVE_FORMAT_CREATIVE_FASTSPEECH10
WAVE_FORMAT_UHER_ADPCM	WAVE_FORMAT_UHER_ADPCM

Enumerator

WAVE_FORMAT_QUARTERDECK	WAVE_FORMAT_QUARTERDECK
WAVE_FORMAT_ILINK_VC	WAVE_FORMAT_ILINK_VC
WAVE_FORMAT_RAW_SPORT	WAVE_FORMAT_RAW_SPORT
WAVE_FORMAT_ESST_AC3	WAVE_FORMAT_ESST_AC3
WAVE_FORMAT_IPI_HSX	WAVE_FORMAT_IPI_HSX
WAVE_FORMAT_IPI_RPELP	WAVE_FORMAT_IPI_RPELP
WAVE_FORMAT_CS2	WAVE_FORMAT_CS2
WAVE_FORMAT_SONY_SCX	WAVE_FORMAT_SONY_SCX
WAVE_FORMAT_FM_TOWNS_SND	WAVE_FORMAT_FM_TOWNS_SND
WAVE_FORMAT_BTV_DIGITAL	WAVE_FORMAT_BTV_DIGITAL
WAVE_FORMAT_QDESIGN_MUSIC	WAVE_FORMAT_QDESIGN_MUSIC
WAVE_FORMAT_VME_VMPCM	WAVE_FORMAT_VME_VMPCM
WAVE_FORMAT_TPC	WAVE_FORMAT_TPC
WAVE_FORMAT_OLIGSM	WAVE_FORMAT_OLIGSM
WAVE_FORMAT_OLIADPCM	WAVE_FORMAT_OLIADPCM
WAVE_FORMAT_OLICELP	WAVE_FORMAT_OLICELP
WAVE_FORMAT_OLISBC	WAVE_FORMAT_OLISBC
WAVE_FORMAT_OLIOPR	WAVE_FORMAT_OLIOPR
WAVE_FORMAT_LH_CODEC	WAVE_FORMAT_LH_CODEC
WAVE_FORMAT_NORRIS	WAVE_FORMAT_NORRIS
WAVE_FORMAT_SOUNDSPACE_MUSICOMPR↩ ESS	WAVE_FORMAT_SOUNDSPACE_MUSICOMPRE↩ SS
MPEG_ADTS_AAC	Advanced Audio Coding (AAC) audio in Audio Data Transport Stream (ADTS) format. The format block is a WAVEFORMATEX structure with wFormatTag equal to WAVE_FORMAT_MPEG_ADTS_AAC. The WAVEFORMATEX structure specifies the core AAC-LC sample rate and number of channels, prior to applying spectral band replication (SBR) or parametric stereo (PS) tools, if present. No additional data is required after the WAVEFORMATEX structure. http://msdn.microsoft.↩ com/en-us/library/dd317599%28VS.↩ 85%29.aspx
MPEG_RAW_AAC	MPEG_RAW_AAC Source wmCodec.h
MPEG_LOAS	MPEG-4 audio transport stream with a synchronization layer (LOAS) and a multiplex layer (LATM). The format block is a WAVEFORMATEX structure with wFormatTag equal to WAVE_FORMAT_MPEG_LOAS. See . The WAVEFORMATEX structure specifies the core AAC-LC sample rate and number of channels, prior to applying spectral SBR or PS tools, if present. No additional data is required after the WAVEFORMATEX structure.
NOKIA_MPEG_ADTS_AAC	NOKIA_MPEG_ADTS_AAC Source wmCodec.h
NOKIA_MPEG_RAW_AAC	NOKIA_MPEG_RAW_AAC Source wmCodec.h
VODAFONE_MPEG_ADTS_AAC	VODAFONE_MPEG_ADTS_AAC Source wmCodec.h
VODAFONE_MPEG_RAW_AAC	VODAFONE_MPEG_RAW_AAC Source wmCodec.h

Enumerator

MPEG_HEAAC	High-Efficiency Advanced Audio Coding (HE-AAC) stream. The format block is an HEAACWAVEFORMAT structure. See .
WAVE_FORMAT_DVM	WAVE_FORMAT_DVM
Vorbis1	WAVE_FORMAT_VORBIS1 "Og" Original stream compatible
Vorbis2	WAVE_FORMAT_VORBIS2 "Pg" Have independent header
Vorbis3	WAVE_FORMAT_VORBIS3 "Qg" Have no codebook header
Vorbis1P	WAVE_FORMAT_VORBIS1P "og" Original stream compatible
Vorbis2P	WAVE_FORMAT_VORBIS2P "pg" Have independent headere
Vorbis3P	WAVE_FORMAT_VORBIS3P "qg" Have no codebook header
WAVE_FORMAT_RAW_AAC1	Raw AAC1
WAVE_FORMAT_WMAVOICE9	Windows Media Audio Voice (WMA Voice)
Extensible	Extensible
WAVE_FORMAT_DEVELOPMENT	WAVE_FORMAT_DEVELOPMENT
WAVE_FORMAT_FLAC	FLAC

3.1.1.2 ChannelMask

`enum ChannelMask [strong]`

Channelmask used by WaveFormatExtensible. For more information see `http://msdn.microsoft.↩ com/en-us/library/windows/desktop/dd757714(v=vs.85).aspx`

Enumerator

SpeakerFrontLeft	Front left speaker.
SpeakerFrontRight	Front right speaker.
SpeakerFrontCenter	Front center speaker.
SpeakerLowFrequency	Low frequency speaker.
SpeakerBackLeft	Back left speaker.
SpeakerBackRight	Back right speaker.
SpeakerFrontLeftOfCenter	Front left of center speaker.
SpeakerFrontRightOfCenter	Front right of center speaker.
SpeakerBackCenter	Back center speaker.
SpeakerSideLeft	Side left speaker.
SpeakerSideRight	Side right speaker.
SpeakerTopCenter	Top center speaker.
SpeakerTopFrontLeft	Top front left speaker.
SpeakerTopFrontCenter	Top front center speaker.
SpeakerTopFrontRight	Top front right speaker.
SpeakerTopBackLeft	Top back left speaker.
SpeakerTopBackCenter	Top back center speaker.
SpeakerTopBackRight	Top back right speaker.

3.2 CSCore.Codecs Namespace Reference

3.3 CSCore.Codecs.WAV Namespace Reference

Classes

- class WaveWriter

  Encoder for wave files.

3.4 Photon Namespace Reference

3.5 Photon.Voice Namespace Reference

Classes

- class AudioDesc
- class AudioInChangeNotifier
- class AudioInEnumerator

  Enumerates microphones available on device.
- class AudioSyncBuffer
- 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 BufferReaderPushAdapterAsyncPoolShortToFloat

  BufferReaderPushAdapter implementation using asynchronous LocalVoice.PushDataAsync, converting short samples to float.
- 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.
- struct FrameBuffer
- class FrameOut
- 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 IDecoderDirect

  Interface for an decoder which outputs data via explicit call.
- interface IDecoderQueuedOutputImageNative
- interface IEncoder

  Generic encoder interface.
- interface IEncoderDirect

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

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

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

  Audio Processor interface.
- interface IResettable
- interface IServiceable

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

  Extends LoadBalancingClient with media streaming functionality.
- class LoadBalancingTransport2

  Variant of LoadBalancingTransport. Aims to be non-alloc at the cost of breaking compatibility with older clients.
- 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

- class MonoPInvokeCallbackAttribute
- 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 Platform
- class PrimitiveArrayPool

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

  The type of samples used for audio processing.
- enum **FrameFlags** : byte
- enum Codec

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

3.5.1 Enumeration Type Documentation

3.5.1.1 AudioSampleType

```
enum AudioSampleType  [strong]
```

The type of samples used for audio processing.

3.5.1.2 Codec

```
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

3.6 Photon.Voice.IOS Namespace Reference

Classes

- struct AudioSessionParameters
- class AudioSessionParametersPresets

Enumerations

- enum AudioSessionCategory
- enum AudioSessionMode
- enum AudioSessionCategoryOption

3.6.1 Enumeration Type Documentation

3.6.1.1 AudioSessionCategory

```
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_AVAILABLE(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_UNAVAILABLE(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);

Enumerator

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_UNAVAILABLE(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);

3.6.1.2 AudioSessionCategoryOption

```
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) AVAudioSessionCategoryPlayAndRecord 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 AVAudioSessionCategoryMultiRoute
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 AVAudioSessionCategoryMultiRoute, by default the audio session will be non-mixable and non-ducking. Setting this option will also make your category mixable with others (AVAudioSessionCategoryOptionMixWithOthers will be set). DuckOthers is only valid with AVAudioSessionCategoryAmbient, AVAudioSessionCategoryPlayAndRecord, AVAudioSessionCategoryPlayback, and AVAudioSessionCategoryMultiRoute

Enumerator

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) AVAudioSessionCategoryPlayAndRecord 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

3.6.1.3 AudioSessionMode

```
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 AVAudioSessionCategoryRecord. 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_UNAVAILABLE(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);

Enumerator

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 AVAudioSessionCategoryOptionAllowBluetooth and AVAudioSessionCategoryOptionDefaultToSpeaker. API_AVAILABLE(ios(7.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);

3.7 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.

3.8 Photon.Voice.PUN.UtilityScripts Namespace Reference

Classes

- class VoiceDebugScript

  Utility script to be attached next to PhotonVoiceView & PhotonView on the player prefab to be network instantiated. Call voiceDebugScript.CantHearYou() on the networked object of the remote (or local) player if you can't hear the corresponding player.

3.9 Photon.Voice.Unity Namespace Reference

Classes

- class AndroidAudioInAEC
- class AudioClipWrapper
- class AudioOutCapture
- interface ILoggable
- interface ILoggableDependent
- class IOSAudioForceToSpeaker
- class Logger
- class MicWrapper
- class MicWrapperPusher
- struct NativeAndroidMicrophoneSettings
- class PhotonVoiceCreatedParams
- struct PlaybackDelaySettings

Playback delay configuration container.

- class Recorder

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

  Component representing remote audio stream in local scene.
- class UnityAudioOut
- class UnityMicrophone

  A wrapper around UnityEngine.Microphone to be able to safely use Microphone and compile for WebGL.
- class VoiceComponent
- class VoiceConnection

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

3.10 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 SaveIncomingStreamToFile
- class SaveOutgoingStreamToFile
- class TestTone
- class ToneAudioReader

3.11 POpusCodec Namespace Reference

Classes

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

3.12 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

3.12.1 Enumeration Type Documentation

3.12.1.1 Bandwidth

```
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)

3.12.1.2 Channels

```
enum Channels : int [strong]
```

Enumerator

Mono	1 Channel
Stereo	2 Channels

3.12.1.3 Delay

```
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

3.12.1.4 OpusApplicationType

```
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.

3.12.1.5 SignalHint

```
enum SignalHint : int [strong]
```

Enumerator

Auto	(default)
Voice	Bias thresholds towards choosing LPC or Hybrid modes
Music	Bias thresholds towards choosing MDCT modes.

Chapter 4

Class Documentation

4.1 AndroidAudioInAEC Class Reference

Inherits IAudioPusher< short >, and IResettable.

Public Member Functions

- **AndroidAudioInAEC** (Voice.ILogger logger, bool enableAEC=false, bool enableAGC=false, bool enable↩
  NS=false)
- void **SetCallback** (Action< short[ ]> callback, ObjectFactory< short[ ], int > bufferFactory)
- void **Reset** ()
- void **Dispose** ()

Properties

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

4.2 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]`

4.3 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]`

4.4 AudioInChangeNotifier Class Reference

Inherits IDisposable.

Public Member Functions

- **AudioInChangeNotifier** (Action callback, ILogger logger)
- void **Dispose** ()

Public Attributes

- readonly bool **IsSupported** = false

Properties

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

4.5 AudioInEnumerator Class Reference

Enumerates microphones available on device.

Inherits IDisposable.

Public Member Functions

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

Public Attributes

- readonly bool **IsSupported** = false

Properties

- string **Error** `[get]`
- int **Count** `[get]`
- IEnumerable< string > **Names** `[get]`

4.5.1 Detailed Description

Enumerates microphones available on device.

4.6 AudioOutCapture Class Reference

Inherits MonoBehaviour.

Events

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

4.7 AudioSessionParameters Struct Reference

Public Member Functions

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

Public Attributes

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

4.8 AudioSessionParametersPresets Class Reference

Static Public Attributes

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

4.8.1 Member Data Documentation

4.8.1.1 Game

```
AudioSessionParameters Game  [static]
```

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

4.8.1.2 VoIP

