

Photon Voice

v2.1.3

Generated by Doxygen 1.8.10

Fri Dec 7 2018 13:57:15



# Contents

<b>1</b>	<b>Photon Voice Doxygen Readme</b>	<b>1</b>
<b>2</b>	<b>Namespace Documentation</b>	<b>3</b>
2.1	Photon Namespace Reference . . . . .	3
2.2	Photon.Voice Namespace Reference . . . . .	3
2.2.1	Enumeration Type Documentation . . . . .	6
2.2.1.1	Codec . . . . .	6
2.3	Photon.Voice.PUN Namespace Reference . . . . .	6
2.4	Photon.Voice.Unity Namespace Reference . . . . .	6
2.5	Photon.Voice.Unity.UtilityScripts Namespace Reference . . . . .	7
2.6	POpusCodec Namespace Reference . . . . .	7
2.7	POpusCodec.Enums Namespace Reference . . . . .	7
2.7.1	Enumeration Type Documentation . . . . .	7
2.7.1.1	Bandwidth . . . . .	7
2.7.1.2	Channels . . . . .	8
2.7.1.3	Delay . . . . .	8
2.7.1.4	OpusApplicationType . . . . .	8
2.7.1.5	SignalHint . . . . .	8
<b>3</b>	<b>Class Documentation</b>	<b>9</b>
3.1	SpeexProcessor.AECLatencyResultType Struct Reference . . . . .	9
3.2	AudioClipWrapper Class Reference . . . . .	9
3.3	AudioDesc Class Reference . . . . .	9
3.4	AudioInEnumerator Class Reference . . . . .	10
3.5	AudioOutCapture Class Reference . . . . .	10
3.6	AudioStreamPlayer Class Reference . . . . .	10
3.7	AudioUtil Class Reference . . . . .	11
3.7.1	Detailed Description . . . . .	12
3.7.2	Member Function Documentation . . . . .	12
3.7.2.1	Convert(float[] src, short[] dst, int dstCount) . . . . .	12
3.7.2.2	Convert(short[] src, float[] dst, int dstCount) . . . . .	12
3.7.2.3	ForceToStereo< T >(T[] src, T[] dst, int srcChannels) . . . . .	12

3.7.2.4	Resample< T >(T[] src, T[] dst, int dstCount, int channels)	12
3.7.2.5	ResampleAndConvert(short[] src, float[] dst, int dstCount, int channels)	13
3.7.2.6	ResampleAndConvert(float[] src, short[] dst, int dstCount, int channels)	13
3.8	BufferReaderPushAdapter< T > Class Template Reference	13
3.8.1	Detailed Description	14
3.8.2	Constructor & Destructor Documentation	14
3.8.2.1	BufferReaderPushAdapter(LocalVoice localVoice, IDataReader< T > reader)	14
3.8.3	Member Function Documentation	14
3.8.3.1	Service(LocalVoice localVoice)	14
3.9	BufferReaderPushAdapterAsyncPool< T > Class Template Reference	14
3.9.1	Detailed Description	14
3.9.2	Constructor & Destructor Documentation	14
3.9.2.1	BufferReaderPushAdapterAsyncPool(LocalVoice localVoice, IDataReader< T > reader)	15
3.9.3	Member Function Documentation	16
3.9.3.1	Service(LocalVoice localVoice)	16
3.10	BufferReaderPushAdapterAsyncPoolCopy< T > Class Template Reference	16
3.10.1	Detailed Description	16
3.10.2	Constructor & Destructor Documentation	16
3.10.2.1	BufferReaderPushAdapterAsyncPoolCopy(LocalVoice localVoice, IDataReader< T > reader)	16
3.10.3	Member Function Documentation	17
3.10.3.1	Service(LocalVoice localVoice)	17
3.11	BufferReaderPushAdapterAsyncPoolFloatToShort Class Reference	17
3.11.1	Detailed Description	17
3.11.2	Constructor & Destructor Documentation	17
3.11.2.1	BufferReaderPushAdapterAsyncPoolFloatToShort(Voice.LocalVoice localVoice, Voice.IDataReader< float > reader)	17
3.11.3	Member Function Documentation	18
3.11.3.1	Service(Voice.LocalVoice localVoice)	18
3.12	BufferReaderPushAdapterBase< T > Class Template Reference	18
3.12.1	Detailed Description	18
3.12.2	Constructor & Destructor Documentation	18
3.12.2.1	BufferReaderPushAdapterBase(IDataReader< T > reader)	18
3.12.3	Member Function Documentation	19
3.12.3.1	Dispose()	19
3.12.3.2	Service(LocalVoice localVoice)	19
3.13	WebRTCAudioLib.ConfigParam Struct Reference	19
3.14	ConnectAndJoin Class Reference	19
3.15	OpusCodec.Decoder Class Reference	20
3.15.1	Member Function Documentation	20