```
AudioSessionParameters VoIP  [static]
```

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


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

4.9 AudioSubTypes Class Reference

Defines AudioSubTypes and provides methods to convert between AudioEncoding-values and AudioSubTypes-values.

Static Public Member Functions

- static AudioEncoding EncodingFromSubType (Guid audioSubType)

    Converts a AudioSubTypes-value to a AudioEncoding-value.
- static Guid SubTypeFromEncoding (AudioEncoding audioEncoding)

    Converts a AudioEncoding value to a AudioSubTypes-value.

Static Public Attributes

- static readonly Guid MediaTypeAudio = new Guid("73647561-0000-0010-8000-00AA00389B71")

    The Major Type for `Audio` media types.

- static readonly Guid Unknown = new Guid((short)AudioEncoding.Unknown & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_UNKNOWN, Microsoft Corporation

- static readonly Guid Pcm = new Guid((short)AudioEncoding.Pcm & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_PCM Microsoft Corporation

- static readonly Guid Adpcm = new Guid((short)AudioEncoding.Adpcm & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_ADPCM Microsoft Corporation

- static readonly Guid IeeeFloat = new Guid((short)AudioEncoding.IeeeFloat & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_IEEE_FLOAT Microsoft Corporation

- static readonly Guid Vselp = new Guid((short)AudioEncoding.Vselp & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_VSELP Compaq Computer Corp.

- static readonly Guid IbmCvsd = new Guid((short)AudioEncoding.IbmCvsd & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_IBM_CVSD IBM Corporation

- static readonly Guid ALaw = new Guid((short)AudioEncoding.ALaw & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_ALAW Microsoft Corporation

- static readonly Guid MuLaw = new Guid((short)AudioEncoding.MuLaw & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_MULAW Microsoft Corporation

- static readonly Guid Dts = new Guid((short)AudioEncoding.Dts & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_DTS Microsoft Corporation

- static readonly Guid Drm = new Guid((short)AudioEncoding.Drm & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_DRM Microsoft Corporation

- static readonly Guid WmaVoice9 = new Guid((short)AudioEncoding.WmaVoice9 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_WMAVOICE9

- static readonly Guid OkiAdpcm = new Guid((short)AudioEncoding.OkiAdpcm & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_OKI_ADPCM OKI

- static readonly Guid DviAdpcm = new Guid((short)AudioEncoding.DviAdpcm & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_DVI_ADPCM Intel Corporation

- static readonly Guid ImaAdpcm = new Guid((short)AudioEncoding.ImaAdpcm & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_IMA_ADPCM Intel Corporation

- static readonly Guid MediaspaceAdpcm = new Guid((short)AudioEncoding.MediaspaceAdpcm & 0x0000F↩ FFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_MEDIASPACE_ADPCM Videologic

- static readonly Guid SierraAdpcm = new Guid((short)AudioEncoding.SierraAdpcm & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_SIERRA_ADPCM Sierra Semiconductor Corp

- static readonly Guid G723Adpcm = new Guid((short)AudioEncoding.G723Adpcm & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_G723_ADPCM Antex Electronics Corporation
- static readonly Guid DigiStd = new Guid((short)AudioEncoding.DigiStd & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DIGISTD DSP Solutions, Inc.
- static readonly Guid DigiFix = new Guid((short)AudioEncoding.DigiFix & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DIGIFIX DSP Solutions, Inc.
- static readonly Guid DialogicOkiAdpcm = new Guid((short)AudioEncoding.DialogicOkiAdpcm & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DIALOGIC_OKI_ADPCM Dialogic Corporation
- static readonly Guid MediaVisionAdpcm = new Guid((short)AudioEncoding.MediaVisionAdpcm & 0x0000F↩ FFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_MEDIAVISION_ADPCM Media Vision, Inc.
- static readonly Guid CUCodec = new Guid((short)AudioEncoding.CUCodec & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_CU_CODEC Hewlett-Packard Company
- static readonly Guid YamahaAdpcm = new Guid((short)AudioEncoding.YamahaAdpcm & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_YAMAHA_ADPCM Yamaha Corporation of America
- static readonly Guid SonarC = new Guid((short)AudioEncoding.SonarC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_SONARC Speech Compression
- static readonly Guid DspGroupTrueSpeech = new Guid((short)AudioEncoding.DspGroupTrueSpeech & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DSPGROUP_TRUESPEECH DSP Group, Inc
- static readonly Guid EchoSpeechCorporation1 = new Guid((short)AudioEncoding.EchoSpeechCorporation1 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_ECHOSC1 Echo Speech Corporation
- static readonly Guid AudioFileAf36 = new Guid((short)AudioEncoding.AudioFileAf36 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_AUDIOFILE_AF36, Virtual Music, Inc.
- static readonly Guid Aptx = new Guid((short)AudioEncoding.Aptx & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_APTX Audio Processing Technology
- static readonly Guid AudioFileAf10 = new Guid((short)AudioEncoding.AudioFileAf10 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_AUDIOFILE_AF10, Virtual Music, Inc.
- static readonly Guid Prosody1612 = new Guid((short)AudioEncoding.Prosody1612 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_PROSODY_1612, Aculab plc
- static readonly Guid Lrc = new Guid((short)AudioEncoding.Lrc & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_LRC, Merging Technologies S.A.
- static readonly Guid DolbyAc2 = new Guid((short)AudioEncoding.DolbyAc2 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DOLBY_AC2, Dolby Laboratories
- static readonly Guid Gsm610 = new Guid((short)AudioEncoding.Gsm610 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_GSM610, Microsoft Corporation
- static readonly Guid MsnAudio = new Guid((short)AudioEncoding.MsnAudio & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

WAVE_FORMAT_MSNAUDIO, Microsoft Corporation

- static readonly Guid AntexAdpcme = new Guid((short)AudioEncoding.AntexAdpcme & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_ANTEX_ADPCME, Antex Electronics Corporation
- static readonly Guid ControlResVqlpc = new Guid((short)AudioEncoding.ControlResVqlpc & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_CONTROL_RES_VQLPC, Control Resources Limited
- static readonly Guid DigiReal = new Guid((short)AudioEncoding.DigiReal & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DIGIREAL, DSP Solutions, Inc.
- static readonly Guid DigiAdpcm = new Guid((short)AudioEncoding.DigiAdpcm & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DIGIADPCM, DSP Solutions, Inc.
- static readonly Guid ControlResCr10 = new Guid((short)AudioEncoding.ControlResCr10 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_CONTROL_RES_CR10, Control Resources Limited
- static readonly Guid WAVE_FORMAT_NMS_VBXADPCM = new Guid((short)AudioEncoding.WAVE_FOR↩ MAT_NMS_VBXADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_NMS_VBXADPCM
- static readonly Guid WAVE_FORMAT_CS_IMAADPCM = new Guid((short)AudioEncoding.WAVE_FORM↩ AT_CS_IMAADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_CS_IMAADPCM
- static readonly Guid WAVE_FORMAT_ECHOSC3 = new Guid((short)AudioEncoding.WAVE_FORMAT_E↩ CHOSC3 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_ECHOSC3
- static readonly Guid WAVE_FORMAT_ROCKWELL_ADPCM = new Guid((short)AudioEncoding.WAVE_F↩ ORMAT_ROCKWELL_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_ROCKWELL_ADPCM
- static readonly Guid WAVE_FORMAT_ROCKWELL_DIGITALK = new Guid((short)AudioEncoding.WAVE↩ _FORMAT_ROCKWELL_DIGITALK & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_ROCKWELL_DIGITALK
- static readonly Guid WAVE_FORMAT_XEBEC = new Guid((short)AudioEncoding.WAVE_FORMAT_XEBEC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_XEBEC
- static readonly Guid WAVE_FORMAT_G721_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMA↩ T_G721_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_G721_ADPCM
- static readonly Guid WAVE_FORMAT_G728_CELP = new Guid((short)AudioEncoding.WAVE_FORMAT_↩ G728_CELP & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_G728_CELP
- static readonly Guid WAVE_FORMAT_MSG723 = new Guid((short)AudioEncoding.WAVE_FORMAT_MS↩ G723 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_MSG723
- static readonly Guid Mpeg = new Guid((short)AudioEncoding.Mpeg & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_MPEG, Microsoft Corporation
- static readonly Guid WAVE_FORMAT_RT24 = new Guid((short)AudioEncoding.WAVE_FORMAT_RT24 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_RT24
- static readonly Guid WAVE_FORMAT_PAC = new Guid((short)AudioEncoding.WAVE_FORMAT_PAC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

WAVE_FORMAT_PAC

- static readonly Guid MpegLayer3 = new Guid((short)AudioEncoding.MpegLayer3 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_MPEGLAYER3, ISO/MPEG Layer3 Format Tag
- static readonly Guid WAVE_FORMAT_LUCENT_G723 = new Guid((short)AudioEncoding.WAVE_FORM←AT_LUCENT_G723 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_LUCENT_G723
- static readonly Guid WAVE_FORMAT_CIRRUS = new Guid((short)AudioEncoding.WAVE_FORMAT_CIR←RUS & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_CIRRUS
- static readonly Guid WAVE_FORMAT_ESPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_ESP←CM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_ESPCM
- static readonly Guid WAVE_FORMAT_VOXWARE = new Guid((short)AudioEncoding.WAVE_FORMAT_←VOXWARE & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_VOXWARE
- static readonly Guid WAVE_FORMAT_CANOPUS_ATRAC = new Guid((short)AudioEncoding.WAVE_FO←RMAT_CANOPUS_ATRAC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_CANOPUS_ATRAC
- static readonly Guid WAVE_FORMAT_G726_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMA←T_G726_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_G726_ADPCM
- static readonly Guid WAVE_FORMAT_G722_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMA←T_G722_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_G722_ADPCM
- static readonly Guid WAVE_FORMAT_DSAT_DISPLAY = new Guid((short)AudioEncoding.WAVE_FORM←AT_DSAT_DISPLAY & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_DSAT_DISPLAY
- static readonly Guid WAVE_FORMAT_VOXWARE_BYTE_ALIGNED = new Guid((short)AudioEncoding.←WAVE_FORMAT_VOXWARE_BYTE_ALIGNED & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_VOXWARE_BYTE_ALIGNED
- static readonly Guid WAVE_FORMAT_VOXWARE_AC8 = new Guid((short)AudioEncoding.WAVE_FORM←AT_VOXWARE_AC8 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_VOXWARE_AC8
- static readonly Guid WAVE_FORMAT_VOXWARE_AC10 = new Guid((short)AudioEncoding.WAVE_FOR←MAT_VOXWARE_AC10 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_VOXWARE_AC10
- static readonly Guid WAVE_FORMAT_VOXWARE_AC16 = new Guid((short)AudioEncoding.WAVE_FOR←MAT_VOXWARE_AC16 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_VOXWARE_AC16
- static readonly Guid WAVE_FORMAT_VOXWARE_AC20 = new Guid((short)AudioEncoding.WAVE_FOR←MAT_VOXWARE_AC20 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_VOXWARE_AC20
- static readonly Guid WAVE_FORMAT_VOXWARE_RT24 = new Guid((short)AudioEncoding.WAVE_FOR←MAT_VOXWARE_RT24 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_VOXWARE_RT24
- static readonly Guid WAVE_FORMAT_VOXWARE_RT29 = new Guid((short)AudioEncoding.WAVE_FOR←MAT_VOXWARE_RT29 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

    WAVE_FORMAT_VOXWARE_RT29
- static readonly Guid WAVE_FORMAT_VOXWARE_RT29HW = new Guid((short)AudioEncoding.WAVE_F←ORMAT_VOXWARE_RT29HW & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

WAVE_FORMAT_VOXWARE_RT29HW

- static readonly Guid WAVE_FORMAT_VOXWARE_VR12 = new Guid((short)AudioEncoding.WAVE_FOR←MAT_VOXWARE_VR12 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VOXWARE_VR12
- static readonly Guid WAVE_FORMAT_VOXWARE_VR18 = new Guid((short)AudioEncoding.WAVE_FOR←MAT_VOXWARE_VR18 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VOXWARE_VR18
- static readonly Guid WAVE_FORMAT_VOXWARE_TQ40 = new Guid((short)AudioEncoding.WAVE_FOR←MAT_VOXWARE_TQ40 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VOXWARE_TQ40
- static readonly Guid WAVE_FORMAT_SOFTSOUND = new Guid((short)AudioEncoding.WAVE_FORMAT←_SOFTSOUND & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_SOFTSOUND
- static readonly Guid WAVE_FORMAT_VOXWARE_TQ60 = new Guid((short)AudioEncoding.WAVE_FOR←MAT_VOXWARE_TQ60 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VOXWARE_TQ60
- static readonly Guid WAVE_FORMAT_MSRT24 = new Guid((short)AudioEncoding.WAVE_FORMAT_MS←RT24 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_MSRT24
- static readonly Guid WAVE_FORMAT_G729A = new Guid((short)AudioEncoding.WAVE_FORMAT_G729A & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_G729A
- static readonly Guid WAVE_FORMAT_MVI_MVI2 = new Guid((short)AudioEncoding.WAVE_FORMAT_M←VI_MVI2 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_MVI_MVI2
- static readonly Guid WAVE_FORMAT_DF_G726 = new Guid((short)AudioEncoding.WAVE_FORMAT_DF←_G726 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DF_G726
- static readonly Guid WAVE_FORMAT_DF_GSM610 = new Guid((short)AudioEncoding.WAVE_FORMAT←_DF_GSM610 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DF_GSM610
- static readonly Guid WAVE_FORMAT_ISIAUDIO = new Guid((short)AudioEncoding.WAVE_FORMAT_ISI←AUDIO & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_ISIAUDIO
- static readonly Guid WAVE_FORMAT_ONLIVE = new Guid((short)AudioEncoding.WAVE_FORMAT_ON←LIVE & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_ONLIVE
- static readonly Guid WAVE_FORMAT_SBC24 = new Guid((short)AudioEncoding.WAVE_FORMAT_SBC24 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_SBC24
- static readonly Guid WAVE_FORMAT_DOLBY_AC3_SPDIF = new Guid((short)AudioEncoding.WAVE_F←ORMAT_DOLBY_AC3_SPDIF & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DOLBY_AC3_SPDIF
- static readonly Guid WAVE_FORMAT_MEDIASONIC_G723 = new Guid((short)AudioEncoding.WAVE_F←ORMAT_MEDIASONIC_G723 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_MEDIASONIC_G723
- static readonly Guid WAVE_FORMAT_PROSODY_8KBPS = new Guid((short)AudioEncoding.WAVE_FO←RMAT_PROSODY_8KBPS & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_PROSODY_8KBPS
- static readonly Guid WAVE_FORMAT_ZYXEL_ADPCM = new Guid((short)AudioEncoding.WAVE_FORM←AT_ZYXEL_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

WAVE_FORMAT_ZYXEL_ADPCM

- static readonly Guid WAVE_FORMAT_PHILIPS_LPCBB = new Guid((short)AudioEncoding.WAVE_FOR←MAT_PHILIPS_LPCBB & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_PHILIPS_LPCBB
- static readonly Guid WAVE_FORMAT_PACKED = new Guid((short)AudioEncoding.WAVE_FORMAT_PA←CKED & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_PACKED
- static readonly Guid WAVE_FORMAT_MALDEN_PHONYTALK = new Guid((short)AudioEncoding.WAVE←_FORMAT_MALDEN_PHONYTALK & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_MALDEN_PHONYTALK
- static readonly Guid Gsm = new Guid((short)AudioEncoding.Gsm & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_GSM
- static readonly Guid G729 = new Guid((short)AudioEncoding.G729 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_G729
- static readonly Guid G723 = new Guid((short)AudioEncoding.G723 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_G723
- static readonly Guid Acelp = new Guid((short)AudioEncoding.Acelp & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_ACELP
- static readonly Guid RawAac = new Guid((short)AudioEncoding.RawAac & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_RAW_AAC1
- static readonly Guid WAVE_FORMAT_RHETOREX_ADPCM = new Guid((short)AudioEncoding.WAVE_F←ORMAT_RHETOREX_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_RHETOREX_ADPCM
- static readonly Guid WAVE_FORMAT_IRAT = new Guid((short)AudioEncoding.WAVE_FORMAT_IRAT & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_IRAT
- static readonly Guid WAVE_FORMAT_VIVO_G723 = new Guid((short)AudioEncoding.WAVE_FORMAT_←VIVO_G723 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VIVO_G723
- static readonly Guid WAVE_FORMAT_VIVO_SIREN = new Guid((short)AudioEncoding.WAVE_FORMAT←_VIVO_SIREN & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VIVO_SIREN
- static readonly Guid WAVE_FORMAT_DIGITAL_G723 = new Guid((short)AudioEncoding.WAVE_FORMA←T_DIGITAL_G723 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DIGITAL_G723
- static readonly Guid WAVE_FORMAT_SANYO_LD_ADPCM = new Guid((short)AudioEncoding.WAVE_F←ORMAT_SANYO_LD_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_SANYO_LD_ADPCM
- static readonly Guid WAVE_FORMAT_SIPROLAB_ACEPLNET = new Guid((short)AudioEncoding.WAVE←_FORMAT_SIPROLAB_ACEPLNET & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_SIPROLAB_ACEPLNET
- static readonly Guid WAVE_FORMAT_SIPROLAB_ACELP4800 = new Guid((short)AudioEncoding.WAVE←_FORMAT_SIPROLAB_ACELP4800 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

WAVE_FORMAT_SIPROLAB_ACELP4800

- static readonly Guid WAVE_FORMAT_SIPROLAB_ACELP8V3 = new Guid((short)AudioEncoding.WAVE↩_FORMAT_SIPROLAB_ACELP8V3 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_SIPROLAB_ACELP8V3
- static readonly Guid WAVE_FORMAT_SIPROLAB_G729 = new Guid((short)AudioEncoding.WAVE_FOR↩MAT_SIPROLAB_G729 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_SIPROLAB_G729
- static readonly Guid WAVE_FORMAT_SIPROLAB_G729A = new Guid((short)AudioEncoding.WAVE_FO↩RMAT_SIPROLAB_G729A & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_SIPROLAB_G729A
- static readonly Guid WAVE_FORMAT_SIPROLAB_KELVIN = new Guid((short)AudioEncoding.WAVE_F↩ORMAT_SIPROLAB_KELVIN & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_SIPROLAB_KELVIN
- static readonly Guid WAVE_FORMAT_G726ADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT↩_G726ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_G726ADPCM
- static readonly Guid WAVE_FORMAT_QUALCOMM_PUREVOICE = new Guid((short)AudioEncoding.WA↩VE_FORMAT_QUALCOMM_PUREVOICE & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_QUALCOMM_PUREVOICE
- static readonly Guid WAVE_FORMAT_QUALCOMM_HALFRATE = new Guid((short)AudioEncoding.WA↩VE_FORMAT_QUALCOMM_HALFRATE & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_QUALCOMM_HALFRATE
- static readonly Guid WAVE_FORMAT_TUBGSM = new Guid((short)AudioEncoding.WAVE_FORMAT_TU↩BGSM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_TUBGSM
- static readonly Guid WAVE_FORMAT_MSAUDIO1 = new Guid((short)AudioEncoding.WAVE_FORMAT_↩MSAUDIO1 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_MSAUDIO1
- static readonly Guid WindowsMediaAudio = new Guid((short)AudioEncoding.WindowsMediaAudio & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  Windows Media Audio, WAVE_FORMAT_WMAUDIO2, Microsoft Corporation
- static readonly Guid WindowsMediaAudioProfessional = new Guid((short)AudioEncoding.WindowsMedia↩AudioProfessional & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  Windows Media Audio Professional WAVE_FORMAT_WMAUDIO3, Microsoft Corporation
- static readonly Guid WindowsMediaAudioLosseless = new Guid((short)AudioEncoding.WindowsMedia↩AudioLosseless & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  Windows Media Audio Lossless, WAVE_FORMAT_WMAUDIO_LOSSLESS
- static readonly Guid WindowsMediaAudioSpdif = new Guid((short)AudioEncoding.WindowsMediaAudioSpdif & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  Windows Media Audio Professional over SPDIF WAVE_FORMAT_WMASPDIF (0x0164)
- static readonly Guid WAVE_FORMAT_UNISYS_NAP_ADPCM = new Guid((short)AudioEncoding.WAVE↩_FORMAT_UNISYS_NAP_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_UNISYS_NAP_ADPCM
- static readonly Guid WAVE_FORMAT_UNISYS_NAP_ULAW = new Guid((short)AudioEncoding.WAVE_F↩ORMAT_UNISYS_NAP_ULAW & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_UNISYS_NAP_ULAW

- static readonly Guid WAVE_FORMAT_UNISYS_NAP_ALAW = new Guid((short)AudioEncoding.WAVE_F←ORMAT_UNISYS_NAP_ALAW & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_UNISYS_NAP_ALAW
- static readonly Guid WAVE_FORMAT_UNISYS_NAP_16K = new Guid((short)AudioEncoding.WAVE_FO←RMAT_UNISYS_NAP_16K & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_UNISYS_NAP_16K
- static readonly Guid WAVE_FORMAT_CREATIVE_ADPCM = new Guid((short)AudioEncoding.WAVE_F←ORMAT_CREATIVE_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_CREATIVE_ADPCM
- static readonly Guid WAVE_FORMAT_CREATIVE_FASTSPEECH8 = new Guid((short)AudioEncoding.←WAVE_FORMAT_CREATIVE_FASTSPEECH8 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_CREATIVE_FASTSPEECH8
- static readonly Guid WAVE_FORMAT_CREATIVE_FASTSPEECH10 = new Guid((short)AudioEncoding.←WAVE_FORMAT_CREATIVE_FASTSPEECH10 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_CREATIVE_FASTSPEECH10
- static readonly Guid WAVE_FORMAT_UHER_ADPCM = new Guid((short)AudioEncoding.WAVE_FORM←AT_UHER_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_UHER_ADPCM
- static readonly Guid WAVE_FORMAT_QUARTERDECK = new Guid((short)AudioEncoding.WAVE_FOR←MAT_QUARTERDECK & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_QUARTERDECK
- static readonly Guid WAVE_FORMAT_ILINK_VC = new Guid((short)AudioEncoding.WAVE_FORMAT_ILI←NK_VC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_ILINK_VC
- static readonly Guid WAVE_FORMAT_RAW_SPORT = new Guid((short)AudioEncoding.WAVE_FORMA←T_RAW_SPORT & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_RAW_SPORT
- static readonly Guid WAVE_FORMAT_ESST_AC3 = new Guid((short)AudioEncoding.WAVE_FORMAT_←ESST_AC3 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_ESST_AC3
- static readonly Guid WAVE_FORMAT_IPI_HSX = new Guid((short)AudioEncoding.WAVE_FORMAT_IPI_←HSX & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_IPI_HSX
- static readonly Guid WAVE_FORMAT_IPI_RPELP = new Guid((short)AudioEncoding.WAVE_FORMAT_I←PI_RPELP & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_IPI_RPELP
- static readonly Guid WAVE_FORMAT_CS2 = new Guid((short)AudioEncoding.WAVE_FORMAT_CS2 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_CS2
- static readonly Guid WAVE_FORMAT_SONY_SCX = new Guid((short)AudioEncoding.WAVE_FORMAT_←SONY_SCX & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_SONY_SCX
- static readonly Guid WAVE_FORMAT_FM_TOWNS_SND = new Guid((short)AudioEncoding.WAVE_FOR←MAT_FM_TOWNS_SND & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_FM_TOWNS_SND
- static readonly Guid WAVE_FORMAT_BTV_DIGITAL = new Guid((short)AudioEncoding.WAVE_FORMA←T_BTV_DIGITAL & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_BTV_DIGITAL

- static readonly Guid WAVE_FORMAT_QDESIGN_MUSIC = new Guid((short)AudioEncoding.WAVE_FOR↩ MAT_QDESIGN_MUSIC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_QDESIGN_MUSIC
- static readonly Guid WAVE_FORMAT_VME_VMPCM = new Guid((short)AudioEncoding.WAVE_FORMA↩ T_VME_VMPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VME_VMPCM
- static readonly Guid WAVE_FORMAT_TPC = new Guid((short)AudioEncoding.WAVE_FORMAT_TPC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_TPC
- static readonly Guid WAVE_FORMAT_OLIGSM = new Guid((short)AudioEncoding.WAVE_FORMAT_OLI↩ GSM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_OLIGSM
- static readonly Guid WAVE_FORMAT_OLIADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_↩ OLIADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_OLIADPCM
- static readonly Guid WAVE_FORMAT_OLICELP = new Guid((short)AudioEncoding.WAVE_FORMAT_OL↩ ICELP & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_OLICELP
- static readonly Guid WAVE_FORMAT_OLISBC = new Guid((short)AudioEncoding.WAVE_FORMAT_OLI↩ SBC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_OLISBC
- static readonly Guid WAVE_FORMAT_OLIOPR = new Guid((short)AudioEncoding.WAVE_FORMAT_OLI↩ OPR & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_OLIOPR
- static readonly Guid WAVE_FORMAT_LH_CODEC = new Guid((short)AudioEncoding.WAVE_FORMAT_↩ LH_CODEC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_LH_CODEC
- static readonly Guid WAVE_FORMAT_NORRIS = new Guid((short)AudioEncoding.WAVE_FORMAT_NO↩ RRIS & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_NORRIS
- static readonly Guid WAVE_FORMAT_SOUNDSPACE_MUSICOMPRESS = new Guid((short)Audio↩ Encoding.WAVE_FORMAT_SOUNDSPACE_MUSICOMPRESS & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_SOUNDSPACE_MUSICOMPRESS
- static readonly Guid MPEG_ADTS_AAC = new Guid((short)AudioEncoding.MPEG_ADTS_AAC & 0x0000↩ FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  Advanced Audio Coding (AAC) audio in Audio Data Transport Stream (ADTS) format. The format block is a WAVE↩ FORMATEX structure with wFormatTag equal to WAVE_FORMAT_MPEG_ADTS_AAC.
- static readonly Guid MPEG_RAW_AAC = new Guid((short)AudioEncoding.MPEG_RAW_AAC & 0x0000F↩ FFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  MPEG_RAW_AAC
- static readonly Guid MPEG_LOAS = new Guid((short)AudioEncoding.MPEG_LOAS & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  MPEG-4 audio transport stream with a synchronization layer (LOAS) and a multiplex layer (LATM). The format block is a WAVEFORMATEX structure with wFormatTag equal to WAVE_FORMAT_MPEG_LOAS. See .
- static readonly Guid NOKIA_MPEG_ADTS_AAC = new Guid((short)AudioEncoding.NOKIA_MPEG_ADTS↩ _AAC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  NOKIA_MPEG_ADTS_AAC
- static readonly Guid NOKIA_MPEG_RAW_AAC = new Guid((short)AudioEncoding.NOKIA_MPEG_RAW_↩ AAC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  NOKIA_MPEG_RAW_AAC
- static readonly Guid VODAFONE_MPEG_ADTS_AAC = new Guid((short)AudioEncoding.VODAFONE_M↩ PEG_ADTS_AAC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

VODAFONE_MPEG_ADTS_AAC

- static readonly Guid VODAFONE_MPEG_RAW_AAC = new Guid((short)AudioEncoding.VODAFONE_MP←EG_RAW_AAC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  VODAFONE_MPEG_RAW_AAC
- static readonly Guid MPEG_HEAAC = new Guid((short)AudioEncoding.MPEG_HEAAC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  High-Efficiency Advanced Audio Coding (HE-AAC) stream. The format block is an HEAACWAVEFORMAT structure. See .
- static readonly Guid WAVE_FORMAT_DVM = new Guid((short)AudioEncoding.WAVE_FORMAT_DVM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DVM
- static readonly Guid Vorbis1 = new Guid((short)AudioEncoding.Vorbis1 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VORBIS1 "Og" Original stream compatible
- static readonly Guid Vorbis2 = new Guid((short)AudioEncoding.Vorbis2 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VORBIS2 "Pg" Have independent header
- static readonly Guid Vorbis3 = new Guid((short)AudioEncoding.Vorbis3 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VORBIS3 "Qg" Have no codebook header
- static readonly Guid Vorbis1P = new Guid((short)AudioEncoding.Vorbis1P & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VORBIS1P "og" Original stream compatible
- static readonly Guid Vorbis2P = new Guid((short)AudioEncoding.Vorbis2P & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VORBIS2P "pg" Have independent headere
- static readonly Guid Vorbis3P = new Guid((short)AudioEncoding.Vorbis3P & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_VORBIS3P "qg" Have no codebook header
- static readonly Guid WAVE_FORMAT_RAW_AAC1 = new Guid((short)AudioEncoding.WAVE_FORMAT_←RAW_AAC1 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  Raw AAC1
- static readonly Guid WAVE_FORMAT_WMAVOICE9 = new Guid((short)AudioEncoding.WAVE_FORMAT←_WMAVOICE9 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  Windows Media Audio Voice (WMA Voice)
- static readonly Guid Extensible = new Guid((short)AudioEncoding.Extensible & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  Extensible
- static readonly Guid WAVE_FORMAT_DEVELOPMENT = new Guid((short)AudioEncoding.WAVE_FORM←AT_DEVELOPMENT & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  WAVE_FORMAT_DEVELOPMENT
- static readonly Guid WAVE_FORMAT_FLAC = new Guid((short)AudioEncoding.WAVE_FORMAT_FLAC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)

  FLAC

4.9.1 Detailed Description

Defines AudioSubTypes and provides methods to convert between AudioEncoding-values and AudioSubTypes-values.

AudioSubTypes are used by the WaveFormatExtensible, the MFMediaType and the MediaType class.

4.9.2 Member Function Documentation

4.9.2.1 EncodingFromSubType()

```
static AudioEncoding EncodingFromSubType (
             Guid audioSubType )  [static]
```

Converts a AudioSubTypes-value to a AudioEncoding-value.

Parameters

audioSubType	The AudioSubTypes-value to convert to the equivalent AudioEncoding-value.

Returns

The AudioEncoding which belongs to the specified *audioSubType* .

4.9.2.2 SubTypeFromEncoding()

```
static Guid SubTypeFromEncoding (
             AudioEncoding audioEncoding )  [static]
```

Converts a AudioEncoding value to a AudioSubTypes-value.

Parameters

audioEncoding	The AudioEncoding to convert to the equivalent AudioSubTypes-value.

Returns

The AudioSubTypes-value which belongs to the specified *audioEncoding* .

4.9.3 Member Data Documentation

4.9.3.1 Acelp

```
readonly Guid Acelp = new Guid((short)AudioEncoding.Acelp & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_ACELP

4.9.3.2 Adpcm

```
readonly Guid Adpcm = new Guid((short)AudioEncoding.Adpcm & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_ADPCM Microsoft Corporation

4.9.3.3 ALaw

```
readonly Guid ALaw = new Guid((short)AudioEncoding.ALaw & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_ALAW Microsoft Corporation

4.9.3.4 AntexAdpcme

```
readonly Guid AntexAdpcme = new Guid((short)AudioEncoding.AntexAdpcme & 0x0000FFFF, 0x0000,
0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_ANTEX_ADPCME, Antex Electronics Corporation

4.9.3.5 Aptx

```
readonly Guid Aptx = new Guid((short)AudioEncoding.Aptx & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_APTX Audio Processing Technology

4.9.3.6 AudioFileAf10

```
readonly Guid AudioFileAf10 = new Guid((short)AudioEncoding.AudioFileAf10 & 0x0000FFFF, 0x0000,
0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_AUDIOFILE_AF10, Virtual Music, Inc.

4.9.3.7 AudioFileAf36

```
readonly Guid AudioFileAf36 = new Guid((short)AudioEncoding.AudioFileAf36 & 0x0000FFFF, 0x0000,
0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_AUDIOFILE_AF36, Virtual Music, Inc.