3.15.1.1	<a href="#">DecodeToByte(byte[] buf)</a>	20
3.15.1.2	<a href="#">DecodeToFloat(byte[] buf)</a>	20
3.15.1.3	<a href="#">DecodeToShort(byte[] buf)</a>	21
3.15.1.4	<a href="#">Open(VoiceInfo i)</a>	21
3.16	<a href="#">OpusCodec.Encoder&lt; T &gt; Class Template Reference</a>	21
3.16.1	<a href="#">Member Function Documentation</a>	21
3.16.1.1	<a href="#">EncodeAndGetOutput(T[] buf)</a>	21
3.17	<a href="#">OpusCodec.EncoderFactory Class Reference</a>	22
3.18	<a href="#">OpusCodec.EncoderFloat Class Reference</a>	22
3.19	<a href="#">OpusCodec.EncoderShort Class Reference</a>	22
3.20	<a href="#">FactoryPrimitiveArrayPool&lt; T &gt; Class Template Reference</a>	22
3.20.1	<a href="#">Detailed Description</a>	23
3.21	<a href="#">FactoryReusableArray&lt; T &gt; Class Template Reference</a>	23
3.21.1	<a href="#">Detailed Description</a>	23
3.22	<a href="#">Framer&lt; T &gt; Class Template Reference</a>	23
3.22.1	<a href="#">Detailed Description</a>	24
3.22.2	<a href="#">Constructor &amp; Destructor Documentation</a>	24
3.22.2.1	<a href="#">Framer(int frameSize)</a>	24
3.22.3	<a href="#">Member Function Documentation</a>	24
3.22.3.1	<a href="#">Count(int bufLen)</a>	24
3.22.3.2	<a href="#">Frame(T[] buf)</a>	24
3.23	<a href="#">IAudioDesc Interface Reference</a>	24
3.23.1	<a href="#">Detailed Description</a>	25
3.23.2	<a href="#">Property Documentation</a>	25
3.23.2.1	<a href="#">Channels</a>	25
3.23.2.2	<a href="#">Error</a>	25
3.23.2.3	<a href="#">SamplingRate</a>	25
3.24	<a href="#">IAudioOut Interface Reference</a>	25
3.25	<a href="#">IAudioPusher&lt; T &gt; Interface Template Reference</a>	25
3.25.1	<a href="#">Detailed Description</a>	26
3.25.2	<a href="#">Member Function Documentation</a>	26
3.25.2.1	<a href="#">SetCallback(Action&lt; T[]&gt; callback, ObjectFactory&lt; T[], int &gt; bufferFactory)</a>	26
3.26	<a href="#">IAudioReader&lt; T &gt; Interface Template Reference</a>	26
3.26.1	<a href="#">Detailed Description</a>	26
3.27	<a href="#">IDataReader&lt; T &gt; Interface Template Reference</a>	26
3.27.1	<a href="#">Detailed Description</a>	26
3.27.2	<a href="#">Member Function Documentation</a>	27
3.27.2.1	<a href="#">Read(T[] buffer)</a>	27
3.28	<a href="#">IDecoder Interface Reference</a>	27
3.28.1	<a href="#">Detailed Description</a>	27

3.28.2	Member Function Documentation	27
3.28.2.1	Open(VoiceInfo info)	27
3.28.3	Property Documentation	27
3.28.3.1	Error	27
3.29	IDecoderDirect Interface Reference	28
3.29.1	Detailed Description	28
3.29.2	Member Function Documentation	28
3.29.2.1	DecodeToByte(byte[] buf)	28
3.29.2.2	DecodeToFloat(byte[] buf)	28
3.29.2.3	DecodeToShort(byte[] buf)	29
3.30	IDecoderQueued Interface Reference	29
3.30.1	Detailed Description	29
3.30.2	Member Function Documentation	29
3.30.2.1	Decode(byte[] buf)	29
3.31	IDecoderQueuedOutputImageNative Interface Reference	29
3.32	IEncoder Interface Reference	30
3.32.1	Detailed Description	30
3.32.2	Property Documentation	30
3.32.2.1	Error	30
3.33	IEncoderDataFlow< T > Interface Template Reference	30
3.33.1	Detailed Description	30
3.34	IEncoderDataFlowDirect< T > Interface Template Reference	30
3.34.1	Detailed Description	31
3.34.2	Member Function Documentation	31
3.34.2.1	EncodeAndGetOutput(T[] buf)	31
3.35	IEncoderNativeImageDirect Interface Reference	31
3.36	IEncoderQueued Interface Reference	31
3.36.1	Detailed Description	32
3.36.2	Member Function Documentation	32
3.36.2.1	GetOutput()	32
3.37	AudioUtil.ILevelMeter Interface Reference	32
3.37.1	Detailed Description	32
3.37.2	Member Function Documentation	32
3.37.2.1	ResetAccumAvgPeakAmp()	32
3.37.3	Property Documentation	33
3.37.3.1	AccumAvgPeakAmp	33
3.37.3.2	CurrentAvgAmp	33
3.37.3.3	CurrentPeakAmp	33
3.38	ILocalVoiceAudio Interface Reference	33
3.38.1	Detailed Description	33

3.38.2 Member Function Documentation	33
3.38.2.1 VoiceDetectorCalibrate(int durationMs)	33
3.38.3 Property Documentation	34
3.38.3.1 LevelMeter	34
3.38.3.2 VoiceDetector	34
3.38.3.3 VoiceDetectorCalibrating	34
3.39 ILoggable Interface Reference	34
3.40 ILogger Interface Reference	34
3.41 ImageBufferInfo Class Reference	34
3.42 ImageBufferNative Class Reference	35
3.43 ImageBufferNativeAlloc Class Reference	35
3.44 ImageBufferNativeGCHandleSinglePlane Class Reference	35
3.45 ImageBufferNativePool< T > Class Template Reference	36
3.46 IOSAudioForceToSpeaker Class Reference	36
3.47 IProcessor< T > Interface Template Reference	36
3.47.1 Detailed Description	36
3.47.2 Member Function Documentation	36
3.47.2.1 Process(T[] buf)	36
3.48 IServiceable Interface Reference	37
3.48.1 Detailed Description	37
3.48.2 Member Function Documentation	37
3.48.2.1 Service(LocalVoice localVoice)	37
3.49 ISyncAudioOut Interface Reference	37
3.50 AudioUtil.IVoiceDetector Interface Reference	38
3.50.1 Detailed Description	38
3.50.2 Property Documentation	38
3.50.2.1 ActivityDelayMs	38
3.50.2.2 Detected	38
3.50.2.3 DetectedTime	38
3.50.2.4 On	38
3.50.2.5 Threshold	38
3.50.3 Event Documentation	39
3.50.3.1 OnDetected	39
3.51 IVoiceFrontend Interface Reference	39
3.52 AudioUtil.LevelMeter< T > Class Template Reference	39
3.52.1 Detailed Description	40
3.52.2 Member Function Documentation	40
3.52.2.1 Process(T[] buf)	40
3.52.2.2 ResetAccumAvgPeakAmp()	40
3.53 AudioUtil.LevelMeterDummy Class Reference	40

3.53.1	Detailed Description	40
3.53.2	Member Function Documentation	41
3.53.2.1	ResetAccumAvgPeakAmp()	41
3.54	AudioUtil.LevelMeterFloat Class Reference	41
3.54.1	Detailed Description	41
3.54.2	Constructor & Destructor Documentation	41
3.54.2.1	LevelMeterFloat(int samplingRate, int numChannels)	41
3.55	AudioUtil.LevelMeterShort Class Reference	41
3.55.1	Detailed Description	42
3.55.2	Constructor & Destructor Documentation	42
3.55.2.1	LevelMeterShort(int samplingRate, int numChannels)	42
3.56	LoadBalancingFrontend Class Reference	42
3.56.1	Detailed Description	43
3.56.2	Constructor & Destructor Documentation	43
3.56.2.1	LoadBalancingFrontend(ConnectionProtocol connectionProtocol=ConnectionProtocol.Udp)	43
3.56.3	Member Function Documentation	43
3.56.3.1	ChangeAudioGroups(byte[] groupsToRemove, byte[] groupsToAdd)	43
3.56.3.2	Dispose()	44
3.56.3.3	SendDebugEchoVoicesInfo(int channelId)	44
3.56.3.4	Service()	44
3.56.4	Property Documentation	44
3.56.4.1	GlobalAudioGroup	44
3.56.4.2	VoiceClient	44
3.57	LocalVoice Class Reference	44
3.57.1	Detailed Description	45
3.57.2	Member Function Documentation	45
3.57.2.1	RemoveSelf()	45
3.57.3	Property Documentation	46
3.57.3.1	DebugEchoMode	46
3.57.3.2	Encrypt	46
3.57.3.3	FramesSent	46
3.57.3.4	FramesSentBytes	46
3.57.3.5	Group	46
3.57.3.6	Info	46
3.57.3.7	IsCurrentlyTransmitting	46
3.57.3.8	LocalUserObject	46
3.57.3.9	LocalUserServiceable	46
3.57.3.10	Reliable	46
3.57.3.11	TransmitEnabled	46