4.9.3.8 ControlResCr10

```
readonly Guid ControlResCr10 = new Guid((short)AudioEncoding.ControlResCr10 & 0x0000FFFF,
0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_CONTROL_RES_CR10, Control Resources Limited

4.9.3.9 ControlResVqlpc

```
readonly Guid ControlResVqlpc = new Guid((short)AudioEncoding.ControlResVqlpc & 0x0000FFFF,
0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_CONTROL_RES_VQLPC, Control Resources Limited

4.9.3.10 CUCodec

```
readonly Guid CUCodec = new Guid((short)AudioEncoding.CUCodec & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_CU_CODEC Hewlett-Packard Company

4.9.3.11 DialogicOkiAdpcm

```
readonly Guid DialogicOkiAdpcm = new Guid((short)AudioEncoding.DialogicOkiAdpcm & 0x0000FFFF,
0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_DIALOGIC_OKI_ADPCM Dialogic Corporation

4.9.3.12 DigiAdpcm

```
readonly Guid DigiAdpcm = new Guid((short)AudioEncoding.DigiAdpcm & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_DIGIADPCM, DSP Solutions, Inc.

4.9.3.13 DigiFix

```
readonly Guid DigiFix = new Guid((short)AudioEncoding.DigiFix & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_DIGIFIX DSP Solutions, Inc.

4.9.3.14 DigiReal

```
readonly Guid DigiReal = new Guid((short)AudioEncoding.DigiReal & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_DIGIREAL, DSP Solutions, Inc.

4.9.3.15 DigiStd

```
readonly Guid DigiStd = new Guid((short)AudioEncoding.DigiStd & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_DIGISTD DSP Solutions, Inc.

4.9.3.16 DolbyAc2

```
readonly Guid DolbyAc2 = new Guid((short)AudioEncoding.DolbyAc2 & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_DOLBY_AC2, Dolby Laboratories

4.9.3.17 Drm

```
readonly Guid Drm = new Guid((short)AudioEncoding.Drm & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_DRM Microsoft Corporation

4.9.3.18 DspGroupTrueSpeech

```
readonly Guid DspGroupTrueSpeech = new Guid((short)AudioEncoding.DspGroupTrueSpeech & 0x0000↩
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_DSPGROUP_TRUESPEECH DSP Group, Inc

4.9.3.19 Dts

```
readonly Guid Dts = new Guid((short)AudioEncoding.Dts & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_DTS Microsoft Corporation

4.9.3.20 DviAdpcm

```
readonly Guid DviAdpcm = new Guid((short)AudioEncoding.DviAdpcm & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_DVI_ADPCM Intel Corporation

4.9.3.21 EchoSpeechCorporation1

```
readonly Guid EchoSpeechCorporation1 = new Guid((short)AudioEncoding.EchoSpeechCorporation1 &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_ECHOSC1 Echo Speech Corporation

4.9.3.22 Extensible

```
readonly Guid Extensible = new Guid((short)AudioEncoding.Extensible & 0x0000FFFF, 0x0000,
0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

Extensible

4.9.3.23 G723

```
readonly Guid G723 = new Guid((short)AudioEncoding.G723 & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_G723

4.9.3.24 G723Adpcm

```
readonly Guid G723Adpcm = new Guid((short)AudioEncoding.G723Adpcm & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_G723_ADPCM Antex Electronics Corporation

4.9.3.25 G729

```
readonly Guid G729 = new Guid((short)AudioEncoding.G729 & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_G729

4.9.3.26 Gsm

```
readonly Guid Gsm = new Guid((short)AudioEncoding.Gsm & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_GSM

4.9.3.27 Gsm610

```
readonly Guid Gsm610 = new Guid((short)AudioEncoding.Gsm610 & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_GSM610, Microsoft Corporation

4.9.3.28 IbmCvsd

```
readonly Guid IbmCvsd = new Guid((short)AudioEncoding.IbmCvsd & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_IBM_CVSD IBM Corporation

4.9.3.29 IeeeFloat

```
readonly Guid IeeeFloat = new Guid((short)AudioEncoding.IeeeFloat & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_IEEE_FLOAT Microsoft Corporation

4.9.3.30 ImaAdpcm

```
readonly Guid ImaAdpcm = new Guid((short)AudioEncoding.ImaAdpcm & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_IMA_ADPCM Intel Corporation

4.9.3.31 Lrc

```
readonly Guid Lrc = new Guid((short)AudioEncoding.Lrc & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_LRC, Merging Technologies S.A.

4.9.3.32 MediaspaceAdpcm

```
readonly Guid MediaspaceAdpcm = new Guid((short)AudioEncoding.MediaspaceAdpcm & 0x0000FFFF,
0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_MEDIASPACE_ADPCM Videologic

4.9.3.33 MediaTypeAudio

```
readonly Guid MediaTypeAudio = new Guid("73647561-0000-0010-8000-00AA00389B71") [static]
```

The Major Type for `Audio` media types.

4.9.3.34 MediaVisionAdpcm

```
readonly Guid MediaVisionAdpcm = new Guid((short)AudioEncoding.MediaVisionAdpcm & 0x0000FFFF,
0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_MEDIAVISION_ADPCM Media Vision, Inc.

4.9.3.35 Mpeg

```
readonly Guid Mpeg = new Guid((short)AudioEncoding.Mpeg & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_MPEG, Microsoft Corporation

4.9.3.36 MPEG_ADTS_AAC

```
readonly Guid MPEG_ADTS_AAC = new Guid((short)AudioEncoding.MPEG_ADTS_AAC & 0x0000FFFF, 0x0000,
0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

Advanced Audio Coding (AAC) audio in Audio Data Transport Stream (ADTS) format. The format block is a WAV↩
EFORMATEX structure with wFormatTag equal to WAVE_FORMAT_MPEG_ADTS_AAC.

The WAVEFORMATEX structure specifies the core AAC-LC sample rate and number of channels, prior to applying spectral band replication (SBR) or parametric stereo (PS) tools, if present. No additional data is required after the WAVEFORMATEX structure.

http://msdn.microsoft.com/en-us/library/dd317599%28VS.85%29.aspx

4.9.3.37 MPEG_HEAAC

```
readonly Guid MPEG_HEAAC = new Guid((short)AudioEncoding.MPEG_HEAAC & 0x0000FFFF, 0x0000,
0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

High-Efficiency Advanced Audio Coding (HE-AAC) stream. The format block is an HEAACWAVEFORMAT structure. See .

4.9.3.38 MPEG_LOAS

```
readonly Guid MPEG_LOAS = new Guid((short)AudioEncoding.MPEG_LOAS & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

MPEG-4 audio transport stream with a synchronization layer (LOAS) and a multiplex layer (LATM). The format block is a WAVEFORMATEX structure with wFormatTag equal to WAVE_FORMAT_MPEG_LOAS. See .

The WAVEFORMATEX structure specifies the core AAC-LC sample rate and number of channels, prior to applying spectral SBR or PS tools, if present. No additional data is required after the WAVEFORMATEX structure.

4.9.3.39 MPEG_RAW_AAC

```
readonly Guid MPEG_RAW_AAC = new Guid((short)AudioEncoding.MPEG_RAW_AAC & 0x0000FFFF, 0x0000,
0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

MPEG_RAW_AAC

Source wmCodec.h

4.9.3.40 MpegLayer3

```
readonly Guid MpegLayer3 = new Guid((short)AudioEncoding.MpegLayer3 & 0x0000FFFF, 0x0000,
0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_MPEGLAYER3, ISO/MPEG Layer3 Format Tag

4.9.3.41 MsnAudio

```
readonly Guid MsnAudio = new Guid((short)AudioEncoding.MsnAudio & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_MSNAUDIO, Microsoft Corporation

4.9.3.42 MuLaw

```
readonly Guid MuLaw = new Guid((short)AudioEncoding.MuLaw & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_MULAW Microsoft Corporation

4.9.3.43 NOKIA_MPEG_ADTS_AAC

```
readonly Guid NOKIA_MPEG_ADTS_AAC = new Guid((short)AudioEncoding.NOKIA_MPEG_ADTS_AAC & 0x0000←
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

NOKIA_MPEG_ADTS_AAC

Source wmCodec.h

4.9.3.44 NOKIA_MPEG_RAW_AAC

```
readonly Guid NOKIA_MPEG_RAW_AAC = new Guid((short)AudioEncoding.NOKIA_MPEG_RAW_AAC & 0x0000←
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

NOKIA_MPEG_RAW_AAC

Source wmCodec.h

4.9.3.45 OkiAdpcm

```
readonly Guid OkiAdpcm = new Guid((short)AudioEncoding.OkiAdpcm & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_OKI_ADPCM OKI

4.9.3.46 Pcm

```
readonly Guid Pcm = new Guid((short)AudioEncoding.Pcm & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_PCM Microsoft Corporation

4.9.3.47 Prosody1612

```
readonly Guid Prosody1612 = new Guid((short)AudioEncoding.Prosody1612 & 0x0000FFFF, 0x0000,
0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_PROSODY_1612, Aculab plc

4.9.3.48 RawAac

```
readonly Guid RawAac = new Guid((short)AudioEncoding.RawAac & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_RAW_AAC1

4.9.3.49 SierraAdpcm

```
readonly Guid SierraAdpcm = new Guid((short)AudioEncoding.SierraAdpcm & 0x0000FFFF, 0x0000,
0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_SIERRA_ADPCM Sierra Semiconductor Corp

4.9.3.50 SonarC

```
readonly Guid SonarC = new Guid((short)AudioEncoding.SonarC & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_SONARC Speech Compression

4.9.3.51 Unknown

```
readonly Guid Unknown = new Guid((short)AudioEncoding.Unknown & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_UNKNOWN, Microsoft Corporation

4.9.3.52 VODAFONE_MPEG_ADTS_AAC

```
readonly Guid VODAFONE_MPEG_ADTS_AAC = new Guid((short)AudioEncoding.VODAFONE_MPEG_ADTS_AAC &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

VODAFONE_MPEG_ADTS_AAC

Source wmCodec.h

4.9.3.53 VODAFONE_MPEG_RAW_AAC

```
readonly Guid VODAFONE_MPEG_RAW_AAC = new Guid((short)AudioEncoding.VODAFONE_MPEG_RAW_AAC &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

VODAFONE_MPEG_RAW_AAC

Source wmCodec.h

4.9.3.54 Vorbis1

```
readonly Guid Vorbis1 = new Guid((short)AudioEncoding.Vorbis1 & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VORBIS1 "Og" Original stream compatible

4.9.3.55 Vorbis1P

```
readonly Guid Vorbis1P = new Guid((short)AudioEncoding.Vorbis1P & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VORBIS1P "og" Original stream compatible

4.9.3.56 Vorbis2

```
readonly Guid Vorbis2 = new Guid((short)AudioEncoding.Vorbis2 & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VORBIS2 "Pg" Have independent header

4.9.3.57 Vorbis2P

```
readonly Guid Vorbis2P = new Guid((short)AudioEncoding.Vorbis2P & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VORBIS2P "pg" Have independent headere

4.9.3.58 Vorbis3

```
readonly Guid Vorbis3 = new Guid((short)AudioEncoding.Vorbis3 & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VORBIS3 "Qg" Have no codebook header

4.9.3.59 Vorbis3P

```
readonly Guid Vorbis3P = new Guid((short)AudioEncoding.Vorbis3P & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VORBIS3P "qg" Have no codebook header

4.9.3.60 Vselp

```
readonly Guid Vselp = new Guid((short)AudioEncoding.Vselp & 0x0000FFFF, 0x0000, 0x0010, 0x80,
0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VSELP Compaq Computer Corp.

4.9.3.61 WAVE_FORMAT_BTV_DIGITAL

```
readonly Guid WAVE_FORMAT_BTV_DIGITAL = new Guid((short)AudioEncoding.WAVE_FORMAT_BTV_DIGITAL
& 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_BTV_DIGITAL

4.9.3.62 WAVE_FORMAT_CANOPUS_ATRAC

```
readonly Guid WAVE_FORMAT_CANOPUS_ATRAC = new Guid((short)AudioEncoding.WAVE_FORMAT_CANOPUS_↩
ATRAC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_CANOPUS_ATRAC

4.9.3.63 WAVE_FORMAT_CIRRUS

```
readonly Guid WAVE_FORMAT_CIRRUS = new Guid((short)AudioEncoding.WAVE_FORMAT_CIRRUS & 0x0000↩
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_CIRRUS

4.9.3.64 WAVE_FORMAT_CREATIVE_ADPCM

```
readonly Guid WAVE_FORMAT_CREATIVE_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_CREATIV↩
E_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_CREATIVE_ADPCM

4.9.3.65 WAVE_FORMAT_CREATIVE_FASTSPEECH10

```
readonly Guid WAVE_FORMAT_CREATIVE_FASTSPEECH10 = new Guid((short)AudioEncoding.WAVE_FORMAT_↩
CREATIVE_FASTSPEECH10 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b,
0x71) [static]
```

WAVE_FORMAT_CREATIVE_FASTSPEECH10

4.9.3.66 WAVE_FORMAT_CREATIVE_FASTSPEECH8

```
readonly Guid WAVE_FORMAT_CREATIVE_FASTSPEECH8 = new Guid((short)AudioEncoding.WAVE_FORMAT_↩
CREATIVE_FASTSPEECH8 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b,
0x71) [static]
```

WAVE_FORMAT_CREATIVE_FASTSPEECH8

4.9.3.67 WAVE_FORMAT_CS2

```
readonly Guid WAVE_FORMAT_CS2 = new Guid((short)AudioEncoding.WAVE_FORMAT_CS2 & 0x0000FFFF,
0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_CS2

4.9.3.68 WAVE_FORMAT_CS_IMAADPCM

```
readonly Guid WAVE_FORMAT_CS_IMAADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_CS_IMAADPCM
& 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_CS_IMAADPCM

4.9.3.69 WAVE_FORMAT_DEVELOPMENT

```
readonly Guid WAVE_FORMAT_DEVELOPMENT = new Guid((short)AudioEncoding.WAVE_FORMAT_DEVELOPMENT
& 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_DEVELOPMENT

4.9.3.70 WAVE_FORMAT_DF_G726

```
readonly Guid WAVE_FORMAT_DF_G726 = new Guid((short)AudioEncoding.WAVE_FORMAT_DF_G726 & 0x0000←
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_DF_G726

4.9.3.71 WAVE_FORMAT_DF_GSM610

```
readonly Guid WAVE_FORMAT_DF_GSM610 = new Guid((short)AudioEncoding.WAVE_FORMAT_DF_GSM610 &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_DF_GSM610

4.9.3.72 WAVE_FORMAT_DIGITAL_G723

```
readonly Guid WAVE_FORMAT_DIGITAL_G723 = new Guid((short)AudioEncoding.WAVE_FORMAT_DIGITAL_←
G723 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_DIGITAL_G723

4.9.3.73 WAVE_FORMAT_DOLBY_AC3_SPDIF

```
readonly Guid WAVE_FORMAT_DOLBY_AC3_SPDIF = new Guid((short)AudioEncoding.WAVE_FORMAT_DOL←
BY_AC3_SPDIF & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_DOLBY_AC3_SPDIF

4.9.3.74 WAVE_FORMAT_DSAT_DISPLAY

```
readonly Guid WAVE_FORMAT_DSAT_DISPLAY = new Guid((short)AudioEncoding.WAVE_FORMAT_DSAT_DISP←
LAY & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_DSAT_DISPLAY

4.9.3.75 WAVE_FORMAT_DVM

```
readonly Guid WAVE_FORMAT_DVM = new Guid((short)AudioEncoding.WAVE_FORMAT_DVM & 0x0000FFFF,
0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_DVM

4.9.3.76 WAVE_FORMAT_ECHOSC3

```
readonly Guid WAVE_FORMAT_ECHOSC3 = new Guid((short)AudioEncoding.WAVE_FORMAT_ECHOSC3 & 0x0000↩
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_ECHOSC3

4.9.3.77 WAVE_FORMAT_ESPCM

```
readonly Guid WAVE_FORMAT_ESPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_ESPCM & 0x0000FF↩
FF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_ESPCM

4.9.3.78 WAVE_FORMAT_ESST_AC3

```
readonly Guid WAVE_FORMAT_ESST_AC3 = new Guid((short)AudioEncoding.WAVE_FORMAT_ESST_AC3 &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_ESST_AC3

4.9.3.79 WAVE_FORMAT_FLAC

```
readonly Guid WAVE_FORMAT_FLAC = new Guid((short)AudioEncoding.WAVE_FORMAT_FLAC & 0x0000FFFF,
0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

FLAC

4.9.3.80 WAVE_FORMAT_FM_TOWNS_SND

```
readonly Guid WAVE_FORMAT_FM_TOWNS_SND = new Guid((short)AudioEncoding.WAVE_FORMAT_FM_TOWNS_↩
SND & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_FM_TOWNS_SND

4.9.3.81 WAVE_FORMAT_G721_ADPCM

```
readonly Guid WAVE_FORMAT_G721_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_G721_ADPCM &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_G721_ADPCM

4.9.3.82 WAVE_FORMAT_G722_ADPCM

```
readonly Guid WAVE_FORMAT_G722_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_G722_ADPCM &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_G722_ADPCM

4.9.3.83 WAVE_FORMAT_G726_ADPCM

```
readonly Guid WAVE_FORMAT_G726_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_G726_ADPCM &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_G726_ADPCM

4.9.3.84 WAVE_FORMAT_G726ADPCM

```
readonly Guid WAVE_FORMAT_G726ADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_G726ADPCM &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_G726ADPCM

4.9.3.85 WAVE_FORMAT_G728_CELP

```
readonly Guid WAVE_FORMAT_G728_CELP = new Guid((short)AudioEncoding.WAVE_FORMAT_G728_CELP &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_G728_CELP

4.9.3.86 WAVE_FORMAT_G729A

```
readonly Guid WAVE_FORMAT_G729A = new Guid((short)AudioEncoding.WAVE_FORMAT_G729A & 0x0000FF↵
FF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_G729A

4.9.3.87 WAVE_FORMAT_ILINK_VC

```
readonly Guid WAVE_FORMAT_ILINK_VC = new Guid((short)AudioEncoding.WAVE_FORMAT_ILINK_VC &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_ILINK_VC

4.9.3.88 WAVE_FORMAT_IPI_HSX

```
readonly Guid WAVE_FORMAT_IPI_HSX = new Guid((short)AudioEncoding.WAVE_FORMAT_IPI_HSX & 0x0000←
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_IPI_HSX

4.9.3.89 WAVE_FORMAT_IPI_RPELP

```
readonly Guid WAVE_FORMAT_IPI_RPELP = new Guid((short)AudioEncoding.WAVE_FORMAT_IPI_RPELP &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_IPI_RPELP

4.9.3.90 WAVE_FORMAT_IRAT

```
readonly Guid WAVE_FORMAT_IRAT = new Guid((short)AudioEncoding.WAVE_FORMAT_IRAT & 0x0000FFFF,
0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_IRAT

4.9.3.91 WAVE_FORMAT_ISIAUDIO

```
readonly Guid WAVE_FORMAT_ISIAUDIO = new Guid((short)AudioEncoding.WAVE_FORMAT_ISIAUDIO &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_ISIAUDIO

4.9.3.92 WAVE_FORMAT_LH_CODEC

```
readonly Guid WAVE_FORMAT_LH_CODEC = new Guid((short)AudioEncoding.WAVE_FORMAT_LH_CODEC &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_LH_CODEC

4.9.3.93 WAVE_FORMAT_LUCENT_G723

```
readonly Guid WAVE_FORMAT_LUCENT_G723 = new Guid((short)AudioEncoding.WAVE_FORMAT_LUCENT_G723
& 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_LUCENT_G723

4.9.3.94 WAVE_FORMAT_MALDEN_PHONYTALK

```
readonly Guid WAVE_FORMAT_MALDEN_PHONYTALK = new Guid((short)AudioEncoding.WAVE_FORMAT_MAL↩
DEN_PHONYTALK & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_MALDEN_PHONYTALK

4.9.3.95 WAVE_FORMAT_MEDIASONIC_G723

```
readonly Guid WAVE_FORMAT_MEDIASONIC_G723 = new Guid((short)AudioEncoding.WAVE_FORMAT_MED↩
IASONIC_G723 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_MEDIASONIC_G723

4.9.3.96 WAVE_FORMAT_MSAUDIO1

```
readonly Guid WAVE_FORMAT_MSAUDIO1 = new Guid((short)AudioEncoding.WAVE_FORMAT_MSAUDIO1 &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_MSAUDIO1

4.9.3.97 WAVE_FORMAT_MSG723

```
readonly Guid WAVE_FORMAT_MSG723 = new Guid((short)AudioEncoding.WAVE_FORMAT_MSG723 & 0x0000↩
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_MSG723

4.9.3.98 WAVE_FORMAT_MSRT24

```
readonly Guid WAVE_FORMAT_MSRT24 = new Guid((short)AudioEncoding.WAVE_FORMAT_MSRT24 & 0x0000←
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_MSRT24

4.9.3.99 WAVE_FORMAT_MVI_MVI2

```
readonly Guid WAVE_FORMAT_MVI_MVI2 = new Guid((short)AudioEncoding.WAVE_FORMAT_MVI_MVI2 &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_MVI_MVI2

4.9.3.100 WAVE_FORMAT_NMS_VBXADPCM

```
readonly Guid WAVE_FORMAT_NMS_VBXADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_NMS_VBXAD←
PCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_NMS_VBXADPCM

4.9.3.101 WAVE_FORMAT_NORRIS

```
readonly Guid WAVE_FORMAT_NORRIS = new Guid((short)AudioEncoding.WAVE_FORMAT_NORRIS & 0x0000←
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_NORRIS

4.9.3.102 WAVE_FORMAT_OLIADPCM

```
readonly Guid WAVE_FORMAT_OLIADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_OLIADPCM &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_OLIADPCM

4.9.3.103 WAVE_FORMAT_OLICELP

```
readonly Guid WAVE_FORMAT_OLICELP = new Guid((short)AudioEncoding.WAVE_FORMAT_OLICELP & 0x0000←
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_OLICELP

4.9.3.104 WAVE_FORMAT_OLIGSM