3.58 LocalVoiceAudio< T > Class Template Reference . . . . .	47
3.58.1 Detailed Description . . . . .	47
3.58.2 Member Function Documentation . . . . .	47
3.58.2.1 Create(VoiceClient voiceClient, byte voiceId, IEncoder encoder, VoiceInfo voiceInfo, int channelId) . . . . .	47
3.58.2.2 VoiceDetectorCalibrate(int durationMs) . . . . .	48
3.58.3 Property Documentation . . . . .	48
3.58.3.1 VoiceDetectorCalibrating . . . . .	48
3.59 LocalVoiceAudioDummy Class Reference . . . . .	48
3.59.1 Detailed Description . . . . .	49
3.59.2 Member Function Documentation . . . . .	49
3.59.2.1 VoiceDetectorCalibrate(int durationMs) . . . . .	49
3.59.3 Member Data Documentation . . . . .	49
3.59.3.1 Dummy . . . . .	49
3.60 LocalVoiceAudioFloat Class Reference . . . . .	49
3.60.1 Detailed Description . . . . .	49
3.61 LocalVoiceAudioShort Class Reference . . . . .	49
3.61.1 Detailed Description . . . . .	49
3.62 LocalVoiceFramed< T > Class Template Reference . . . . .	50
3.62.1 Detailed Description . . . . .	50
3.62.2 Member Function Documentation . . . . .	50
3.62.2.1 AddPostProcessor(params IProcessor< T >[] processors) . . . . .	50
3.62.2.2 AddPreProcessor(params IProcessor< T >[] processors) . . . . .	51
3.62.2.3 ClearProcessors() . . . . .	51
3.62.2.4 Dispose() . . . . .	51
3.62.2.5 PushData(T[] buf) . . . . .	51
3.62.2.6 PushDataAsync(T[] buf) . . . . .	51
3.62.3 Property Documentation . . . . .	51
3.62.3.1 PushDataAsyncReady . . . . .	51
3.63 LocalVoiceFramedBase Class Reference . . . . .	51
3.63.1 Detailed Description . . . . .	52
3.63.2 Property Documentation . . . . .	52
3.63.2.1 FrameSize . . . . .	52
3.64 Logger Class Reference . . . . .	52
3.65 MicWrapper Class Reference . . . . .	52
3.66 ObjectFactory< TType, TInfo > Interface Template Reference . . . . .	52
3.66.1 Detailed Description . . . . .	53
3.67 ObjectPool< TType, TInfo > Class Template Reference . . . . .	53
3.67.1 Detailed Description . . . . .	54
3.67.2 Constructor & Destructor Documentation . . . . .	54

3.67.2.1	ObjectPool(int capacity, string name)	54
3.67.2.2	ObjectPool(int capacity, string name, TInfo info)	54
3.67.3	Member Function Documentation	54
3.67.3.1	AcquireOrCreate()	54
3.67.3.2	AcquireOrCreate(TInfo info)	54
3.67.3.3	Dispose()	55
3.67.3.4	Init(TInfo info)	55
3.67.3.5	Release(TType obj, TInfo objInfo)	55
3.67.3.6	Release(TType obj)	55
3.67.4	Property Documentation	55
3.67.4.1	Info	55
3.68	OpusCodec Class Reference	55
3.69	OpusDecoder Class Reference	56
3.70	OpusEncoder Class Reference	56
3.70.1	Property Documentation	57
3.70.1.1	EncoderDelay	57
3.71	OpusException Class Reference	57
3.72	WebRTCAudioLib.Param Struct Reference	57
3.73	Recorder.PhotonVoiceCreatedParams Class Reference	57
3.74	PhotonVoiceLagSimulationGui Class Reference	58
3.74.1	Member Data Documentation	58
3.74.1.1	Visible	58
3.74.1.2	WindowId	58
3.74.1.3	WindowRect	58
3.74.2	Property Documentation	58
3.74.2.1	Peer	58
3.75	PhotonVoiceNetwork Class Reference	58
3.75.1	Detailed Description	59
3.75.2	Member Function Documentation	59
3.75.2.1	ConnectAndJoinRoom()	59
3.75.2.2	Disconnect()	59
3.75.3	Member Data Documentation	60
3.75.3.1	AutoConnectAndJoin	60
3.75.3.2	AutoCreateSpeakerIfNotFound	60
3.75.3.3	AutoLeaveAndDisconnect	60
3.75.3.4	VoiceRoomNameSuffix	60
3.75.4	Property Documentation	60
3.75.4.1	Instance	60
3.76	PhotonVoiceView Class Reference	60
3.76.1	Detailed Description	61

3.76.2	Member Data Documentation	61
3.76.2.1	AutoCreateRecorderIfNotFound	61
3.76.2.2	SetupDebugSpeaker	61
3.76.2.3	UsePrimaryRecorder	61
3.76.3	Property Documentation	61
3.76.3.1	IsRecorder	61
3.76.3.2	IsRecording	61
3.76.3.3	IsSetup	62
3.76.3.4	IsSpeaker	62
3.76.3.5	IsSpeaking	62
3.76.3.6	RecorderInUse	62
3.76.3.7	SpeakerInUse	62
3.77	PrimitiveArrayPool< T > Class Template Reference	62
3.77.1	Detailed Description	62
3.78	Recorder Class Reference	63
3.78.1	Detailed Description	64
3.78.2	Member Function Documentation	64
3.78.2.1	Init(VoiceClient voiceClient, object customObj=null)	64
3.78.2.2	ReInit()	65
3.78.2.3	VoiceDetectorCalibrate(int durationMs)	65
3.78.3	Property Documentation	65
3.78.3.1	AudioClip	65
3.78.3.2	AudioGroup	65
3.78.3.3	Bitrate	65
3.78.3.4	DebugEchoMode	65
3.78.3.5	Encrypt	65
3.78.3.6	FrameDuration	65
3.78.3.7	InputFactory	65
3.78.3.8	IsCurrentlyTransmitting	66
3.78.3.9	IsInitialized	66
3.78.3.10	LevelMeter	66
3.78.3.11	LoopAudioClip	66
3.78.3.12	MicrophoneType	66
3.78.3.13	PhotonMicrophoneDeviceId	66
3.78.3.14	PhotonMicrophoneEnumerator	66
3.78.3.15	ReliableMode	66
3.78.3.16	RequiresInit	66
3.78.3.17	SamplingRate	66
3.78.3.18	SourceType	66
3.78.3.19	TransmitEnabled	66

3.78.3.20 TypeConvert . . . . .	67
3.78.3.21 UnityMicrophoneDevice . . . . .	67
3.78.3.22 UserData . . . . .	67
3.78.3.23 VoiceDetection . . . . .	67
3.78.3.24 VoiceDetectionDelayMs . . . . .	67
3.78.3.25 VoiceDetectionThreshold . . . . .	67
3.78.3.26 VoiceDetector . . . . .	67
3.78.3.27 VoiceDetectorCalibrating . . . . .	67
3.79 RemoteVoiceInfo Class Reference . . . . .	67
3.79.1 Detailed Description . . . . .	68
3.79.2 Property Documentation . . . . .	68
3.79.2.1 ChannelId . . . . .	68
3.79.2.2 Info . . . . .	68
3.79.2.3 LocalUserObject . . . . .	68
3.79.2.4 PlayerId . . . . .	68
3.79.2.5 VoicId . . . . .	68
3.80 RemoteVoiceOptions Struct Reference . . . . .	68
3.80.1 Detailed Description . . . . .	68
3.80.2 Property Documentation . . . . .	69
3.80.2.1 Decoder . . . . .	69
3.80.2.2 LocalUserObject . . . . .	69
3.80.2.3 OnDecodedFrameByteAction . . . . .	69
3.80.2.4 OnDecodedFrameFloatAction . . . . .	69
3.80.2.5 OnDecodedFrameShortAction . . . . .	69
3.80.2.6 OnRemoteVoiceRemoveAction . . . . .	69
3.81 AudioUtil.Resampler< T > Class Template Reference . . . . .	69
3.81.1 Detailed Description . . . . .	69
3.81.2 Constructor & Destructor Documentation . . . . .	70
3.81.2.1 Resampler(int dstSize, int channels) . . . . .	70
3.81.3 Member Function Documentation . . . . .	70
3.81.3.1 Process(T[] buf) . . . . .	70
3.82 Speaker Class Reference . . . . .	70
3.82.1 Detailed Description . . . . .	71
3.82.2 Property Documentation . . . . .	71
3.82.2.1 Actor . . . . .	71
3.82.2.2 IsPlaying . . . . .	71
3.82.2.3 Lag . . . . .	71
3.82.2.4 OnRemoteVoiceRemoveAction . . . . .	71
3.83 SpeexLib Class Reference . . . . .	71
3.83.1 Member Data Documentation . . . . .	72