```
readonly Guid WAVE_FORMAT_OLIGSM = new Guid((short)AudioEncoding.WAVE_FORMAT_OLIGSM & 0x0000←
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_OLIGSM

4.9.3.105 WAVE_FORMAT_OLIOPR

```
readonly Guid WAVE_FORMAT_OLIOPR = new Guid((short)AudioEncoding.WAVE_FORMAT_OLIOPR & 0x0000←
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_OLIOPR

4.9.3.106 WAVE_FORMAT_OLISBC

```
readonly Guid WAVE_FORMAT_OLISBC = new Guid((short)AudioEncoding.WAVE_FORMAT_OLISBC & 0x0000←
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_OLISBC

4.9.3.107 WAVE_FORMAT_ONLIVE

```
readonly Guid WAVE_FORMAT_ONLIVE = new Guid((short)AudioEncoding.WAVE_FORMAT_ONLIVE & 0x0000←
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_ONLIVE

4.9.3.108 WAVE_FORMAT_PAC

```
readonly Guid WAVE_FORMAT_PAC = new Guid((short)AudioEncoding.WAVE_FORMAT_PAC & 0x0000FFFF,
0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_PAC

4.9.3.109 WAVE_FORMAT_PACKED

```
readonly Guid WAVE_FORMAT_PACKED = new Guid((short)AudioEncoding.WAVE_FORMAT_PACKED & 0x0000←
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_PACKED

4.9.3.110 WAVE_FORMAT_PHILIPS_LPCBB

```
readonly Guid WAVE_FORMAT_PHILIPS_LPCBB = new Guid((short)AudioEncoding.WAVE_FORMAT_PHILIPS_↩
LPCBB & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_PHILIPS_LPCBB

4.9.3.111 WAVE_FORMAT_PROSODY_8KBPS

```
readonly Guid WAVE_FORMAT_PROSODY_8KBPS = new Guid((short)AudioEncoding.WAVE_FORMAT_PROSODY_↩
8KBPS & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_PROSODY_8KBPS

4.9.3.112 WAVE_FORMAT_QDESIGN_MUSIC

```
readonly Guid WAVE_FORMAT_QDESIGN_MUSIC = new Guid((short)AudioEncoding.WAVE_FORMAT_QDESIGN_↩
MUSIC & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_QDESIGN_MUSIC

4.9.3.113 WAVE_FORMAT_QUALCOMM_HALFRATE

```
readonly Guid WAVE_FORMAT_QUALCOMM_HALFRATE = new Guid((short)AudioEncoding.WAVE_FORMAT_QUA↩
LCOMM_HALFRATE & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_QUALCOMM_HALFRATE

4.9.3.114 WAVE_FORMAT_QUALCOMM_PUREVOICE

```
readonly Guid WAVE_FORMAT_QUALCOMM_PUREVOICE = new Guid((short)AudioEncoding.WAVE_FORMAT_QUA↩
LCOMM_PUREVOICE & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_QUALCOMM_PUREVOICE

4.9.3.115 WAVE_FORMAT_QUARTERDECK

```
readonly Guid WAVE_FORMAT_QUARTERDECK = new Guid((short)AudioEncoding.WAVE_FORMAT_QUARTERDECK
& 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_QUARTERDECK

4.9.3.116 WAVE_FORMAT_RAW_AAC1

```
readonly Guid WAVE_FORMAT_RAW_AAC1 = new Guid((short)AudioEncoding.WAVE_FORMAT_RAW_AAC1 &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

Raw AAC1

4.9.3.117 WAVE_FORMAT_RAW_SPORT

```
readonly Guid WAVE_FORMAT_RAW_SPORT = new Guid((short)AudioEncoding.WAVE_FORMAT_RAW_SPORT &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_RAW_SPORT

4.9.3.118 WAVE_FORMAT_RHETOREX_ADPCM

```
readonly Guid WAVE_FORMAT_RHETOREX_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_RHETORE↩
X_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_RHETOREX_ADPCM

4.9.3.119 WAVE_FORMAT_ROCKWELL_ADPCM

```
readonly Guid WAVE_FORMAT_ROCKWELL_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_ROCKWEL↩
L_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_ROCKWELL_ADPCM

4.9.3.120 WAVE_FORMAT_ROCKWELL_DIGITALK

```
readonly Guid WAVE_FORMAT_ROCKWELL_DIGITALK = new Guid((short)AudioEncoding.WAVE_FORMAT_ROC↩
KWELL_DIGITALK & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_ROCKWELL_DIGITALK

4.9.3.121 WAVE_FORMAT_RT24

```
readonly Guid WAVE_FORMAT_RT24 = new Guid((short)AudioEncoding.WAVE_FORMAT_RT24 & 0x0000FFFF,
0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_RT24

4.9.3.122 WAVE_FORMAT_SANYO_LD_ADPCM

```
readonly Guid WAVE_FORMAT_SANYO_LD_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_SANYO_L↩
D_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_SANYO_LD_ADPCM

4.9.3.123 WAVE_FORMAT_SBC24

```
readonly Guid WAVE_FORMAT_SBC24 = new Guid((short)AudioEncoding.WAVE_FORMAT_SBC24 & 0x0000FF↩
FF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_SBC24

4.9.3.124 WAVE_FORMAT_SIPROLAB_ACELP4800

```
readonly Guid WAVE_FORMAT_SIPROLAB_ACELP4800 = new Guid((short)AudioEncoding.WAVE_FORMAT_SIP↩
ROLAB_ACELP4800 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_SIPROLAB_ACELP4800

4.9.3.125 WAVE_FORMAT_SIPROLAB_ACELP8V3

```
readonly Guid WAVE_FORMAT_SIPROLAB_ACELP8V3 = new Guid((short)AudioEncoding.WAVE_FORMAT_SIP↩
ROLAB_ACELP8V3 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_SIPROLAB_ACELP8V3

4.9.3.126 WAVE_FORMAT_SIPROLAB_ACEPLNET

```
readonly Guid WAVE_FORMAT_SIPROLAB_ACEPLNET = new Guid((short)AudioEncoding.WAVE_FORMAT_SIP↩
ROLAB_ACEPLNET & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_SIPROLAB_ACEPLNET

4.9.3.127 WAVE_FORMAT_SIPROLAB_G729

```
readonly Guid WAVE_FORMAT_SIPROLAB_G729 = new Guid((short)AudioEncoding.WAVE_FORMAT_SIPROLAB↩
_G729 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_SIPROLAB_G729

4.9.3.128 WAVE_FORMAT_SIPROLAB_G729A

```
readonly Guid WAVE_FORMAT_SIPROLAB_G729A = new Guid((short)AudioEncoding.WAVE_FORMAT_SIPROLA↩
B_G729A & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_SIPROLAB_G729A

4.9.3.129 WAVE_FORMAT_SIPROLAB_KELVIN

```
readonly Guid WAVE_FORMAT_SIPROLAB_KELVIN = new Guid((short)AudioEncoding.WAVE_FORMAT_SIP↩
ROLAB_KELVIN & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_SIPROLAB_KELVIN

4.9.3.130 WAVE_FORMAT_SOFTSOUND

```
readonly Guid WAVE_FORMAT_SOFTSOUND = new Guid((short)AudioEncoding.WAVE_FORMAT_SOFTSOUND &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_SOFTSOUND

4.9.3.131 WAVE_FORMAT_SONY_SCX

```
readonly Guid WAVE_FORMAT_SONY_SCX = new Guid((short)AudioEncoding.WAVE_FORMAT_SONY_SCX &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_SONY_SCX

4.9.3.132 WAVE_FORMAT_SOUNDSPACE_MUSICOMPRESS

```
readonly Guid WAVE_FORMAT_SOUNDSPACE_MUSICOMPRESS = new Guid((short)AudioEncoding.WAVE_FORM↵
AT_SOUNDSPACE_MUSICOMPRESS & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38,
0x9b, 0x71)  [static]
```

WAVE_FORMAT_SOUNDSPACE_MUSICOMPRESS

4.9.3.133 WAVE_FORMAT_TPC

```
readonly Guid WAVE_FORMAT_TPC = new Guid((short)AudioEncoding.WAVE_FORMAT_TPC & 0x0000FFFF,
0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_TPC

4.9.3.134 WAVE_FORMAT_TUBGSM

```
readonly Guid WAVE_FORMAT_TUBGSM = new Guid((short)AudioEncoding.WAVE_FORMAT_TUBGSM & 0x0000↵
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_TUBGSM

4.9.3.135 WAVE_FORMAT_UHER_ADPCM

```
readonly Guid WAVE_FORMAT_UHER_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_UHER_ADPCM &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_UHER_ADPCM

4.9.3.136 WAVE_FORMAT_UNISYS_NAP_16K

```
readonly Guid WAVE_FORMAT_UNISYS_NAP_16K = new Guid((short)AudioEncoding.WAVE_FORMAT_UNISYS_↩
NAP_16K & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_UNISYS_NAP_16K

4.9.3.137 WAVE_FORMAT_UNISYS_NAP_ADPCM

```
readonly Guid WAVE_FORMAT_UNISYS_NAP_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_UNI↩
SYS_NAP_ADPCM & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_UNISYS_NAP_ADPCM

4.9.3.138 WAVE_FORMAT_UNISYS_NAP_ALAW

```
readonly Guid WAVE_FORMAT_UNISYS_NAP_ALAW = new Guid((short)AudioEncoding.WAVE_FORMAT_UNI↩
SYS_NAP_ALAW & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_UNISYS_NAP_ALAW

4.9.3.139 WAVE_FORMAT_UNISYS_NAP_ULAW

```
readonly Guid WAVE_FORMAT_UNISYS_NAP_ULAW = new Guid((short)AudioEncoding.WAVE_FORMAT_UNI↩
SYS_NAP_ULAW & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)
[static]
```

WAVE_FORMAT_UNISYS_NAP_ULAW

4.9.3.140 WAVE_FORMAT_VIVO_G723

```
readonly Guid WAVE_FORMAT_VIVO_G723 = new Guid((short)AudioEncoding.WAVE_FORMAT_VIVO_G723 &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VIVO_G723

4.9.3.141 WAVE_FORMAT_VIVO_SIREN

```
readonly Guid WAVE_FORMAT_VIVO_SIREN = new Guid((short)AudioEncoding.WAVE_FORMAT_VIVO_SIREN &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VIVO_SIREN

4.9.3.142 WAVE_FORMAT_VME_VMPCM

```
readonly Guid WAVE_FORMAT_VME_VMPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_VME_VMPCM &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VME_VMPCM

4.9.3.143 WAVE_FORMAT_VOXWARE

```
readonly Guid WAVE_FORMAT_VOXWARE = new Guid((short)AudioEncoding.WAVE_FORMAT_VOXWARE & 0x0000↩
FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VOXWARE

4.9.3.144 WAVE_FORMAT_VOXWARE_AC10

```
readonly Guid WAVE_FORMAT_VOXWARE_AC10 = new Guid((short)AudioEncoding.WAVE_FORMAT_VOXWARE_A↩
C10 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VOXWARE_AC10

4.9.3.145 WAVE_FORMAT_VOXWARE_AC16

```
readonly Guid WAVE_FORMAT_VOXWARE_AC16 = new Guid((short)AudioEncoding.WAVE_FORMAT_VOXWARE_A↩
C16 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VOXWARE_AC16

4.9.3.146 WAVE_FORMAT_VOXWARE_AC20

```
readonly Guid WAVE_FORMAT_VOXWARE_AC20 = new Guid((short)AudioEncoding.WAVE_FORMAT_VOXWARE_A↩
C20 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VOXWARE_AC20

4.9.3.147 WAVE_FORMAT_VOXWARE_AC8

```
readonly Guid WAVE_FORMAT_VOXWARE_AC8 = new Guid((short)AudioEncoding.WAVE_FORMAT_VOXWARE_AC8
& 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VOXWARE_AC8

4.9.3.148 WAVE_FORMAT_VOXWARE_BYTE_ALIGNED

```
readonly Guid WAVE_FORMAT_VOXWARE_BYTE_ALIGNED = new Guid((short)AudioEncoding.WAVE_FORMAT_↩
VOXWARE_BYTE_ALIGNED & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b,
0x71)  [static]
```

WAVE_FORMAT_VOXWARE_BYTE_ALIGNED

4.9.3.149 WAVE_FORMAT_VOXWARE_RT24

```
readonly Guid WAVE_FORMAT_VOXWARE_RT24 = new Guid((short)AudioEncoding.WAVE_FORMAT_VOXWARE_R↩
T24 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VOXWARE_RT24

4.9.3.150 WAVE_FORMAT_VOXWARE_RT29

```
readonly Guid WAVE_FORMAT_VOXWARE_RT29 = new Guid((short)AudioEncoding.WAVE_FORMAT_VOXWARE_R↩
T29 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_VOXWARE_RT29

4.9.3.151 WAVE_FORMAT_VOXWARE_RT29HW

```
readonly Guid WAVE_FORMAT_VOXWARE_RT29HW = new Guid((short)AudioEncoding.WAVE_FORMAT_VOXWARE←
_RT29HW & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_VOXWARE_RT29HW

4.9.3.152 WAVE_FORMAT_VOXWARE_TQ40

```
readonly Guid WAVE_FORMAT_VOXWARE_TQ40 = new Guid((short)AudioEncoding.WAVE_FORMAT_VOXWARE_T←
Q40 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_VOXWARE_TQ40

4.9.3.153 WAVE_FORMAT_VOXWARE_TQ60

```
readonly Guid WAVE_FORMAT_VOXWARE_TQ60 = new Guid((short)AudioEncoding.WAVE_FORMAT_VOXWARE_T←
Q60 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_VOXWARE_TQ60

4.9.3.154 WAVE_FORMAT_VOXWARE_VR12

```
readonly Guid WAVE_FORMAT_VOXWARE_VR12 = new Guid((short)AudioEncoding.WAVE_FORMAT_VOXWARE_V←
R12 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_VOXWARE_VR12

4.9.3.155 WAVE_FORMAT_VOXWARE_VR18

```
readonly Guid WAVE_FORMAT_VOXWARE_VR18 = new Guid((short)AudioEncoding.WAVE_FORMAT_VOXWARE_V←
R18 & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_VOXWARE_VR18

4.9.3.156 WAVE_FORMAT_WMAVOICE9

```
readonly Guid WAVE_FORMAT_WMAVOICE9 = new Guid((short)AudioEncoding.WAVE_FORMAT_WMAVOICE9 &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

Windows Media Audio Voice (WMA Voice)

4.9.3.157 WAVE_FORMAT_XEBEC

```
readonly Guid WAVE_FORMAT_XEBEC = new Guid((short)AudioEncoding.WAVE_FORMAT_XEBEC & 0x0000FF←
FF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_XEBEC

4.9.3.158 WAVE_FORMAT_ZYXEL_ADPCM

```
readonly Guid WAVE_FORMAT_ZYXEL_ADPCM = new Guid((short)AudioEncoding.WAVE_FORMAT_ZYXEL_ADPCM
& 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

WAVE_FORMAT_ZYXEL_ADPCM

4.9.3.159 WindowsMediaAudio

```
readonly Guid WindowsMediaAudio = new Guid((short)AudioEncoding.WindowsMediaAudio & 0x0000FF←
FF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

Windows Media Audio, WAVE_FORMAT_WMAUDIO2, Microsoft Corporation

4.9.3.160 WindowsMediaAudioLosseless

```
readonly Guid WindowsMediaAudioLosseless = new Guid((short)AudioEncoding.WindowsMediaAudio←
Losseless & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71) [static]
```

Windows Media Audio Lossless, WAVE_FORMAT_WMAUDIO_LOSSLESS

4.9.3.161 WindowsMediaAudioProfessional

```
readonly Guid WindowsMediaAudioProfessional = new Guid((short)AudioEncoding.WindowsMedia←
AudioProfessional & 0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b,
0x71) [static]
```

Windows Media Audio Professional WAVE_FORMAT_WMAUDIO3, Microsoft Corporation

4.9.3.162 WindowsMediaAudioSpdif

```
readonly Guid WindowsMediaAudioSpdif = new Guid((short)AudioEncoding.WindowsMediaAudioSpdif &
0x0000FFFF, 0x0000, 0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

Windows Media Audio Professional over SPDIF WAVE_FORMAT_WMASPDIF (0x0164)

4.9.3.163 WmaVoice9

```
readonly Guid WmaVoice9 = new Guid((short)AudioEncoding.WmaVoice9 & 0x0000FFFF, 0x0000, 0x0010,
0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_WMAVOICE9

4.9.3.164 YamahaAdpcm

```
readonly Guid YamahaAdpcm = new Guid((short)AudioEncoding.YamahaAdpcm & 0x0000FFFF, 0x0000,
0x0010, 0x80, 0x00, 0x00, 0xaa, 0x00, 0x38, 0x9b, 0x71)  [static]
```

WAVE_FORMAT_YAMAHA_ADPCM Yamaha Corporation of America

4.10 AudioSyncBuffer< T > Class Template Reference

Inherits IAudioOut< T >.

Public Member Functions

- **AudioSyncBuffer** (int playDelayMs, ILogger logger, string logPrefix, bool debugInfo)
- void **Start** (int sampleRate, int channels, int frameSamples)
- void **Service** ()
- void **Read** (T[ ] outBuf, int outChannels, int outSampleRate)
- void **Push** (T[ ] frame)
- void **Flush** ()
- void **Stop** ()

Static Public Attributes

- const int **FRAME_POOL_CAPACITY** = 50

Properties

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

4.11 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 **Resample**< **T** > (T[ ] src, int srcOffset, int srcCount, T[ ] dst, int dstOffset, int dstCount, int channels)
- static void **Resample**< **T** > (T[ ] src, int srcOffset, int srcCount, int srcChannels, T[ ] dst, int dstOffset, int dstCount, int dstChannels)
- 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.

4.11.1 Detailed Description

Collection of Audio Utility functions and classes.

4.11.2 Member Function Documentation

4.11.2.1 Convert() [1/2]

```
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.

4.11.2.2 Convert() [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.

4.11.2.3 ForceToStereo< T >()

```
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.

4.11.2.4 Resample< T >()

```
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.

4.11.2.5 ResampleAndConvert() [1/2]

```
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.

4.11.2.6 ResampleAndConvert() [2/2]

```
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.

4.12 BufferReaderPushAdapter< T > Class Template Reference

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

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**

4.12.1 Detailed Description

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

4.12.2 Constructor & Destructor Documentation

4.12.2.1 BufferReaderPushAdapter()

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

Create a new BufferReaderPushAdapter instance

Parameters

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

4.12.3 Member Function Documentation

4.12.3.1 Service()

```
override void Service (
            LocalVoice localVoice )  [virtual]
```

Do the actual data read/push.

Parameters

localVoice	LocalVoice instance to push data to.

Implements BufferReaderPushAdapterBase< T >.

4.13 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

4.13.1 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.

4.13.2 Constructor & Destructor Documentation

4.13.2.1 BufferReaderPushAdapterAsyncPool()

```
BufferReaderPushAdapterAsyncPool (
            LocalVoice localVoice,
            IDataReader< T > reader )
```

Create a new BufferReaderPushAdapter instance

Parameters

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

4.13.3 Member Function Documentation

4.13.3.1 Service()

```
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 >.

4.14 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**

4.14.1 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 c

4.14.2 Constructor & Destructor Documentation

4.14.2.1 BufferReaderPushAdapterAsyncPoolCopy()

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

Create a new BufferReaderPushAdapter instance

Parameters

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

4.14.3 Member Function Documentation

4.14.3.1 Service()

```
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 >.

4.15 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

4.15.1 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.

4.15.2 Constructor & Destructor Documentation

4.15.2.1 BufferReaderPushAdapterAsyncPoolFloatToShort()

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

Create a new BufferReaderPushAdapter instance

Parameters

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

4.15.3 Member Function Documentation

4.15.3.1 Service()

```
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 >.

4.16 BufferReaderPushAdapterAsyncPoolShortToFloat Class Reference

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

Inherits BufferReaderPushAdapterBase< short >.

Public Member Functions

- BufferReaderPushAdapterAsyncPoolShortToFloat (LocalVoice localVoice, IDataReader< short > reader)

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

  Do the actual data read/push.

Additional Inherited Members

4.16.1 Detailed Description

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

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

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

4.16.2 Constructor & Destructor Documentation

4.16.2.1 BufferReaderPushAdapterAsyncPoolShortToFloat()

```
BufferReaderPushAdapterAsyncPoolShortToFloat (
            LocalVoice localVoice,
            IDataReader< short > reader )
```

Create a new BufferReaderPushAdapter instance

Parameters

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

4.16.3 Member Function Documentation

4.16.3.1 Service()

```
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< short >.

4.17 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 BufferReaderPushAdapterAsyncPool

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**

4.17.1 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.

4.17.2 Constructor & Destructor Documentation

4.17.2.1 BufferReaderPushAdapterBase()

```
BufferReaderPushAdapterBase (
            IDataReader< T > reader )
```

Create a new BufferReaderPushAdapterBase instance

Parameters

reader	DataReader to read from.

4.17.3 Member Function Documentation

4.17.3.1 Dispose()

```
void Dispose ( )
```

Release resources associated with this instance.

4.17.3.2 Service()

```
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 BufferReaderPushAdapterAsyncPoolShortToFloat, BufferReaderPushAdapterAsyncPoolFloatToShort, BufferReaderPushAdapterAsyncPoolCopy< T >, BufferReaderPushAdapterAsyncPool< T >, and BufferReaderPushAdapter< T >.

4.18 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]`

4.19 OpusCodec.Decoder< T > Class Template Reference

Inherits IDecoder.

Public Member Functions

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

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

    Consumes the given encoded data.

Protected Attributes

- OpusDecoder< T > **decoder**

Properties

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

4.19.1 Member Function Documentation

4.19.1.1 Input()

```
void Input (
             byte[ ] buf,
             FrameFlags flags )
```

Consumes the given encoded data.

Implements IDecoder.

4.19.1.2 Open()

```
void Open (
             VoiceInfo info )
```

Open (initialize) the decoder.

Parameters

info	Properties of the data stream to decode.

Implements IDecoder.

4.20 RawCodec.Decoder< T > Class Template Reference

Inherits IDecoder.

Public Member Functions

- **Decoder** (Action< FrameOut< T >> output)
- void Open (VoiceInfo info)

  Open (initialize) the decoder.
- void Input (byte[ ] byteBuf, FrameFlags flags)

  Consumes the given encoded data.
- void **Dispose** ()

Properties

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

4.20.1 Member Function Documentation

4.20.1.1 Input()

```
void Input (
            byte[ ] buf,
            FrameFlags flags )
```

Consumes the given encoded data.

Implements IDecoder.

4.20.1.2 Open()

```
void Open (
            VoiceInfo info )
```

Open (initialize) the decoder.

Parameters

info	Properties of the data stream to decode.

Implements IDecoder.

4.21 OpusCodec.DecoderFactory Class Reference

Static Public Member Functions

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

4.22 OpusCodec.Encoder< T > Class Template Reference

Inherits IEncoderDirect< T[ ]>.

Public Member Functions

- void **Input** (T[ ] buf)
- void **EndOfStream** ()
- ArraySegment< byte > **DequeueOutput** (out FrameFlags flags)
- I **GetPlatformAPI**< **I** > ()
- 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 >, FrameFlags > **Output** `[get, set]`

4.23 RawCodec.Encoder< T > Class Template Reference

Inherits IEncoderDirect< T[ ]>.

Public Member Functions

- ArraySegment< byte > **DequeueOutput** (out FrameFlags flags)
- void **EndOfStream** ()
- I **GetPlatformAPI**< **I** > ()
- void **Dispose** ()
- void **Input** (T[ ] buf)

Properties

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

4.24 OpusCodec.EncoderFloat Class Reference

Inherits OpusCodec.Encoder< float >.

Protected Member Functions

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

Additional Inherited Members

4.25 OpusCodec.EncoderShort Class Reference

Inherits OpusCodec.Encoder< short >.

Protected Member Functions

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

Additional Inherited Members

4.26 Extensions Class Reference

Provides a few basic extensions.

4.26.1 Detailed Description

Provides a few basic extensions.

4.27 OpusCodec.Factory Class Reference

Static Public Member Functions

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

4.28 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]`

4.28.1 Detailed Description

PrimitiveArrayPool<T> as wrapped in object factory interface.

Template Parameters

T	Array element type.

4.29 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]`

4.29.1 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.

4.30 FrameBuffer Struct Reference

Public Member Functions

- **FrameBuffer** (byte[ ] array, int offset, int count, FrameFlags flags, Action dispose)
- **FrameBuffer** (byte[ ] array, FrameFlags flags)
- byte[ ] **GetArrayAndRelease** (ref byte[ ] copyToArray)

Public Attributes

- readonly byte[ ] **array**
- readonly int **offset**
- readonly int **count**
- readonly Action **release**

Properties

- int **Length** `[get]`
- FrameFlags **Flags** `[get]`

4.31 FrameOut< T > Class Template Reference

Public Member Functions

- **FrameOut** (T[ ] buf, bool endOfStream)
- FrameOut< T > **Set** (T[ ] buf, bool endOfStream)

Properties

- T[ ] **Buf** `[get]`
- bool **EndOfStream** `[get]`

4.32 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.

4.32.1 Detailed Description

Utility class to re-frame audio packets.

4.32.2 Constructor & Destructor Documentation

4.32.2.1 Framer()

```
Framer (
             int frameSize )
```

Create new Framer instance.

4.32.3 Member Function Documentation

4.32.3.1 Count()

```
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.

4.32.3.2 Frame()

```
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).

4.33 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.

4.33.1 Detailed Description

Audio Source interface.

4.33.2 Property Documentation

4.33.2.1 Channels

`int Channels [get]`

Number of channels in the audio signal.

4.33.2.2 Error

`string Error [get]`

If not null, audio object is in invalid state.

4.33.2.3 SamplingRate

`int SamplingRate [get]`

Sampling rate of the audio signal (in Hz).

4.34 IAudioOut< T > Interface Template Reference

Inherited by AudioSyncBuffer< T >.

Public Member Functions

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

Properties

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

4.35 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

4.35.1 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,

4.35.2 Member Function Documentation

4.35.2.1 SetCallback()

```
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 >.

4.36 IAudioReader< T > Interface Template Reference

Audio Reader interface.

Inherits IDataReader< T >, and IAudioDesc.

Inherited by AudioUtil.ToneAudioReader< T >.

Additional Inherited Members

4.36.1 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).

4.37 IAudioSource Interface Reference

Defines the base for all audio streams.

Inherits IDisposable.

Inherited by IReadableAudioSource< in in T >.

Properties

- bool CanSeek `[get]`

  Gets a value indicating whether the IAudioSource supports seeking.
- WaveFormat WaveFormat `[get]`

  Gets the WaveFormat of the waveform-audio data.
- long Position `[get, set]`

  Gets or sets the current position. The unit of this property depends on the implementation of this interface. Some implementations may not support this property.
- long Length `[get]`

  Gets the length of the waveform-audio data. The unit of this property depends on the implementation of this interface. Some implementations may not support this property.

4.37.1 Detailed Description

Defines the base for all audio streams.

4.37.2 Property Documentation

4.37.2.1 CanSeek

```
bool CanSeek  [get]
```

Gets a value indicating whether the IAudioSource supports seeking.

4.37.2.2 Length

```
long Length  [get]
```

Gets the length of the waveform-audio data. The unit of this property depends on the implementation of this interface. Some implementations may not support this property.

4.37.2.3 Position

```
long Position  [get], [set]
```

Gets or sets the current position. The unit of this property depends on the implementation of this interface. Some implementations may not support this property.

4.37.2.4 WaveFormat

```
WaveFormat WaveFormat  [get]
```

Gets the WaveFormat of the waveform-audio data.

4.38 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.

4.38.1 Detailed Description

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

4.38.2 Member Function Documentation

4.38.2.1 Read()

```
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 >.

4.39 IDecoder Interface Reference

Generic decoder interface.

Inherits IDisposable.

Inherited by IDecoderDirect< B >, OpusCodec.Decoder< T >, and RawCodec.Decoder< T >.

Public Member Functions

- void Open (VoiceInfo info)

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

  Consumes the given encoded data.

Properties

- string Error `[get]`

  If not null, the object is in invalid state.

4.39.1 Detailed Description

Generic decoder interface.

4.39.2 Member Function Documentation

4.39.2.1 Input()

```
void Input (
             byte[ ] buf,
             FrameFlags flags )
```

Consumes the given encoded data.

Implemented in RawCodec.Decoder< T >, and OpusCodec.Decoder< T >.

4.39.2.2 Open()

```
void Open (
             VoiceInfo info )
```

Open (initialize) the decoder.

Parameters

info	Properties of the data stream to decode.

Implemented in RawCodec.Decoder< T >, and OpusCodec.Decoder< T >.

4.39.3 Property Documentation

4.39.3.1 Error

```
string Error  [get]
```

If not null, the object is in invalid state.

4.40 IDecoderDirect< B > Interface Template Reference

Interface for an decoder which outputs data via explicit call.

Inherits IDecoder.

Properties

- Action< B > **Output** `[get, set]`

Additional Inherited Members

4.40.1 Detailed Description

Interface for an decoder which outputs data via explicit call.

4.41 IDecoderQueuedOutputImageNative Interface Reference

Inherits IDecoderDirect< ImageOutputBuf >.

Properties

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

4.42 IEncoder Interface Reference

Generic encoder interface.

Inherits IDisposable.

Inherited by IEncoderDirect< B >.

Public Member Functions

- ArraySegment< byte > DequeueOutput (out FrameFlags flags)

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

    Forces an encoder to flush and produce frame with EndOfStream flag (in output queue).
- I **GetPlatformAPI< I >** ()

Properties

- string Error `[get]`

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

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

4.42.1 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.

4.42.2 Member Function Documentation

4.42.2.1 DequeueOutput()

```
ArraySegment<byte> DequeueOutput (
             out FrameFlags flags )
```

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

4.42.2.2 EndOfStream()

```
void EndOfStream ( )
```

Forces an encoder to flush and produce frame with EndOfStream flag (in output queue).

4.42.3 Property Documentation

4.42.3.1 Error

```
string Error [get]
```

If not null, the object is in invalid state.

4.42.3.2 Output

```
Action<ArraySegment<byte>, FrameFlags> Output [set]
```

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

4.43 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

4.43.1 Detailed Description

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

4.43.2 Member Function Documentation

4.43.2.1 Input()

```
void Input (
             B buf )
```

Consumes the given raw data.

Parameters

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

4.44 IEncoderDirectImage Interface Reference

Interface for an encoder which consumes images via explicit call.

Inherits IEncoderDirect< ImageBufferNative >.

Properties

- ImageFormat **ImageFormat** `[get]`

Additional Inherited Members

4.44.1 Detailed Description

Interface for an encoder which consumes images via explicit call.

4.45 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.

4.45.1 Detailed Description

Audio Level Metering interface.

4.45.2 Member Function Documentation

4.45.2.1 ResetAccumAvgPeakAmp()

```
void ResetAccumAvgPeakAmp ( )
```

Reset AccumAvgPeakAmp.

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

4.45.3 Property Documentation

4.45.3.1 AccumAvgPeakAmp

```
float AccumAvgPeakAmp  [get]
```

Average of CurrentPeakAmps since last reset.

4.45.3.2 CurrentAvgAmp

```
float CurrentAvgAmp  [get]
```

Average amplitude value over last half second.

4.45.3.3 CurrentPeakAmp

```
float CurrentPeakAmp  [get]
```

Maximum amplitude value over last half second sec.

4.46 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.

4.46.1 Detailed Description

Interface for an outgoing audio stream.

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

4.46.2 Member Function Documentation

4.46.2.1 VoiceDetectorCalibrate()

```
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 >.

4.46.3 Property Documentation

4.46.3.1 LevelMeter

```
AudioUtil.ILevelMeter LevelMeter  [get]
```

The LevelMeter utility in use.

4.46.3.2 VoiceDetector

```
AudioUtil.IVoiceDetector VoiceDetector  [get]
```

The VoiceDetector in use.

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

4.46.3.3 VoiceDetectorCalibrating

```
bool VoiceDetectorCalibrating  [get]
```

If true, voice detector calibration is in progress.

4.47 ILoggable Interface Reference

Inherited by ILoggableDependent, and VoiceConnection.

Properties

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

4.48 ILoggableDependent Interface Reference

Inherits ILoggable.

Inherited by VoiceComponent.

Properties

- bool **IgnoreGlobalLogLevel** `[get, set]`

4.49 ILogger Interface Reference

Inherited by LoadBalancingTransport, 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)

4.50 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]`

4.51 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]`

4.52 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

4.53 ImageBufferNativeGCHandleSinglePlane Class Reference

Inherits ImageBufferNative, and IDisposable.

Public Member Functions

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

Additional Inherited Members

4.54 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

4.55 ImageOutputBuf Struct Reference

Public Attributes

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

4.56 IOSAudioForceToSpeaker Class Reference

Inherits MonoBehaviour.

4.57 IProcessor< T > Interface Template Reference

Audio Processor interface.

Inherits IDisposable.

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

Public Member Functions

- T[ ] Process (T[ ] buf)

  Process a frame of audio data.

4.57.1 Detailed Description

Audio Processor interface.

4.57.2 Member Function Documentation

4.57.2.1 Process()

```
T [] Process (
            T[] buf )
```

Process a frame of audio data.

Parameters

buf	Buffer containing input audio data

Returns

Buffer containing output audio data or null if frame has been discarded (VAD)

Implemented in AudioUtil.VoiceLevelDetectCalibrate< T >, AudioUtil.VoiceDetector< T >, AudioUtil.VoiceDetectorCalibration< T >,
AudioUtil.LevelMeter< T >, and AudioUtil.Resampler< T >.

4.58 IReadableAudioSource< in in T > Interface Template Reference

Defines a generic base for all readable audio streams.

Inherits IAudioSource.

Public Member Functions

- int Read (T[ ] buffer, int offset, int count)

  Reads a sequence of elements from the IReadableAudioSource<T> and advances the position within the stream by the number of elements read.

Additional Inherited Members

4.58.1 Detailed Description

Defines a generic base for all readable audio streams.

Template Parameters

T	The type of the provided audio data.

4.58.2 Member Function Documentation

4.58.2.1 Read()

```
int Read (
            T[ ] buffer,
            int offset,
            int count )
```

Reads a sequence of elements from the IReadableAudioSource<T> and advances the position within the stream by the number of elements read.

Parameters

buffer	An array of elements. When this method returns, the *buffer* contains the specified array of elements with the values between *offset* and (*offset* + *count* - 1) replaced by the elements read from the current source.
offset	The zero-based offset in the *buffer* at which to begin storing the data read from the current stream.
count	The maximum number of elements to read from the current source.

Returns

The total number of elements read into the buffer.

4.59 IResettable Interface Reference

Inherited by AndroidAudioInAEC.

Public Member Functions

- void **Reset** ()

4.60 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.

4.60.1 Detailed Description

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

4.60.2 Member Function Documentation

4.60.2.1 Service()

```
void Service (
            LocalVoice localVoice )
```

Service function that should be called regularly.

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

4.61 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.

4.61.1 Detailed Description

Voice Activity Detector interface.

4.61.2 Property Documentation

4.61.2.1 ActivityDelayMs

```
int ActivityDelayMs  [get], [set]
```

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

4.61.2.2 Detected

```
bool Detected  [get]
```

If true, voice detected.

4.61.2.3 DetectedTime

```
DateTime DetectedTime  [get]
```

Last time when switched to detected state.

4.61.2.4 On

```
bool On  [get], [set]
```

If true, voice detection enabled.

4.61.2.5 Threshold

```
float Threshold  [get], [set]
```

Voice detected as soon as signal level exceeds threshold.

4.61.3 Event Documentation

4.61.3.1 OnDetected

```
Action OnDetected
```

Called when switched to detected state.

4.62 IVoiceTransport Interface Reference

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, FrameFlags flags, byte evNumber, byte voiceId, int channelId, int targetPlayerId, bool reliable, LocalVoice localVoice)
- string **ChannelIdStr** (int channelId)
- string **PlayerIdStr** (int playerId)

4.63 IWaveSource Interface Reference

Defines the base for all audio streams which provide raw byte data.

Inherits IReadableAudioSource< byte >.

Additional Inherited Members

4.63.1 Detailed Description

Defines the base for all audio streams which provide raw byte data.

Compared to the ISampleSource, the IWaveSource provides raw bytes instead of samples. That means that the IAudioSource.Position and the IAudioSource.Position properties are expressed in bytes. Also the IReadableAudioSource<T>.Read method provides samples instead of raw bytes.

4.64 IWriteable Interface Reference

Provides the Write method.

Inherited by WaveWriter.

Public Member Functions

- void Write (byte[ ] buffer, int offset, int count)

  Used to write down raw byte data.

4.64.1 Detailed Description

Provides the Write method.

4.64.2 Member Function Documentation

4.64.2.1 Write()

```
void Write (
          byte[ ] buffer,
          int offset,
          int count )
```

Used to write down raw byte data.

Parameters

buffer	Byte array which contains the data to write down.
offset	Zero-based offset in the *buffer* .
count	Number of bytes to write.

Implemented in WaveWriter.

4.65 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**
- float **currentPeakAmp**
- float **norm**

Properties

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

4.65.1 Detailed Description

Audio Level Meter.

4.65.2 Member Function Documentation

4.65.2.1 Process()

```
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 or null if frame has been discarded (VAD)

Implements IProcessor< T >.

4.65.2.2 ResetAccumAvgPeakAmp()

```
void ResetAccumAvgPeakAmp ( )
```

Reset AccumAvgPeakAmp.

Implements AudioUtil.ILevelMeter.

4.66 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]`

4.66.1 Detailed Description

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

4.66.2 Member Function Documentation

4.66.2.1 ResetAccumAvgPeakAmp()

```
void ResetAccumAvgPeakAmp ( )
```

Reset AccumAvgPeakAmp.

Implements AudioUtil.ILevelMeter.

4.67 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

4.67.1 Detailed Description

LevelMeter specialization for float audio.

4.67.2 Constructor & Destructor Documentation

4.67.2.1 LevelMeterFloat()

```
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.

4.68 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

4.68.1 Detailed Description

LevelMeter specialization for short audio.

4.68.2 Constructor & Destructor Documentation

4.68.2.1 LevelMeterShort()

```
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.

4.69 LoadBalancingFrontend Class Reference

Inherits LoadBalancingTransport.

Additional Inherited Members

4.70 LoadBalancingTransport Class Reference

Extends LoadBalancingClient with media streaming functionality.

Inherits LoadBalancingClient, IVoiceTransport, ILogger, and IDisposable.

Inherited by LoadBalancingFrontend, and LoadBalancingTransport2.

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)
- LoadBalancingTransport (ILogger logger=null, ConnectionProtocol connectionProtocol=Connection↩ Protocol.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 **SendVoiceRemove** (LocalVoice voice, int channelId, int targetPlayerId)
- virtual void **SendFrame** (ArraySegment< byte > data, FrameFlags flags, byte evNumber, byte voiceId, int channelId, int targetPlayerId, bool reliable, LocalVoice localVoice)
- string **ChannelIdStr** (int channelId)
- string **PlayerIdStr** (int playerId)
- void Dispose ()

  Releases all resources used by the LoadBalancingTransport instance.

Protected Member Functions

- virtual void **onEventActionVoiceClient** (EventData ev)

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 interest 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.

4.70.1 Detailed Description

Extends LoadBalancingClient with media streaming functionality.

Use your normal LoadBalancing workflow to join a Voice room. All standard LoadBalancing features are available. Use VoiceClient to work with media streams.

4.70.2 Constructor & Destructor Documentation

4.70.2.1 LoadBalancingTransport()

```
LoadBalancingTransport (
             ILogger logger = null,
             ConnectionProtocol connectionProtocol = ConnectionProtocol.Udp )
```

Initializes a new LoadBalancingTransport.

Parameters

logger	ILogger instance. If null, this instance LoadBalancingClient.DebugReturn implementation is used.ConnectionProtocol
connectionProtocol	Connection protocol (UDP or TCP). ConnectionProtocol

4.70.3 Member Function Documentation

4.70.3.1 Dispose()

```
void Dispose ( )
```

Releases all resources used by the LoadBalancingTransport instance.

4.70.3.2 Service()

```
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).

4.70.4 Property Documentation

4.70.4.1 GlobalInterestGroup

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

Set global interest 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[ ])

4.70.4.2 VoiceClient

```
VoiceClient VoiceClient  [get]
```

The VoiceClient implementation associated with this LoadBalancingTransport.

4.71 LoadBalancingTransport2 Class Reference

Variant of LoadBalancingTransport. Aims to be non-alloc at the cost of breaking compatibility with older clients.

Inherits LoadBalancingTransport.

Public Member Functions

- **LoadBalancingTransport2** (ILogger logger=null, ConnectionProtocol connectionProtocol=Connection↵ Protocol.Udp)
- override void **SendFrame** (ArraySegment< byte > data, FrameFlags flags, byte evNumber, byte voiceId, int channelId, int targetPlayerId, bool reliable, LocalVoice localVoice)

Protected Member Functions

- override void **onEventActionVoiceClient** (EventData ev)

Additional Inherited Members

4.71.1 Detailed Description

Variant of LoadBalancingTransport. Aims to be non-alloc at the cost of breaking compatibility with older clients.

4.72 LocalVoice Class Reference

Represents outgoing data stream.

Inherits IDisposable.

Inherited by LocalVoiceAudioDummy, and LocalVoiceFramedBase.

Public Member Functions

- void RemoveSelf ()

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

Static Public Attributes

- const int **DATA_POOL_CAPACITY** = 50

Protected Attributes

- VoiceInfo **info**
- IEncoder **encoder**
- VoiceClient **voiceClient**
- ArraySegment< byte > **configFrame**
- 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]`

  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.

4.72.1 Detailed Description

Represents outgoing data stream.

4.72.2 Member Function Documentation

4.72.2.1 RemoveSelf()

```
void RemoveSelf ( )
```

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

.

4.72.3 Property Documentation

4.72.3.1 DebugEchoMode

```
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.

4.72.3.2 Encrypt

```
bool Encrypt  [get], [set]
```

Send data encrypted.

4.72.3.3 FramesSent

```
int FramesSent  [get]
```

Sent frames counter.

4.72.3.4 FramesSentBytes

```
int FramesSentBytes  [get]
```

Sent frames bytes counter.

4.72.3.5 Info

```
VoiceInfo Info  [get]
```

Returns Info structure assigned on local voice cration.

4.72.3.6 InterestGroup

```
byte InterestGroup  [get], [set]
```

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

4.72.3.7 IsCurrentlyTransmitting

```
bool IsCurrentlyTransmitting  [get]
```

Returns true if stream broadcasts.

4.72.3.8 LocalUserServiceable

```
IServiceable LocalUserServiceable  [get], [set]
```

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

4.72.3.9 Reliable

```
bool Reliable  [get], [set]
```

Send data reliable.

4.72.3.10 TransmitEnabled

```
bool TransmitEnabled  [get], [set]
```

If true, stream data broadcasted.

4.73 LocalVoiceAudio< T > Class Template Reference

Outgoing audio stream.

Inherits LocalVoiceFramed< T >, and ILocalVoiceAudio.

Public Member Functions

- 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

4.73.1 Detailed Description

Outgoing audio stream.

4.73.2 Member Function Documentation

4.73.2.1 Create()

```
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.

4.73.2.2 VoiceDetectorCalibrate()

```
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.

4.73.3 Property Documentation

4.73.3.1 VoiceDetectorCalibrating

```
bool VoiceDetectorCalibrating  [get]
```

True if the VoiceDetector is currently calibrating.