3.83.1.1	SPEEX_ECHO_GET_FRAME_SIZE	72
3.83.1.2	SPEEX_ECHO_GET_IMPULSE_RESPONSE	73
3.83.1.3	SPEEX_ECHO_GET_IMPULSE_RESPONSE_SIZE	73
3.83.1.4	SPEEX_ECHO_GET_SAMPLING_RATE	73
3.83.1.5	SPEEX_ECHO_SET_SAMPLING_RATE	73
3.83.1.6	SPEEX_PREPROCESS_GET_AGC	73
3.83.1.7	SPEEX_PREPROCESS_GET_AGC_DECREMENT	73
3.83.1.8	SPEEX_PREPROCESS_GET_AGC_GAIN	73
3.83.1.9	SPEEX_PREPROCESS_GET_AGC_INCREMENT	73
3.83.1.10	SPEEX_PREPROCESS_GET_AGC_LEVEL	73
3.83.1.11	SPEEX_PREPROCESS_GET_AGC_LOUDNESS	73
3.83.1.12	SPEEX_PREPROCESS_GET_AGC_MAX_GAIN	73
3.83.1.13	SPEEX_PREPROCESS_GET_AGC_TARGET	73
3.83.1.14	SPEEX_PREPROCESS_GET_DENOISE	74
3.83.1.15	SPEEX_PREPROCESS_GET_DEREVERB	74
3.83.1.16	SPEEX_PREPROCESS_GET_DEREVERB_DECAY	74
3.83.1.17	SPEEX_PREPROCESS_GET_DEREVERB_LEVEL	74
3.83.1.18	SPEEX_PREPROCESS_GET_ECHO_STATE	74
3.83.1.19	SPEEX_PREPROCESS_GET_ECHO_SUPPRESS	74
3.83.1.20	SPEEX_PREPROCESS_GET_ECHO_SUPPRESS_ACTIVE	74
3.83.1.21	SPEEX_PREPROCESS_GET_NOISE_PSD	74
3.83.1.22	SPEEX_PREPROCESS_GET_NOISE_PSD_SIZE	74
3.83.1.23	SPEEX_PREPROCESS_GET_NOISE_SUPPRESS	74
3.83.1.24	SPEEX_PREPROCESS_GET_PROB	74
3.83.1.25	SPEEX_PREPROCESS_GET_PROB_CONTINUE	74
3.83.1.26	SPEEX_PREPROCESS_GET_PROB_START	75
3.83.1.27	SPEEX_PREPROCESS_GET_PSD	75
3.83.1.28	SPEEX_PREPROCESS_GET_PSD_SIZE	75
3.83.1.29	SPEEX_PREPROCESS_GET_VAD	75
3.83.1.30	SPEEX_PREPROCESS_SET_AGC	75
3.83.1.31	SPEEX_PREPROCESS_SET_AGC_DECREMENT	75
3.83.1.32	SPEEX_PREPROCESS_SET_AGC_INCREMENT	75
3.83.1.33	SPEEX_PREPROCESS_SET_AGC_LEVEL	75
3.83.1.34	SPEEX_PREPROCESS_SET_AGC_MAX_GAIN	75
3.83.1.35	SPEEX_PREPROCESS_SET_AGC_TARGET	75
3.83.1.36	SPEEX_PREPROCESS_SET_DENOISE	75
3.83.1.37	SPEEX_PREPROCESS_SET_DEREVERB	75
3.83.1.38	SPEEX_PREPROCESS_SET_DEREVERB_DECAY	76
3.83.1.39	SPEEX_PREPROCESS_SET_DEREVERB_LEVEL	76
3.83.1.40	SPEEX_PREPROCESS_SET_ECHO_STATE	76

3.83.1.41 SPEEX_PREPROCESS_SET_ECHO_SUPPRESS	76
3.83.1.42 SPEEX_PREPROCESS_SET_ECHO_SUPPRESS_ACTIVE	76
3.83.1.43 SPEEX_PREPROCESS_SET_NOISE_SUPPRESS	76
3.83.1.44 SPEEX_PREPROCESS_SET_PROB_CONTINUE	76
3.83.1.45 SPEEX_PREPROCESS_SET_PROB_START	76
3.83.1.46 SPEEX_PREPROCESS_SET_VAD	76
3.84 SpeexProcessor Class Reference	76
3.85 TestTone Class Reference	77
3.86 AudioUtil.ToneAudioPusher< T > Class Template Reference	77
3.86.1 Detailed Description	78
3.86.2 Constructor & Destructor Documentation	78
3.86.2.1 ToneAudioPusher(int frequency=440, int bufSizeMs=100, int samplingRate=441000, int channels=2)	78
3.86.3 Member Function Documentation	78
3.86.3.1 SetCallback(Action< T[] > callback, ObjectFactory< T[], int > bufferFactory)	78
3.87 ToneAudioReader Class Reference	78
3.88 AudioUtil.ToneAudioReader< T > Class Template Reference	78
3.88.1 Detailed Description	79
3.88.2 Constructor & Destructor Documentation	79
3.88.2.1 ToneAudioReader(Func< double > clockSec=null, double frequency=440, int samplingRate=441000, int channels=2)	79
3.88.3 Member Function Documentation	79
3.88.3.1 Read(T[] buf)	79
3.88.4 Property Documentation	80
3.88.4.1 Channels	80
3.88.4.2 Error	80
3.88.4.3 SamplingRate	80
3.89 UnityAndroidAudioInAEC Class Reference	80
3.90 UnityAudioOut Class Reference	80
3.91 UnsupportedCodecException Class Reference	81
3.91.1 Detailed Description	81
3.91.2 Constructor & Destructor Documentation	81
3.91.2.1 UnsupportedCodecException(Codec codec, LocalVoice voice)	81
3.92 UnsupportedSampleTypeException Class Reference	81
3.92.1 Detailed Description	81
3.92.2 Constructor & Destructor Documentation	82
3.92.2.1 UnsupportedSampleTypeException(Type t)	82
3.93 OpusCodec.Util Class Reference	82
3.94 VoiceClient Class Reference	82
3.94.1 Detailed Description	83