4.74 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

4.74.1 Detailed Description

Dummy LocalVoiceAudio

For testing, this LocalVoiceAudio implementation features a AudioUtil.VoiceDetectorDummy and a AudioUtil.LevelMeterDummy

4.74.2 Member Function Documentation

4.74.2.1 VoiceDetectorCalibrate()

```
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.

4.74.3 Member Data Documentation

4.74.3.1 Dummy

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

A Dummy LocalVoiceAudio instance.

4.75 LocalVoiceAudioFloat Class Reference

Specialization of LocalVoiceAudio for float audio

Inherits LocalVoiceAudio< float >.

Additional Inherited Members

4.75.1 Detailed Description

Specialization of LocalVoiceAudio for float audio

4.76 LocalVoiceAudioShort Class Reference

Specialization of LocalVoiceAudio for short audio

Inherits LocalVoiceAudio< short >.

Additional Inherited Members

4.76.1 Detailed Description

Specialization of LocalVoiceAudio for short audio

4.77 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'.

Protected Member Functions

- T[ ] **processFrame** (T[ ] buf)

Properties

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

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

Additional Inherited Members

4.77.1 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.
channel↩ Id	Transport channel specific to transport.
encoder	Encoder producing the stream.

Returns

Outgoing stream handler.

4.77.2 Member Function Documentation

4.77.2.1 AddPostProcessor()

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

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

Parameters

processors	

4.77.2.2 AddPreProcessor()

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

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

Parameters

processors	

4.77.2.3 ClearProcessors()

```
void ClearProcessors ( )
```

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

4.77.2.4 Dispose()

```
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.

4.77.2.5 PushData()

```
void PushData (
             T[] buf )
```

Synchronously push data into this stream.

4.77.2.6 PushDataAsync()

```
void PushDataAsync (
             T[] buf )
```

Asynchronously push data into this stream.

4.77.3 Property Documentation

4.77.3.1 PushDataAsyncReady

```
bool PushDataAsyncReady  [get]
```

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

4.78 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

4.78.1 Detailed Description

Typed re-framing LocalVoice

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

4.78.2 Property Documentation

4.78.2.1 FrameSize

```
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.

4.79 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)

4.80 MicAmplifier Class Reference

Inherits VoiceComponent.

Properties

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

Additional Inherited Members

4.81 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]`

4.82 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]`

4.83 MicWrapper Class Reference

Inherits IAudioReader< float >.

Public Member Functions

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

Properties

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

4.84 MicWrapperPusher Class Reference

Inherits IAudioPusher< float >.

Public Member Functions

- **MicWrapperPusher** (string device, AudioSource aS, int suggestedFrequency, ILogger lg, bool destroyOn↩ Stop=true)
- **MicWrapperPusher** (string device, GameObject gO, int suggestedFrequency, ILogger lg, bool destroyOn↩ Stop=true)
- **MicWrapperPusher** (string device, Transform parentTransform, int suggestedFrequency, ILogger lg, bool destroyOnStop=true)
- void **SetCallback** (Action< float[ ]> callback, ObjectFactory< float[ ], int > bufferFactory)
- void **Dispose** ()

Properties

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

4.85 MonoPInvokeCallbackAttribute Class Reference

Inherits Attribute.

Public Member Functions

- **MonoPInvokeCallbackAttribute** (Type t)

4.86 NativeAndroidMicrophoneSettings Struct Reference

Public Attributes

- bool **AcousticEchoCancellation**
- bool **AutomaticGainControl**
- bool **NoiseSuppression**

4.87 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]`

4.87.1 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).

4.88 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.

4.88.1 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).

4.88.2 Constructor & Destructor Documentation

4.88.2.1 ObjectPool() `[1/2]`

```
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.

4.88.2.2 ObjectPool() `[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.

4.88.3 Member Function Documentation

4.88.3.1 AcquireOrCreate() `[1/2]`

```
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.

4.88.3.2 AcquireOrCreate() `[2/2]`

```
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.

4.88.3.3 Dispose()

```
void Dispose ( )
```

Free resources assoicated with this ObjectPool

4.88.3.4 Init()

```
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.

4.88.3.5 Release() `[1/2]`

```
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.

4.88.3.6 Release() [2/2]

```
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.

4.88.4 Property Documentation

4.88.4.1 Info

```
TInfo Info  [get]
```

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

4.89 OpusCodec Class Reference

Classes

- class Decoder
- class DecoderFactory
- class Encoder
- class EncoderFloat
- class EncoderShort
- class Factory
- class Util

Public Types

- enum **FrameDuration**

Properties

- static string **Version** `[get]`

4.90 OpusDecoder< T > Class Template Reference

Inherits IDisposable.

Public Member Functions

- **OpusDecoder** (SamplingRate outputSamplingRateHz, Channels numChannels)
- T[ ] **DecodePacket** (byte[ ] packetData)
- T[ ] **DecodeEndOfStream** ()
- void **Dispose** ()

Properties

- Bandwidth? **PreviousPacketBandwidth** `[get]`

4.91 OpusEncoder Class Reference

Inherits IDisposable.

Public Member Functions

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

Static Public Attributes

- const int **BitrateMax** = -1

Properties

- SamplingRate **InputSamplingRate** `[get]`
- Channels **InputChannels** `[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]`
- int **PacketLossPercentage** `[get, set]`
- bool? **UseUnconstrainedVBR** `[get, set]`
- bool? **DtxEnabled** `[get, set]`

4.91.1 Property Documentation

4.91.1.1 EncoderDelay

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

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

4.92 OpusException Class Reference

Inherits Exception.

Public Member Functions

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

Properties

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

4.93 OpusLib Class Reference

Properties

- static string **Version** `[get]`

4.94 Recorder.PhotonVoiceCreatedParams Class Reference

Inherits PhotonVoiceCreatedParams.

Additional Inherited Members

4.95 PhotonVoiceCreatedParams Class Reference

Inherited by Recorder.PhotonVoiceCreatedParams.

Properties

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

4.96 PhotonVoiceLagSimulationGui Class Reference

Inherits MonoBehaviour.

Public Member Functions

- void **OnEnable** ()

4.97 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

- 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 WorkInOfflineMode = true

  Whether or not Photon Voice client should follow PUN client if the latter is in offline mode.

Static Public Attributes

- const string VoiceRoomNameSuffix = "_voice_"

  Suffix for voice room names appended to PUN room names.

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
- bool UsePunAuthValues `[get, set]`

  Whether or not to use the same PhotonNetwork.AuthValues in PhotonVoiceNetwork.Instance.Client.AuthValues.

Additional Inherited Members

4.97.1 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.

4.97.2 Member Function Documentation

4.97.2.1 ConnectAndJoinRoom()

```
bool ConnectAndJoinRoom ( )
```

Connect voice client to Photon servers and join a Voice room

Returns

If true, connection command send from client

4.97.2.2 Disconnect()

```
void Disconnect ( )
```

Disconnect voice client from all Photon servers

4.97.3 Member Data Documentation

4.97.3.1 AutoConnectAndJoin

```
bool AutoConnectAndJoin = true
```

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

4.97.3.2 AutoLeaveAndDisconnect

```
bool AutoLeaveAndDisconnect = true
```

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

4.97.3.3 VoiceRoomNameSuffix

```
const string VoiceRoomNameSuffix = "_voice_"  [static]
```

Suffix for voice room names appended to PUN room names.

4.97.3.4 WorkInOfflineMode

```
bool WorkInOfflineMode = true
```

Whether or not Photon Voice client should follow PUN client if the latter is in offline mode.

4.97.4 Property Documentation

4.97.4.1 Instance

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

Singleton instance for PhotonVoiceNetwork

4.97.4.2 UsePunAuthValues

```
bool UsePunAuthValues  [get], [set]
```

Whether or not to use the same PhotonNetwork.AuthValues in PhotonVoiceNetwork.Instance.Client.AuthValues.

4.98 PhotonVoiceStatsGui Class Reference

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

Inherits MonoBehaviour.

4.98.1 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).

4.99 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 Member Functions

- void Init ()

    Initializes this PhotonVoiceView for Voice usage based on the PhotonView, Recorder and Speaker components.

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]`

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

    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]`

    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
- bool IsSpeakerLinked `[get]`

    If true, the SpeakerInUse is linked to the remote voice stream
- bool IsPhotonViewReady `[get]`

    If true, the PhotonView attached to the same GameObject has a valid ViewID $>$ 0

Additional Inherited Members

4.99.1 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.

4.99.2 Member Function Documentation

4.99.2.1 Init()

```
void Init ( )
```

Initializes this PhotonVoiceView for Voice usage based on the PhotonView, Recorder and Speaker components.

The initialization should happen automatically. Call this method explicitly if this does not succeed. The initialization is a two steps operation: step one is the setup of Recorder and Speaker to be used. Step two is the late-linking -if needed- of the SpeakerInUse and corresponding remote voice info -if any- via ViewID.

4.99.3 Member Data Documentation

4.99.3.1 AutoCreateRecorderIfNotFound

```
bool AutoCreateRecorderIfNotFound
```

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

4.99.3.2 SetupDebugSpeaker

```
bool SetupDebugSpeaker
```

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

4.99.3.3 UsePrimaryRecorder

```
bool UsePrimaryRecorder
```

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

4.99.4 Property Documentation

4.99.4.1 IsPhotonViewReady

```
bool IsPhotonViewReady  [get]
```

If true, the PhotonView attached to the same GameObject has a valid ViewID > 0

4.99.4.2 IsRecorder

```
bool IsRecorder  [get]
```

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

4.99.4.3 IsRecording

```
bool IsRecording  [get]
```

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

4.99.4.4 IsSetup

```
bool IsSetup  [get]
```

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

4.99.4.5 IsSpeaker

```
bool IsSpeaker  [get]
```

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

4.99.4.6 IsSpeakerLinked

```
bool IsSpeakerLinked  [get]
```

If true, the SpeakerInUse is linked to the remote voice stream

4.99.4.7 IsSpeaking

```
bool IsSpeaking  [get]
```

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

4.99.4.8 RecorderInUse

```
Recorder RecorderInUse  [get], [set]
```

The Recorder component currently used by this PhotonVoiceView

4.99.4.9 SpeakerInUse

```
Speaker SpeakerInUse  [get], [set]
```

The Speaker component currently used by this PhotonVoiceView

4.100 Platform Class Reference

Static Public Member Functions

- static IEncoder **CreateDefaultAudioEncoder**$<$ **T** $>$ (ILogger logger, VoiceInfo info)

4.101 PlaybackDelaySettings Struct Reference

Playback delay configuration container.

Public Member Functions

- override string **ToString** ()

Public Attributes

- int MinDelaySoft

  ms: Audio player tries to keep the delay above this value.
- int MaxDelaySoft

  ms: Audio player tries to keep the delay below this value.
- int MaxDelayHard

  ms: Audio player guarantees that the delay never exceeds this value.

Static Public Attributes

- const int **DEFAULT_LOW** = 200
- const int **DEFAULT_HIGH** = 400
- const int **DEFAULT_MAX** = 1000

4.101.1 Detailed Description

Playback delay configuration container.

4.101.2 Member Data Documentation

4.101.2.1 MaxDelayHard

```
int MaxDelayHard
```

ms: Audio player guarantees that the delay never exceeds this value.

4.101.2.2 MaxDelaySoft

```
int MaxDelaySoft
```

ms: Audio player tries to keep the delay below this value.

4.101.2.3 MinDelaySoft

```
int MinDelaySoft
```

ms: Audio player tries to keep the delay above this value.

4.102 UnityAudioOut.PlayDelayConfig Struct Reference

Static Public Attributes

- static PlayDelayConfig **Default** = new PlayDelayConfig { Low = 200, High = 400, Max = 1000 }

Properties

- int **Low** `[get, set]`
- int **High** `[get, set]`
- int **Max** `[get, set]`

4.103 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

4.103.1 Detailed Description

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

Template Parameters

T	Array element type.

4.104 RawCodec Class Reference

Classes

- class Decoder
- class Encoder

4.105 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 Init (VoiceConnection voiceConnection)

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

  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 background noise level.
- void StartRecording ()

  Starts recording.
- void StopRecording ()

  Stops recording.
- bool ResetLocalAudio ()

  Resets audio session and parameters locally to fix broken recording due to system configuration modifications or audio interruptions or audio routing changes.

Static Public Member Functions

- static bool **CompareUnityMicNames** (string mic1, string mic2)
- static bool **IsDefaultUnityMic** (string mic)

Static Public Attributes

- const int **MIN_OPUS_BITRATE** = 6000
- const int **MAX_OPUS_BITRATE** = 510000

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(VoiceConnection).
- 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.
- 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]`

  If true, the Recorder will automatically restart recording to recover from audio device changes.
- bool AutoStart `[get, set]`

  If true, automatically start recording when initialized.
- bool RecordOnlyWhenEnabled `[get, set]`

  If true, component will work only when enabled and active in hierarchy.
- bool SkipDeviceChangeChecks `[get, set]`

  If true, restarts recording without checking if audio config/device changes affected recording.
- bool StopRecordingWhenPaused `[get, set]`

  If true, stop recording when paused resume/restart when un-paused.
- bool UseOnAudioFilterRead `[get, set]`

  If true, recording will make use of Unity's OnAudioFitlerRead callback from a muted local AudioSource.
- bool TrySamplingRateMatch `[get, set]`

  If true, Recorder will try to match sampling rates of microphone device and Opus encoder to avoid re sampling of audio input.
- bool UseMicrophoneTypeFallback `[get, set]`

  If true, if recording fails to start with Unity microphone type, Photon microphone type is used -if available- as a fallback and vice versa.

Additional Inherited Members

4.105.1 Detailed Description

Component representing outgoing audio stream in scene.

4.105.2 Member Function Documentation

4.105.2.1 Init() `[1/2]`

```
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

4.105.2.2 Init() [2/2]

```
void Init (
            VoiceConnection voiceConnection )
```

Initializes the Recorder component to be able to transmit audio.

Parameters

voiceConnection	The VoiceConnection to be used with this Recorder.

4.105.2.3 ResetLocalAudio()

```
bool ResetLocalAudio ( )
```

Resets audio session and parameters locally to fix broken recording due to system configuration modifications or audio interruptions or audio routing changes.

Returns

If reset is done.

4.105.2.4 RestartRecording()

```
void RestartRecording (
            bool force = false )
```

Restarts recording if something has changed that requires this.

Parameters

force	Set to true if you want to restart even if this is not required (RequiresRestart = false)

4.105.2.5 StartRecording()

```
void StartRecording ( )
```

Starts recording.

4.105.2.6 StopRecording()

```
void StopRecording ( )
```

Stops recording.

4.105.2.7 VoiceDetectorCalibrate()

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

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

Parameters

durationMs	Duration of calibration in milliseconds.
detectionEndedCallback	Callback when VAD calibration ends.

4.105.3 Property Documentation

4.105.3.1 AudioClip

```
AudioClip AudioClip  [get], [set]
```

Source audio clip.

4.105.3.2 AudioGroup

```
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.

4.105.3.3 AutoStart

```
bool AutoStart  [get], [set]
```

If true, automatically start recording when initialized.

4.105.3.4 Bitrate

```
int Bitrate  [get], [set]
```

Outgoing audio stream bitrate.

4.105.3.5 DebugEchoMode

```
bool DebugEchoMode  [get], [set]
```

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

4.105.3.6 Encrypt

```
bool Encrypt  [get], [set]
```

If true, voice stream is sent encrypted.

4.105.3.7 FrameDuration

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

Outgoing audio stream encoder delay.

4.105.3.8 InputFactory

```
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

4.105.3.9 InterestGroup

```
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.

4.105.3.10 IsCurrentlyTransmitting

```
bool IsCurrentlyTransmitting  [get]
```

Returns true if audio stream broadcasts.

4.105.3.11 IsInitialized

```
bool IsInitialized  [get]
```

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

4.105.3.12 IsRecording

```
bool IsRecording  [get], [set]
```

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

4.105.3.13 LevelMeter

```
AudioUtil.ILevelMeter?  LevelMeter  [get]
```

Level meter utility.

4.105.3.14 LoopAudioClip

```
bool LoopAudioClip  [get], [set]
```

Loop playback for audio clip sources.

4.105.3.15 MicrophoneType

```
MicType MicrophoneType  [get], [set]
```

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

4.105.3.16 PhotonMicrophoneDeviceId

```
int PhotonMicrophoneDeviceId  [get], [set]
```

Set or get photon microphone device used for streaming.

4.105.3.17 PhotonMicrophoneEnumerator

```
AudioInEnumerator PhotonMicrophoneEnumerator  [static], [get]
```

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

4.105.3.18 ReactOnSystemChanges

```
bool ReactOnSystemChanges  [get], [set]
```

If true, the Recorder will automatically restart recording to recover from audio device changes.

By default, the Recorder will restart recording only when the Recorder.SourceType is InputSourceType.Microphone and the device being used is no longer available or valid, in some cases you may need to force restarts even if the device in use did not change. To enable this set Recorder.SkipDeviceChangeChecks to true.

4.105.3.19 RecordOnlyWhenEnabled

```
bool RecordOnlyWhenEnabled  [get], [set]
```

If true, component will work only when enabled and active in hierarchy.

4.105.3.20 ReliableMode

```
bool ReliableMode  [get], [set]
```

If true, stream data sent in reliable mode.

4.105.3.21 RequiresRestart

```
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.

4.105.3.22 SamplingRate

```
SamplingRate SamplingRate  [get], [set]
```

Outgoing audio stream sampling rate.

4.105.3.23 SkipDeviceChangeChecks

```
bool SkipDeviceChangeChecks  [get], [set]
```

If true, restarts recording without checking if audio config/device changes affected recording.

To be used when Recorder.ReactOnSystemChanges is true.

4.105.3.24 SourceType

```
InputSourceType SourceType  [get], [set]
```

Audio data source.

4.105.3.25 StopRecordingWhenPaused

```
bool StopRecordingWhenPaused  [get], [set]
```

If true, stop recording when paused resume/restart when un-paused.

4.105.3.26 TransmitEnabled

```
bool TransmitEnabled  [get], [set]
```

If true, audio transmission is enabled.

4.105.3.27 TrySamplingRateMatch

```
bool TrySamplingRateMatch [get], [set]
```

If true, Recorder will try to match sampling rates of microphone device and Opus encoder to avoid re sampling of audio input.

4.105.3.28 TypeConvert

```
SampleTypeConv TypeConvert [get], [set]
```

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

4.105.3.29 UnityMicrophoneDevice

```
string UnityMicrophoneDevice [get], [set]
```

Set or get Unity microphone device used for streaming.

4.105.3.30 UseMicrophoneTypeFallback

```
bool UseMicrophoneTypeFallback [get], [set]
```

If true, if recording fails to start with Unity microphone type, Photon microphone type is used -if available- as a fallback and vice versa.

4.105.3.31 UseOnAudioFilterRead

```
bool UseOnAudioFilterRead [get], [set]
```

If true, recording will make use of Unity's OnAudioFitlerRead callback from a muted local AudioSource.

If enabled, 3D sounds and voice positioning can be lost.

4.105.3.32 UserData

```
object UserData [get], [set]
```

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

4.105.3.33 VoiceDetection

```
bool VoiceDetection  [get], [set]
```

If true, voice detection enabled.

4.105.3.34 VoiceDetectionDelayMs

```
int VoiceDetectionDelayMs  [get], [set]
```

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

4.105.3.35 VoiceDetectionThreshold

```
float VoiceDetectionThreshold  [get], [set]
```

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

4.105.3.36 VoiceDetector

```
AudioUtil.IVoiceDetector?  VoiceDetector  [get]
```

Returns voice activity detector for recorder's audio stream.

4.105.3.37 VoiceDetectorCalibrating

```
bool VoiceDetectorCalibrating  [get]
```

If true, voice detector calibration is in progress.

4.106 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).

4.106.1 Detailed Description

Information about a remote voice (incoming stream).

4.106.2 Property Documentation

4.106.2.1 ChannelId

```
int ChannelId [get]
```

ID of channel used for transmission.

4.106.2.2 Info

```
VoiceInfo Info [get]
```

Remote voice info.

4.106.2.3 PlayerId

```
int PlayerId [get]
```

Player ID of voice owner.

4.106.2.4 VoiceId

```
byte VoiceId [get]
```

Voice ID (unique in the room).

4.107 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< FrameOut< float > > **FloatFrameDecoded**
- Action **RemoteVoiceRemoved**

4.108 RemoteVoiceOptions Struct Reference

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

Public Member Functions

- void SetOutput (Action< FrameOut< float >> output)

  Register a method to be called when new data frame received..
- void **SetOutput** (Action< FrameOut< 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]`

4.108.1 Detailed Description

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

4.108.2 Member Function Documentation

4.108.2.1 SetOutput()

```
void SetOutput (
             Action< FrameOut< float >> output )
```

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

4.108.3 Property Documentation

4.108.3.1 Decoder

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

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

4.108.3.2 OnRemoteVoiceRemoveAction

```
Action OnRemoteVoiceRemoveAction  [get], [set]
```

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

4.109 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**

4.109.1 Detailed Description

Sample-rate conversion Audio Processor.

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

4.109.2 Constructor & Destructor Documentation

4.109.2.1 Resampler()

```
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.

4.109.3 Member Function Documentation

4.109.3.1 Process()

```
T [] Process (
            T[] buf )
```

Process a frame of audio data.

Parameters

buf	Buffer containing input audio data

Returns

Buffer containing output audio data or null if frame has been discarded (VAD)

Implements IProcessor< T >.

4.110 SaveIncomingStreamToFile Class Reference

Inherits VoiceComponent.

Protected Member Functions

- override void **Awake** ()

Additional Inherited Members

4.111 SaveOutgoingStreamToFile Class Reference

Inherits VoiceComponent.

Additional Inherited Members

4.112 SavWav Class Reference

Static Public Member Functions

- static bool **Save** (string filename, AudioClip clip, bool trim=false)
- static byte[ ] **GetWav** (AudioClip clip, out uint length, bool trim=false)

4.113 Speaker Class Reference

Component representing remote audio stream in local scene.

Inherits VoiceComponent.

Public Member Functions

- bool StartPlayback ()

  Starts the audio playback of the linked incoming remote audio stream via AudioSource component.
- bool StopPlayback ()

  Stops the audio playback of the linked incoming remote audio stream via AudioSource component.
- bool RestartPlayback ()

  Restarts the audio playback of the linked incoming remote audio stream via AudioSource component.
- bool SetPlaybackDelaySettings (PlaybackDelaySettings pdc)

  Sets the settings for the playback behaviour in case of delays.
- bool SetPlaybackDelaySettings (int low, int high, int max)

  Sets the settings for the playback behaviour in case of delays.

Properties

- int **PlayDelayMs** `[get, set]`
- 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.
- bool PlaybackOnlyWhenEnabled `[get, set]`

  If true, component will work only when enabled and active in hierarchy.
- bool PlaybackStarted `[get]`

  Returns if the playback is on.
- int PlaybackDelayMinSoft `[get]`

  Gets the value in ms above which the audio player tries to keep the delay.
- int PlaybackDelayMaxSoft `[get]`

  Gets the value in ms below which the audio player tries to keep the delay.
- int PlaybackDelayMaxHard `[get]`

  Gets the value in ms that audio play delay will not exceed.

Additional Inherited Members

4.113.1 Detailed Description

Component representing remote audio stream in local scene.

4.113.2 Member Function Documentation

4.113.2.1 RestartPlayback()

```
bool RestartPlayback ( )
```

Restarts the audio playback of the linked incoming remote audio stream via AudioSource component.

Returns

True if playback is successfully restarted.

4.113.2.2 SetPlaybackDelaySettings() `[1/2]`

```
bool SetPlaybackDelaySettings (
            int low,
            int high,
            int max )
```

Sets the settings for the playback behaviour in case of delays.

Parameters

low	In milliseconds, audio player tries to keep the playback delay above this value.
high	In milliseconds, audio player tries to keep the playback below above this value.
max	In milliseconds, audio player guarantees that the playback delay never exceeds this value.

Returns

If a change has been made.

4.113.2.3 SetPlaybackDelaySettings() [2/2]

```
bool SetPlaybackDelaySettings (
            PlaybackDelaySettings pdc )
```

Sets the settings for the playback behaviour in case of delays.

Parameters

pdc	Playback delay configuration struct.

Returns

If a change has been made.

4.113.2.4 StartPlayback()

```
bool StartPlayback ( )
```

Starts the audio playback of the linked incoming remote audio stream via AudioSource component.

Returns

True if playback is successfully started.

4.113.2.5 StopPlayback()

```
bool StopPlayback ( )
```

Stops the audio playback of the linked incoming remote audio stream via AudioSource component.

Returns

True if playback is successfully stopped.

4.113.3 Property Documentation

4.113.3.1 Actor

```
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.

4.113.3.2 IsLinked

```
bool IsLinked  [get]
```

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

4.113.3.3 IsPlaying

```
bool IsPlaying  [get]
```

Is the speaker playing right now.

4.113.3.4 Lag

```
int?  Lag  [get]
```

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

4.113.3.5 OnRemoteVoiceRemoveAction

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

Register a method to be called when remote voice removed.

4.113.3.6 PlaybackDelayMaxHard

```
int PlaybackDelayMaxHard  [get]
```

Gets the value in ms that audio play delay will not exceed.

4.113.3.7 PlaybackDelayMaxSoft

```
int PlaybackDelayMaxSoft  [get]
```

Gets the value in ms below which the audio player tries to keep the delay.

4.113.3.8 PlaybackDelayMinSoft

```
int PlaybackDelayMinSoft  [get]
```

Gets the value in ms above which the audio player tries to keep the delay.

4.113.3.9 PlaybackOnlyWhenEnabled

```
bool PlaybackOnlyWhenEnabled  [get], [set]
```

If true, component will work only when enabled and active in hierarchy.

4.113.3.10 PlaybackStarted

```
bool PlaybackStarted  [get]
```

Returns if the playback is on.

4.114 TestTone Class Reference

Inherits MonoBehaviour.

4.115 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=48000, 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]`

4.115.1 Detailed Description

IAudioPusher that provides a constant tone signal.

4.115.2 Constructor & Destructor Documentation

4.115.2.1 ToneAudioPusher()

```
ToneAudioPusher (
            int frequency = 440,
            int bufSizeMs = 100,
            int samplingRate = 48000,
            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.

4.115.3 Member Function Documentation

4.115.3.1 SetCallback()

```
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 >.

4.116 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]

4.117 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=48000, 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.

4.117.1 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.

4.117.2 Constructor & Destructor Documentation

4.117.2.1 ToneAudioReader()

```
ToneAudioReader (
            Func< double > clockSec = null,
            double frequency = 440,
            int samplingRate = 48000,
            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.

4.117.3 Member Function Documentation

4.117.3.1 Read()

```
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 >.

4.117.4 Property Documentation

4.117.4.1 Channels

```
int Channels  [get]
```

Number of channels in the audio signal.

4.117.4.2 Error

```
string Error  [get]
```

If not null, audio object is in invalid state.

4.117.4.3 SamplingRate

```
int SamplingRate  [get]
```

Sampling rate of the audio signal (in Hz).

4.118 UnityAudioOut Class Reference

Inherits IAudioOut< float >.

Classes

- struct PlayDelayConfig

Public Member Functions

- **UnityAudioOut** (AudioSource audioSource, PlayDelayConfig playDelayConfig, ILogger logger, string log↩ Prefix, bool debugInfo)
- void **Start** (int frequency, int channels, int frameSamples)
- void **Service** ()
- void **Push** (float[ ] frame)
- void **Flush** ()
- void **Stop** ()

Static Public Attributes

- const int **FRAME_POOL_CAPACITY** = 50

Properties

- int? **Lag** `[get]`
- bool **IsFlushed** `[get]`
- bool **IsPlaying** `[get]`

4.119 UnityMicrophone Class Reference

A wrapper around UnityEngine.Microphone to be able to safely use Microphone and compile for WebGL.

Static Public Member Functions

- static void **End** (string deviceName)
- static void **GetDeviceCaps** (string deviceName, out int minFreq, out int maxFreq)
- static int **GetPosition** (string deviceName)
- static bool **IsRecording** (string deviceName)
- static AudioClip **Start** (string deviceName, bool loop, int lengthSec, int frequency)

Properties

- static string[ ] **devices** `[get]`

4.119.1 Detailed Description

A wrapper around UnityEngine.Microphone to be able to safely use Microphone and compile for WebGL.

4.120 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.

4.120.1 Detailed Description

Exception thrown if an unsupported codec is encountered.

PhotonVoice currently only supports one Codec, Codec.AudioOpus.

4.120.2 Constructor & Destructor Documentation

4.120.2.1 UnsupportedCodecException()

```
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.

4.121 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.

4.121.1 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.

4.121.2 Constructor & Destructor Documentation

4.121.2.1 UnsupportedSampleTypeException()

```
UnsupportedSampleTypeException (
             Type t )
```

Create a new UnsupportedSampleTypeException.

Parameters

t	The sample type actually encountered.

4.122 OpusCodec.Util Class Reference

4.123 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 **LogRemoteVoiceStates** ()
- void **SetRemoteVoiceDelayFrames** (Codec codec, int delayFrames)
- 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.
- LocalVoice CreateLocalVoiceAudioFromSource (VoiceInfo voiceInfo, IAudioDesc source, AudioSampleType sampleType, IEncoder encoder=null, int channelId=0)

  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.

4.123.1 Detailed Description

Voice client interact with other clients on network via IVoiceTransport.

4.123.2 Member Function Documentation

4.123.2.1 CreateLocalVoice()

```
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.
channel↩ Id	Transport channel specific to transport.
encoder	Encoder producing the stream.

Returns

Outgoing stream handler.

4.123.2.2 CreateLocalVoiceAudioFromSource()

```
LocalVoice CreateLocalVoiceAudioFromSource (
            VoiceInfo voiceInfo,
            IAudioDesc source,
            AudioSampleType sampleType,
            IEncoder encoder = null,
            int channelId = 0 )
```

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.
sampleType	Voice's audio sample type. If does not match source audio sample type, conversion will occur.
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.

4.123.2.3 CreateLocalVoiceFramed< T >()

```
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.
channel↩ Id	Transport channel specific to transport.
encoder	Encoder compressing data stream in pipeline.

Returns

Outgoing stream handler.

4.123.2.4 LocalVoicesInChannel()

```
IEnumerable<LocalVoice> LocalVoicesInChannel (
             int channelId )
```

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

4.123.2.5 RemoteVoiceInfoDelegate()

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

Remote voice info event delegate.

4.123.2.6 RemoveLocalVoice()

```
void RemoveLocalVoice (
             LocalVoice voice )
```

Removes local voice (outgoing data stream).

Parameters

voice	Handler of outgoing stream to be removed.

4.123.2.7 Service()

```
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).

4.123.3 Property Documentation

4.123.3.1 DebugLostPercent

```
int DebugLostPercent [get], [set]
```

Lost frames simulation ratio.

4.123.3.2 FramesLost

```
int FramesLost [get], [set]
```

Lost frames counter.

4.123.3.3 FramesReceived

```
int FramesReceived [get]
```

Received frames counter.

4.123.3.4 FramesSent

```
int FramesSent [get]
```

Sent frames counter.

4.123.3.5 FramesSentBytes

```
int FramesSentBytes  [get]
```

Sent frames bytes counter.

4.123.3.6 LocalVoices

```
IEnumerable<LocalVoice> LocalVoices  [get]
```

Iterates through copy of all local voices list.

4.123.3.7 OnRemoteVoiceInfoAction

```
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);

4.123.3.8 RemoteVoiceInfos

```
IEnumerable<RemoteVoiceInfo> RemoteVoiceInfos  [get]
```

Iterates through all remote voices infos.

4.123.3.9 RoundTripTime

```
int RoundTripTime  [get]
```

Average time required voice packet to return to sender.

4.123.3.10 RoundTripTimeVariance

```
int RoundTripTimeVariance  [get]
```

Average round trip time variation.

4.123.3.11 SuppressInfoDuplicateWarning

```
bool SuppressInfoDuplicateWarning  [get], [set]
```

Do not log warning when duplicate info received.

4.124 VoiceComponent Class Reference

Inherits MonoBehaviour, and ILoggableDependent.

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

Protected Member Functions

- virtual void **Awake** ()

Protected Attributes

- DebugLevel **logLevel** = DebugLevel.INFO

Properties

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

4.125 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
- void InitRecorder (Recorder rec)

  Initializes the Recorder component to be able to transmit audio.
- void SetPlaybackDelaySettings (PlaybackDelaySettings gpds)

  Sets the global configuration for the playback behaviour in case of delays.
- void SetGlobalPlaybackDelaySettings (int low, int high, int max)

  Sets the global configuration for the playback behaviour in case of delays.

Public Attributes

- AppSettings Settings

  Settings to be used by this voice connection
- Func< int, byte, object, Speaker > SpeakerFactory

  Special factory to link Speaker components with incoming remote audio streams
- float MinimalTimeScaleToDispatchInFixedUpdate = -1f

  Configures the minimal Time.timeScale at which Voice client will dispatch incoming messages within LateUpdate.
- 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** ()
- virtual void **Update** ()
- virtual void **FixedUpdate** ()
- void Dispatch ()

  Dispatches incoming network messages for Voice client. Called in FixedUpdate or LateUpdate.
- 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
- Recorder PrimaryRecorder `[get, set]`

Main Recorder to be used for transmission by default
- DebugLevel **GlobalRecordersLogLevel** `[get, set]`
- DebugLevel **GlobalSpeakersLogLevel** `[get, set]`
- int **GlobalPlaybackDelay** `[get, set]`
- string BestRegionSummaryInPreferences `[get, set]`

  Used to store and access the "Best Region Summary" in the Player Preferences.
- int GlobalPlaybackDelayMinSoft `[get]`

  Gets the global value in ms above which the audio player tries to keep the delay.
- int GlobalPlaybackDelayMaxSoft `[get]`

  Gets the global value in ms below which the audio player tries to keep the delay.
- int GlobalPlaybackDelayMaxHard `[get]`

  Gets the global value in ms that audio play delay will not exceed.

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

4.125.1 Detailed Description

Component that represents a client voice connection to Photon Servers.

4.125.2 Member Function Documentation

4.125.2.1 ConnectUsingSettings()

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

Connect to Photon server using Settings

Parameters

overwriteSettings	Overwrites Settings before connecting

Returns

If true voice connection command was sent from client

4.125.2.2 Dispatch()

```
void Dispatch ( )  [protected]
```

Dispatches incoming network messages for Voice client. Called in FixedUpdate or LateUpdate.

It may make sense to dispatch incoming messages, even if the timeScale is near 0. That can be configured with MinimalTimeScaleToDispatchInFixedUpdate.

Without dispatching messages, Voice client won't change state and does not handle updates.

4.125.2.3 InitRecorder()

```
void InitRecorder (
            Recorder rec )
```

Initializes the Recorder component to be able to transmit audio.

Parameters

rec	The Recorder to be initialized.

4.125.2.4 SetGlobalPlaybackDelaySettings()

```
void SetGlobalPlaybackDelaySettings (
            int low,
            int high,
            int max )
```

Sets the global configuration for the playback behaviour in case of delays.

Parameters

low	In milliseconds, audio player tries to keep the playback delay above this value.
high	In milliseconds, audio player tries to keep the playback below above this value.
max	In milliseconds, audio player guarantees that the playback delay never exceeds this value.

4.125.2.5 SetPlaybackDelaySettings()

```
void SetPlaybackDelaySettings (
            PlaybackDelaySettings gpds )
```

Sets the global configuration for the playback behaviour in case of delays.

Parameters

gpds	Playback delay configuration struct.

4.125.3 Member Data Documentation

4.125.3.1 AutoCreateSpeakerIfNotFound

```
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

4.125.3.2 MinimalTimeScaleToDispatchInFixedUpdate

```
float MinimalTimeScaleToDispatchInFixedUpdate = -1f
```

Configures the minimal Time.timeScale at which Voice client will dispatch incoming messages within LateUpdate.

It may make sense to dispatch incoming messages, even if the timeScale is near 0. In some cases, stopping the game time makes sense, so this option defaults to -1f, which is "off". Without dispatching messages, Voice client won't change state and does not handle updates.

4.125.3.3 Settings

```
AppSettings Settings
```

Settings to be used by this voice connection

4.125.3.4 SpeakerFactory

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

Special factory to link Speaker components with incoming remote audio streams

4.125.4 Property Documentation

4.125.4.1 BestRegionSummaryInPreferences

```
string BestRegionSummaryInPreferences  [get], [set]
```

Used to store and access the "Best Region Summary" in the Player Preferences.

4.125.4.2 ClientState

```
ClientState ClientState  [get]
```

Returns Photon Voice client state.

4.125.4.3 FramesLostPercent

```
float FramesLostPercent  [get]
```

Percentage of lost frames.

4.125.4.4 FramesLostPerSecond

```
float FramesLostPerSecond  [get]
```

Number of frames lost per second.

4.125.4.5 FramesReceivedPerSecond

```
float FramesReceivedPerSecond  [get]
```

Number of frames received per second.

4.125.4.6 GlobalPlaybackDelayMaxHard

```
int GlobalPlaybackDelayMaxHard  [get]
```

Gets the global value in ms that audio play delay will not exceed.

4.125.4.7 GlobalPlaybackDelayMaxSoft

```
int GlobalPlaybackDelayMaxSoft  [get]
```

Gets the global value in ms below which the audio player tries to keep the delay.

4.125.4.8 GlobalPlaybackDelayMinSoft

```
int GlobalPlaybackDelayMinSoft  [get]
```

Gets the global value in ms above which the audio player tries to keep the delay.

4.125.4.9 Logger

```
VoiceLogger Logger  [get], [protected set]
```

Logger used by this component

4.125.4.10 LogLevel

```
DebugLevel LogLevel  [get], [set]
```

Log level for this component

4.125.4.11 PrimaryRecorder

```
Recorder PrimaryRecorder  [get], [set]
```

Main Recorder to be used for transmission by default

4.125.4.12 SpeakerPrefab

```
GameObject SpeakerPrefab  [get], [set]
```

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

4.125.4.13 VoiceClient

`VoiceClient VoiceClient [get]`

Returns underlying Photon Voice client.

4.125.5 Event Documentation

4.125.5.1 RemoteVoiceAdded

`Action<RemoteVoiceLink> RemoteVoiceAdded`

Fires when a remote voice stream is added

4.125.5.2 SpeakerLinked

`Action<Speaker> SpeakerLinked`

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

4.126 VoiceDebugScript Class Reference

Utility script to be attached next to PhotonVoiceView & PhotonView on the player prefab to be network instantiated. Call voiceDebugScript.CantHearYou() on the networked object of the remote (or local) player if you can't hear the corresponding player.

Inherits MonoBehaviourPun.

Public Member Functions

- void **CantHearYou** ()

Public Attributes

- bool ForceRecordingAndTransmission

  Make sure recorder.TransmitEnabled and recorder.IsRecording are true.
- AudioClip TestAudioClip

  Audio file to be broadcast when TestUsingAudioClip is enabled.
- bool TestUsingAudioClip

  Broadcast Audio file to make sure transmission over network works if microphone (audio input device/hardware) is not reliable. Requires setting AudioClip in TestAudioClip.
- bool DisableVad

  Disable recorder.VoiceDetection for easier testing.
- bool IncreaseLogLevels

  Set main voice component's log level to ALL (max).
- bool LocalDebug

  Debug DebugEcho mode (Can't Hear My Self?!).

4.126.1 Detailed Description

Utility script to be attached next to PhotonVoiceView & PhotonView on the player prefab to be network instantiated. Call voiceDebugScript.CantHearYou() on the networked object of the remote (or local) player if you can't hear the corresponding player.

4.126.2 Member Data Documentation

4.126.2.1 DisableVad

```
bool DisableVad
```

Disable recorder.VoiceDetection for easier testing.

4.126.2.2 ForceRecordingAndTransmission

```
bool ForceRecordingAndTransmission
```

Make sure recorder.TransmitEnabled and recorder.IsRecording are true.

4.126.2.3 IncreaseLogLevels

```
bool IncreaseLogLevels
```

Set main voice component's log level to ALL (max).

4.126.2.4 LocalDebug

```
bool LocalDebug
```

Debug DebugEcho mode (Can't Hear My Self?!).

4.126.2.5 TestAudioClip

```
AudioClip TestAudioClip
```

Audio file to be broadcast when TestUsingAudioClip is enabled.

4.126.2.6 TestUsingAudioClip

```
bool TestUsingAudioClip
```

Broadcast Audio file to make sure transmission over network works if microphone (audio input device/hardware) is not reliable. Requires setting AudioClip in TestAudioClip.

4.127 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

- float **norm**
- float **threshold**
- 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.

4.127.1 Detailed Description

Simple voice activity detector triggered by signal level.

4.127.2 Member Function Documentation

4.127.2.1 Process()

```
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 or null if frame has been discarded (VAD)

Implements IProcessor< T >.

4.127.3 Property Documentation

4.127.3.1 ActivityDelayMs

```
int ActivityDelayMs  [get], [set]
```

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

4.127.3.2 Detected

```
bool Detected  [get], [protected set]
```

If true, voice detected.

4.127.3.3 DetectedTime

```
DateTime DetectedTime [get]
```

Last time when switched to detected state.

4.127.3.4 On

```
bool On [get], [set]
```

If true, voice detection enabled.

4.127.3.5 Threshold

```
float Threshold [get], [set]
```

Voice detected as soon as signal level exceeds threshold.

4.127.4 Event Documentation

4.127.4.1 OnDetected

```
Action OnDetected
```

Called when switched to detected state.

4.128 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]`

4.128.1 Detailed Description

Calibration Utility for Voice Detector

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

4.128.2 Constructor & Destructor Documentation

4.128.2.1 VoiceDetectorCalibration()

```
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.

4.128.3 Member Function Documentation

4.128.3.1 Calibrate()

```
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.

4.128.3.2 Process()

```
T [] Process (
            T[] buf )
```

Process a frame of audio data.

Parameters

buf	Buffer containing input audio data

Returns

Buffer containing output audio data or null if frame has been discarded (VAD)

Implements IProcessor< T >.

4.129 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

4.129.1 Detailed Description

Dummy VoiceDetector that doesn't actually do anything.

4.130 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

4.130.1 Detailed Description

VoiceDetector specialization for float audio.

4.130.2 Constructor & Destructor Documentation

4.130.2.1 VoiceDetectorFloat()

```
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.

4.131 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

4.131.1 Detailed Description

VoiceDetector specialization for float audio.

4.131.2 Constructor & Destructor Documentation

4.131.2.1 VoiceDetectorShort()

```
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.

4.132 VoiceEvent Class Reference

Static Public Attributes

- const byte Code = 202

  Single event used for voice communications.
- const byte **FrameCode** = 203

4.132.1 Member Data Documentation

4.132.1.1 Code

```
const byte Code = 202  [static]
```

Single event used for voice communications.

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

4.133 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.
- static VoiceInfo CreateAudio (Codec codec, int samplingRate, int channels, int frameDurationUs, object user-
data=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]`

    Number of channels.
- int FrameDurationUs `[get, set]`

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

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

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

    Video height
- int FPS `[get, set]`

    Video frames per second
- int KeyFrameInt `[get, set]`

    Video keyframe interval in frames
- 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.

4.133.1 Detailed Description

Describes stream properties.

4.133.2 Member Function Documentation

4.133.2.1 CreateAudio()

```
static VoiceInfo CreateAudio (
            Codec codec,
            int samplingRate,
            int channels,
            int frameDurationUs,
            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.

4.133.2.2 CreateAudioOpus()

```
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.

4.133.3 Property Documentation

4.133.3.1 Bitrate

```
int Bitrate  [get], [set]
```

Target bitrate (in bits/second).

4.133.3.2 Channels

```
int Channels  [get], [set]
```

Number of channels.

4.133.3.3 FPS

```
int FPS  [get], [set]
```

Video frames per second

4.133.3.4 FrameDurationSamples

```
int FrameDurationSamples  [get]
```

Uncompressed frame (data packet) size in samples.

4.133.3.5 FrameDurationUs

```
int FrameDurationUs  [get], [set]
```

Uncompressed frame (audio packet) size in microseconds.

4.133.3.6 FrameSize

```
int FrameSize  [get]
```

Uncompressed frame (data packet) array size.

4.133.3.7 Height

```
int Height  [get], [set]
```

Video height

4.133.3.8 KeyFrameInt

```
int KeyFrameInt  [get], [set]
```

Video keyframe interval in frames

4.133.3.9 SamplingRate

```
int SamplingRate  [get], [set]
```

Audio sampling rate (frequency, in Hz).

4.133.3.10 UserData

```
object UserData  [get], [set]
```

Optional user data. Should be serializable by Photon.

4.133.3.11 Width

```
int Width  [get], [set]
```

Video width.

4.134 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]`

4.134.1 Detailed Description

Utility Audio Processor Voice Detection Calibration.

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

4.134.2 Constructor & Destructor Documentation

4.134.2.1 VoiceLevelDetectCalibrate()

```
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.