3.94.2	Member Function Documentation	83
3.94.2.1	CreateLocalVoice(VoiceInfo voiceInfo, int channelId=ChannelAuto, IEncoder encoder=null)	83
3.94.2.2	CreateLocalVoiceAudio< T >(VoiceInfo voiceInfo, int channelId=ChannelAuto, IEncoder encoder=null)	84
3.94.2.3	CreateLocalVoiceAudioFromSource(Voice.VoiceInfo voiceInfo, Voice.IAudioDesc source, bool forceShort=false, int channelId=ChannelAuto, IEncoder encoder=null)	85
3.94.2.4	CreateLocalVoiceFramed< T >(VoiceInfo voiceInfo, int frameSize, int channelId=ChannelAuto, IEncoderDataFlow< T > encoder=null)	85
3.94.2.5	LocalVoicesInChannel(int channelId)	86
3.94.2.6	RemoteVoiceInfoDelegate(int channelId, int playerId, byte voiceId, VoiceInfo voiceInfo, ref RemoteVoiceOptions options)	86
3.94.2.7	RemoveLocalVoice(LocalVoice voice)	86
3.94.2.8	Service()	86
3.94.3	Property Documentation	86
3.94.3.1	DebugLostPercent	86
3.94.3.2	FramesLost	86
3.94.3.3	FramesReceived	86
3.94.3.4	FramesSent	86
3.94.3.5	FramesSentBytes	87
3.94.3.6	LocalVoices	87
3.94.3.7	OnRemoteVoiceInfoAction	87
3.94.3.8	RemoteVoiceInfos	87
3.94.3.9	RemoteVoiceLocalUserObjects	87
3.94.3.10	RoundTripTime	87
3.94.3.11	RoundTripTimeVariance	87
3.94.3.12	SuppressInfoDuplicateWarning	87
3.95	VoiceComponent Class Reference	87
3.96	VoiceConnection Class Reference	88
3.96.1	Detailed Description	89
3.96.2	Member Function Documentation	89
3.96.2.1	ConnectUsingSettings(AppSettings overwriteSettings=null)	89
3.96.3	Member Data Documentation	89
3.96.3.1	PrimaryRecorder	89
3.96.3.2	Settings	89
3.96.3.3	SpeakerFactory	89
3.96.4	Property Documentation	89
3.96.4.1	Client	89
3.96.4.2	ClientState	89
3.96.4.3	FramesLostPercent	90
3.96.4.4	FramesLostPerSecond	90

3.96.4.5	FramesReceivedPerSecond	90
3.96.4.6	Logger	90
3.96.4.7	LogLevel	90
3.96.4.8	SpeakerPrefab	90
3.96.4.9	VoiceClient	90
3.96.5	Event Documentation	90
3.96.5.1	SpeakerLinked	90
3.97	AudioUtil.VoiceDetector< T > Class Template Reference	90
3.97.1	Detailed Description	91
3.97.2	Member Function Documentation	91
3.97.2.1	Process(T[] buf)	91
3.97.3	Property Documentation	91
3.97.3.1	ActivityDelayMs	91
3.97.3.2	Detected	92
3.97.3.3	DetectedTime	92
3.97.3.4	On	92
3.97.3.5	Threshold	92
3.97.4	Event Documentation	92
3.97.4.1	OnDetected	92
3.98	AudioUtil.VoiceDetectorCalibration< T > Class Template Reference	92
3.98.1	Detailed Description	93
3.98.2	Constructor & Destructor Documentation	93
3.98.2.1	VoiceDetectorCalibration(IVoiceDetector voiceDetector, ILevelMeter levelMeter, int samplingRate, int channels)	93
3.98.3	Member Function Documentation	93
3.98.3.1	Process(T[] buf)	93
3.98.3.2	VoiceDetectorCalibrate(int durationMs)	93
3.99	AudioUtil.VoiceDetectorDummy Class Reference	93
3.99.1	Detailed Description	94
3.100	AudioUtil.VoiceDetectorFloat Class Reference	94
3.100.1	Detailed Description	94
3.100.2	Constructor & Destructor Documentation	94
3.100.2.1	VoiceDetectorFloat(int samplingRate, int numChannels)	94
3.101	AudioUtil.VoiceDetectorShort Class Reference	94
3.101.1	Detailed Description	95
3.101.2	Constructor & Destructor Documentation	95
3.101.2.1	VoiceDetectorShort(int samplingRate, int numChannels)	95
3.102	VoiceEventCode Class Reference	95
3.102.1	Detailed Description	95
3.102.2	Member Function Documentation	96