4.134.3 Member Function Documentation

4.134.3.1 Calibrate()

```
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.

4.134.3.2 Process()

```
T [] Process (
            T[] buf )
```

Process a frame of audio data.

Parameters

buf	Buffer containing input audio data

Returns

Buffer containing output audio data or null if frame has been discarded (VAD)

Implements IProcessor< T >.

4.134.4 Property Documentation

4.134.4.1 LevelMeter

```
ILevelMeter LevelMeter  [get]
```

The LevelMeter in use.

4.134.4.2 VoiceDetector

```
IVoiceDetector VoiceDetector [get]
```

The VoiceDetector in use

4.135 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]`

4.136 WaveFormat Class Reference

Defines the format of waveform-audio data.

Inherits ICloneable, and IEquatable< WaveFormat >.

Inherited by WaveFormatExtensible.

Public Member Functions

- WaveFormat ()

  Initializes a new instance of the WaveFormat class with a sample rate of 44100 Hz, bits per sample of 16 bit, 2 channels and PCM as the format type.
- WaveFormat (int sampleRate, int bits, int channels)

  Initializes a new instance of the WaveFormat class with PCM as the format type.
- WaveFormat (int sampleRate, int bits, int channels, AudioEncoding encoding)

  Initializes a new instance of the WaveFormat class.
- WaveFormat (int sampleRate, int bits, int channels, AudioEncoding encoding, int extraSize)

  Initializes a new instance of the WaveFormat class.
- long MillisecondsToBytes (double milliseconds)

  Converts a duration in milliseconds to a duration in bytes.
- double BytesToMilliseconds (long bytes)

  Converts a duration in bytes to a duration in milliseconds.
- virtual bool Equals (WaveFormat other)

  Indicates whether the current object is equal to another object of the same type.
- override string ToString ()

  Returns a string which describes the WaveFormat.
- virtual object Clone ()

  Creates a new WaveFormat object that is a copy of the current instance.

Protected Member Functions

- virtual void UpdateProperties ()

  Updates the BlockAlign- and the BytesPerSecond-property.

Properties

- virtual int Channels `[get, set]`

  Gets the number of channels in the waveform-audio data. Mono data uses one channel and stereo data uses two channels.
- virtual int SampleRate `[get, set]`

  Gets the sample rate, in samples per second (hertz).
- virtual int BytesPerSecond `[get, set]`

  Gets the required average data transfer rate, in bytes per second. For example, 16-bit stereo at 44.1 kHz has an average data rate of 176,400 bytes per second (2 channels — 2 bytes per sample per channel — 44,100 samples per second).
- virtual int BlockAlign `[get, set]`

  Gets the block alignment, in bytes. The block alignment is the minimum atomic unit of data. For PCM data, the block alignment is the number of bytes used by a single sample, including data for both channels if the data is stereo. For example, the block alignment for 16-bit stereo PCM is 4 bytes (2 channels x 2 bytes per sample).
- virtual int BitsPerSample `[get, set]`

  Gets the number of bits, used to store one sample.
- virtual int ExtraSize `[get, set]`

  Gets the size (in bytes) of extra information. This value is mainly used for marshalling.
- virtual int BytesPerSample `[get]`

  Gets the number of bytes, used to store one sample.
- virtual int BytesPerBlock `[get]`

  Gets the number of bytes, used to store one block. This value equals BytesPerSample multiplied with Channels.
- virtual AudioEncoding WaveFormatTag `[get, set]`

  Gets the waveform-audio format type.

4.136.1 Detailed Description

Defines the format of waveform-audio data.

4.136.2 Constructor & Destructor Documentation

4.136.2.1 WaveFormat() `[1/4]`

```
WaveFormat ( )
```

Initializes a new instance of the WaveFormat class with a sample rate of 44100 Hz, bits per sample of 16 bit, 2 channels and PCM as the format type.

4.136.2.2 WaveFormat() `[2/4]`

```
WaveFormat (
             int sampleRate,
             int bits,
             int channels )
```

Initializes a new instance of the WaveFormat class with PCM as the format type.

Parameters

sampleRate	Samples per second.
bits	Number of bits, used to store one sample.
channels	Number of channels in the waveform-audio data.

4.136.2.3 WaveFormat() `[3/4]`

```
WaveFormat (
             int sampleRate,
             int bits,
             int channels,
             AudioEncoding encoding )
```

Initializes a new instance of the WaveFormat class.

Parameters

sampleRate	Samples per second.
bits	Number of bits, used to store one sample.
channels	Number of channels in the waveform-audio data.
encoding	Format type or encoding of the wave format.

4.136.2.4 WaveFormat() [4/4]

```
WaveFormat (
             int sampleRate,
             int bits,
             int channels,
             AudioEncoding encoding,
             int extraSize )
```

Initializes a new instance of the WaveFormat class.

Parameters

sampleRate	Samples per second.
bits	Number of bits, used to store one sample.
channels	Number of channels in the waveform-audio data.
encoding	Format type or encoding of the wave format.
extraSize	Size (in bytes) of extra information. This value is mainly used for marshalling.

4.136.3 Member Function Documentation

4.136.3.1 BytesToMilliseconds()

```
double BytesToMilliseconds (
             long bytes )
```

Converts a duration in bytes to a duration in milliseconds.

Parameters

bytes	Duration in bytes to convert to a duration in milliseconds.

Returns

Duration in milliseconds.

4.136.3.2 Clone()

```
virtual object Clone ( )  [virtual]
```

Creates a new WaveFormat object that is a copy of the current instance.

Returns

A copy of the current instance.

Reimplemented in WaveFormatExtensible.

4.136.3.3 Equals()

```
virtual bool Equals (
             WaveFormat other )  [virtual]
```

Indicates whether the current object is equal to another object of the same type.

Parameters

other	The WaveFormat to compare with this WaveFormat.

Returns

true if the current object is equal to the other parameter; otherwise, false.

4.136.3.4 MillisecondsToBytes()

```
long MillisecondsToBytes (
             double milliseconds )
```

Converts a duration in milliseconds to a duration in bytes.

Parameters

milliseconds	Duration in millisecond to convert to a duration in bytes.

Returns

Duration in bytes.

4.136.3.5 ToString()

```
override string ToString ( )
```

Returns a string which describes the WaveFormat.

Returns

A string which describes the WaveFormat.

4.136.3.6 UpdateProperties()

```
virtual void UpdateProperties ( )  [protected], [virtual]
```

Updates the BlockAlign- and the BytesPerSecond-property.

4.136.4 Property Documentation

4.136.4.1 BitsPerSample

```
virtual int BitsPerSample  [get], [set]
```

Gets the number of bits, used to store one sample.

4.136.4.2 BlockAlign

```
virtual int BlockAlign  [get], [set]
```

Gets the block alignment, in bytes. The block alignment is the minimum atomic unit of data. For PCM data, the block alignment is the number of bytes used by a single sample, including data for both channels if the data is stereo. For example, the block alignment for 16-bit stereo PCM is 4 bytes (2 channels x 2 bytes per sample).

4.136.4.3 BytesPerBlock

```
virtual int BytesPerBlock  [get]
```

Gets the number of bytes, used to store one block. This value equals BytesPerSample multiplied with Channels.

4.136.4.4 BytesPerSample

```
virtual int BytesPerSample  [get]
```

Gets the number of bytes, used to store one sample.

4.136.4.5 BytesPerSecond

```
virtual int BytesPerSecond  [get], [set]
```

Gets the required average data transfer rate, in bytes per second. For example, 16-bit stereo at 44.1 kHz has an average data rate of 176,400 bytes per second (2 channels — 2 bytes per sample per channel — 44,100 samples per second).

4.136.4.6 Channels

```
virtual int Channels  [get], [set]
```

Gets the number of channels in the waveform-audio data. Mono data uses one channel and stereo data uses two channels.

4.136.4.7 ExtraSize

```
virtual int ExtraSize  [get], [set]
```

Gets the size (in bytes) of extra information. This value is mainly used for marshalling.

4.136.4.8 SampleRate

```
virtual int SampleRate  [get], [set]
```

Gets the sample rate, in samples per second (hertz).

4.136.4.9 WaveFormatTag

```
virtual AudioEncoding WaveFormatTag  [get], [set]
```

Gets the waveform-audio format type.

4.137 WaveFormatExtensible Class Reference

Defines the format of waveform-audio data for formats having more than two channels or higher sample resolutions than allowed by WaveFormat. Can be used to define any format that can be defined by WaveFormat. For more information see and .

Inherits WaveFormat.

Public Member Functions

- WaveFormatExtensible (int sampleRate, int bits, int channels, Guid subFormat)

  Initializes a new instance of the WaveFormatExtensible class.
- WaveFormatExtensible (int sampleRate, int bits, int channels, Guid subFormat, ChannelMask channelMask)

  Initializes a new instance of the WaveFormatExtensible class.
- WaveFormat ToWaveFormat ()

  Converts the WaveFormatExtensible instance to a raw WaveFormat instance by converting the SubFormat to the equal WaveFormat.WaveFormatTag.
- override object Clone ()

  Creates a new WaveFormat object that is a copy of the current instance.
- override string ToString ()

  Returns a string which describes the WaveFormatExtensible.

Static Public Member Functions

- static Guid SubTypeFromWaveFormat (WaveFormat waveFormat)

  Returns the SubType-Guid of a waveFormat . If the specified waveFormat does not contain a SubType-Guid, the WaveFormat.WaveFormatTag gets converted to the equal SubType-Guid using the AudioSubTypes.SubTypeFromEncoding method.

Properties

- int ValidBitsPerSample `[get, protected set]`

  Gets the number of bits of precision in the signal. Usually equal to WaveFormat.BitsPerSample. However, WaveFormat.BitsPerSample is the container size and must be a multiple of 8, whereas ValidBitsPerSample can be any value not exceeding the container size. For example, if the format uses 20-bit samples, WaveFormat.BitsPerSample must be at least 24, but ValidBitsPerSample is 20.
- int SamplesPerBlock `[get, protected set]`

  Gets the number of samples contained in one compressed block of audio data. This value is used in buffer estimation. This value is used with compressed formats that have a fixed number of samples within each block. This value can be set to 0 if a variable number of samples is contained in each block of compressed audio data. In this case, buffer estimation and position information needs to be obtained in other ways.
- ChannelMask ChannelMask `[get, protected set]`

  Gets a bitmask specifying the assignment of channels in the stream to speaker positions.
- Guid SubFormat `[get, protected set]`

  Subformat of the data, such as AudioSubTypes.Pcm. The subformat information is similar to that provided by the tag in the WaveFormat class's WaveFormat.WaveFormatTag member.

Additional Inherited Members

4.137.1 Detailed Description

Defines the format of waveform-audio data for formats having more than two channels or higher sample resolutions than allowed by WaveFormat. Can be used to define any format that can be defined by WaveFormat. For more information see and .

4.137.2 Constructor & Destructor Documentation

4.137.2.1 WaveFormatExtensible() [1/2]

```
WaveFormatExtensible (
            int sampleRate,
            int bits,
            int channels,
            Guid subFormat )
```

Initializes a new instance of the WaveFormatExtensible class.

Parameters

sampleRate	Samplerate of the waveform-audio. This value will get applied to the WaveFormat.SampleRate property.
bits	Bits per sample of the waveform-audio. This value will get applied to the WaveFormat.BitsPerSample property and the ValidBitsPerSample property.
channels	Number of channels of the waveform-audio. This value will get applied to the WaveFormat.Channels property.
subFormat	Subformat of the data. This value will get applied to the SubFormat property.

4.137.2.2 WaveFormatExtensible() [2/2]

```
WaveFormatExtensible (
            int sampleRate,
            int bits,
            int channels,
            Guid subFormat,
            ChannelMask channelMask )
```

Initializes a new instance of the WaveFormatExtensible class.

Parameters

sampleRate	Samplerate of the waveform-audio. This value will get applied to the WaveFormat.SampleRate property.
bits	Bits per sample of the waveform-audio. This value will get applied to the WaveFormat.BitsPerSample property and the ValidBitsPerSample property.
channels	Number of channels of the waveform-audio. This value will get applied to the WaveFormat.Channels property.
subFormat	Subformat of the data. This value will get applied to the SubFormat property.
channelMask	Bitmask specifying the assignment of channels in the stream to speaker positions. Thie value will get applied to the ChannelMask property.

4.137.3 Member Function Documentation

4.137.3.1 Clone()

```
override object Clone ( )  [virtual]
```

Creates a new WaveFormat object that is a copy of the current instance.

Returns

A copy of the current instance.

Reimplemented from WaveFormat.

4.137.3.2 SubTypeFromWaveFormat()

```
static Guid SubTypeFromWaveFormat (
            WaveFormat waveFormat )  [static]
```

Returns the SubType-Guid of a *waveFormat* . If the specified *waveFormat* does not contain a SubType-Guid, the WaveFormat.WaveFormatTag gets converted to the equal SubType-Guid using the AudioSubTypes.SubTypeFromEncoding method.

Parameters

waveFormat	WaveFormat which gets used to determine the SubType-Guid.

Returns

SubType-Guid of the specified *waveFormat* .

4.137.3.3 ToString()

```
override string ToString ( )
```

Returns a string which describes the WaveFormatExtensible.

Returns

A string which describes the WaveFormatExtensible.

4.137.3.4 ToWaveFormat()

```
WaveFormat ToWaveFormat ( )
```

Converts the WaveFormatExtensible instance to a raw WaveFormat instance by converting the SubFormat to the equal WaveFormat.WaveFormatTag.

Returns

A simple WaveFormat instance.

4.137.4 Property Documentation

4.137.4.1 ChannelMask

```
ChannelMask ChannelMask  [get], [protected set]
```

Gets a bitmask specifying the assignment of channels in the stream to speaker positions.

4.137.4.2 SamplesPerBlock

```
int SamplesPerBlock  [get], [protected set]
```

Gets the number of samples contained in one compressed block of audio data. This value is used in buffer estimation. This value is used with compressed formats that have a fixed number of samples within each block. This value can be set to 0 if a variable number of samples is contained in each block of compressed audio data. In this case, buffer estimation and position information needs to be obtained in other ways.

4.137.4.3 SubFormat

```
Guid SubFormat  [get], [protected set]
```

Subformat of the data, such as AudioSubTypes.Pcm. The subformat information is similar to that provided by the tag in the WaveFormat class's WaveFormat.WaveFormatTag member.

4.137.4.4 ValidBitsPerSample

```
int ValidBitsPerSample  [get], [protected set]
```

Gets the number of bits of precision in the signal. Usually equal to WaveFormat.BitsPerSample. However, WaveFormat.BitsPerSample is the container size and must be a multiple of 8, whereas ValidBitsPerSample can be any value not exceeding the container size. For example, if the format uses 20-bit samples, WaveFormat.BitsPerSample must be at least 24, but ValidBitsPerSample is 20.

4.138 WaveWriter Class Reference

Encoder for wave files.

Inherits IDisposable, and IWriteable.

Public Member Functions

- WaveWriter (string fileName, WaveFormat waveFormat)

    Initializes a new instance of the WaveWriter class.
- WaveWriter (Stream stream, WaveFormat waveFormat)

    Initializes a new instance of the WaveWriter class.
- void Dispose ()

    Disposes the WaveWriter and writes down the wave header.
- void WriteSample (float sample)

    Encodes a single sample.
- void WriteSamples (float[ ] samples, int offset, int count)

    Encodes multiple samples.
- void Write (byte[ ] buffer, int offset, int count)

    Encodes raw data in the form of a byte array.
- void Write (byte value)

    Writes down a single byte.
- void Write (short value)

    Writes down a single 16 bit integer value.
- void Write (int value)

    Writes down a single 32 bit integer value.
- void Write (float value)

    Writes down a single 32 bit float value.

Static Public Member Functions

- static void WriteToFile (string filename, IWaveSource source, bool deleteFileIfAlreadyExists, int maxlength=-1)

    Writes down all audio data of the IWaveSource to a file.

Protected Member Functions

- virtual void Dispose (bool disposing)

    Disposes the WaveWriter and writes down the wave header.

Properties

- bool IsDisposed `[get]`

    Signals if the object has already been disposed
- bool IsDisposing `[get]`

    Signals if the object is in a disposing state

4.138.1 Detailed Description

Encoder for wave files.

4.138.2 Constructor & Destructor Documentation

4.138.2.1 WaveWriter() `[1/2]`

```
WaveWriter (
            string fileName,
            WaveFormat waveFormat )
```

Initializes a new instance of the WaveWriter class.

Parameters

fileName	Filename of the destination file. This filename should typically end with the .wav extension.
waveFormat	Format of the waveform-audio data. Note that the WaveWriter won't convert any data.

4.138.2.2 WaveWriter() `[2/2]`

```
WaveWriter (
            Stream stream,
            WaveFormat waveFormat )
```

Initializes a new instance of the WaveWriter class.

Parameters

stream	Destination stream which should be used to store the
waveFormat	Format of the waveform-audio data. Note that the WaveWriter won't convert any data.

4.138.3 Member Function Documentation

4.138.3.1 Dispose() `[1/2]`

```
void Dispose ( )
```

Disposes the WaveWriter and writes down the wave header.

4.138.3.2 Dispose() `[2/2]`

```
virtual void Dispose (
            bool disposing )  [protected], [virtual]
```

Disposes the WaveWriter and writes down the wave header.

Parameters

disposing	True to release both managed and unmanaged resources; false to release only unmanaged resources.

4.138.3.3 Write() [1/5]

```
void Write (
            byte value )
```

Writes down a single byte.

Parameters

value	Byte to write down.

4.138.3.4 Write() [2/5]

```
void Write (
            byte[] buffer,
            int offset,
            int count )
```

Encodes raw data in the form of a byte array.

Parameters

buffer	Byte array which contains the data to encode.
offset	Zero-based offset in the *buffer* .
count	Number of bytes to encode.

Implements IWriteable.

4.138.3.5 Write() [3/5]

```
void Write (
            float value )
```

Writes down a single 32 bit float value.

Parameters

value	Value to write down.

4.138.3.6 Write() [4/5]

```
void Write (
             int value )
```

Writes down a single 32 bit integer value.

Parameters

value	Value to write down.

4.138.3.7 Write() [5/5]

```
void Write (
             short value )
```

Writes down a single 16 bit integer value.

Parameters

value	Value to write down.

4.138.3.8 WriteSample()

```
void WriteSample (
             float sample )
```

Encodes a single sample.

Parameters

sample	The sample to encode.

4.138.3.9 WriteSamples()

```
void WriteSamples (
            float[] samples,
            int offset,
            int count )
```

Encodes multiple samples.

Parameters

samples	Float array which contains the samples to encode.
offset	Zero-based offset in the *samples* array.
count	Number of samples to encode.

4.138.3.10 WriteToFile()

```
static void WriteToFile (
            string filename,
            IWaveSource source,
            bool deleteFileIfAlreadyExists,
            int maxlength = -1 )  [static]
```

Writes down all audio data of the IWaveSource to a file.

Parameters

filename	The filename.
source	The source to write down to the file.
deleteFileIfAlreadyExists	if set to `true` the file will be overritten if it already exists.
maxlength	The maximum number of bytes to write. Use -1 to write an infinte number of bytes.

This method is obsolete. Use the Extensions.WriteToWaveStream extension instead.

4.138.4 Property Documentation

4.138.4.1 IsDisposed

```
bool IsDisposed  [get]
```

Signals if the object has already been disposed

4.138.4.2 IsDisposing

```
bool IsDisposing  [get]
```

Signals if the object is in a disposing state

4.139 WebRtcAudioDsp Class Reference

Inherits VoiceComponent.

Public Member Functions

- bool SetOrSwitchAudioListener (AudioListener audioListener)

  Set the AudioListener to be used with this WebRtcAudioDsp
- bool SetOrSwitchAudioOutCapture (AudioOutCapture audioOutCapture)

  Set the AudioOutCapture to be used with this WebRtcAudioDsp

Public Attributes

- bool **AECMobileComfortNoise**

Protected Member Functions

- override void **Awake** ()

Properties

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

Additional Inherited Members

4.139.1 Member Function Documentation

4.139.1.1 SetOrSwitchAudioListener()

```
bool SetOrSwitchAudioListener (
            AudioListener audioListener )
```

Set the AudioListener to be used with this WebRtcAudioDsp

Parameters

audioListener	The audioListener to be used

Returns

Success or failure

4.139.1.2 SetOrSwitchAudioOutCapture()

```
bool SetOrSwitchAudioOutCapture (
            AudioOutCapture audioOutCapture )
```

Set the AudioOutCapture to be used with this WebRtcAudioDsp

Parameters

audioOutCapture	The audioOutCapture to be used

Returns

Success or failure

4.140 WebRTCAudioLib Class Reference

Inherited by WebRTCAudioProcessor.

Public Types

- enum **Error**
- enum **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_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)

4.141 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** ()

Static Public Attributes

- static readonly int[ ] **SupportedSamplingRates** = { 8000, 16000, 32000, 48000 }

Properties

- int **AECStreamDelayMs** `[set]`
- bool?? **AEC** `[set]`
- bool? **AECHighPass** `[set]`
- bool?? **AECMobile** `[set]`
- bool? **HighPass** `[set]`
- bool? **NoiseSuppression** `[set]`
- bool? **AGC** `[set]`
- int **AGCCompressionGain** `[set]`
- int **AGCTargetLevel** `[set]`
- bool? **AGC2** `[set]`
- bool? **VAD** `[set]`
- bool **Bypass** `[set]`

Additional Inherited Members

Index