3.102.2.1 GetCode(int channelId) . . . . .	96
3.102.2.2 TryGetChannelID(byte evCode, int maxChannels, out byte channelId) . . . . .	97
3.102.3 Member Data Documentation . . . . .	97
3.102.3.1 Code0 . . . . .	97
3.103 VoiceInfo Struct Reference . . . . .	97
3.103.1 Detailed Description . . . . .	98
3.103.2 Member Function Documentation . . . . .	98
3.103.2.1 CreateAudioOpus(POpusCodec.Enums.SamplingRate samplingRate, int sourceSamplingRate, int channels, OpusCodec.FrameDuration frameDuration←Us, int bitrate, object userdata=null) . . . . .	98
3.103.3 Property Documentation . . . . .	98
3.103.3.1 Bitrate . . . . .	98
3.103.3.2 Channels . . . . .	98
3.103.3.3 FrameDurationSamples . . . . .	99
3.103.3.4 FrameDurationUs . . . . .	99
3.103.3.5 FrameSize . . . . .	99
3.103.3.6 Height . . . . .	99
3.103.3.7 SamplingRate . . . . .	99
3.103.3.8 SourceSamplingRate . . . . .	99
3.103.3.9 UserData . . . . .	99
3.103.3.10 Width . . . . .	99
3.104 AudioUtil.VoiceLevelDetectCalibrate< T > Class Template Reference . . . . .	99
3.104.1 Detailed Description . . . . .	100
3.104.2 Constructor & Destructor Documentation . . . . .	100
3.104.2.1 VoiceLevelDetectCalibrate(int samplingRate, int channels) . . . . .	100
3.104.3 Member Function Documentation . . . . .	100
3.104.3.1 Calibrate(int durationMs) . . . . .	100
3.104.3.2 Process(T[] buf) . . . . .	100
3.104.4 Property Documentation . . . . .	100
3.104.4.1 Detector . . . . .	100
3.104.4.2 Level . . . . .	101
3.105 VoiceLogger Class Reference . . . . .	101
3.106 WebRtcAudioDsp Class Reference . . . . .	101
3.107 WebRTCAudioLib Class Reference . . . . .	101
3.108 WebRTCAudioProcessor Class Reference . . . . .	102



# Chapter 1

## Photon Voice Doxygen Readme

### Offline Docs

#### Manual Generation

To manually generate doxygen offline files:

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

#### Automatic Generation

Simply run "Docs\generate\_offline.bat". Open the file and edit DOXYGEN\_PATH accordingly. Also you need a LaTeX distribution installed and some packages/dependencies.

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

### Files

#### HTML

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

#### CHM

"PhotonVoice-Documentation.chm" should be copied

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

from "Docs\TempOutputDocs\VOICE\_OFFLINE\_HTML\PhotonVoice-Documentation.chm".

#### PDF

"PhotonVoice-Documentation.pdf" should be copied

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

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

### Online Docs

To manually generate doxygen online files:

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



## Chapter 2

# Namespace Documentation

## 2.1 Photon Namespace Reference

### Namespaces

- namespace [Voice](#)

## 2.2 Photon.Voice Namespace Reference

### Namespaces

- namespace [PUN](#)
- namespace [Unity](#)

### Classes

- class [AudioDesc](#)
- class [AudioInEnumerator](#)
- class [AudioStreamPlayer](#)
- class [AudioUtil](#)  
*Collection of Audio Utility functions and classes.*
- class [BufferReaderPushAdapter](#)  
*Simple [BufferReaderPushAdapterBase](#) implementation using a single buffer, using synchronous [LocalVoice.PushData](#).*
- class [BufferReaderPushAdapterAsyncPool](#)  
*[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#).*
- class [BufferReaderPushAdapterAsyncPoolCopy](#)  
*[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#) and data copy.*
- class [BufferReaderPushAdapterAsyncPoolFloatToShort](#)  
*[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#), converting float samples to short.*
- class [BufferReaderPushAdapterBase](#)  
*Adapter base class to move data by reading from [IDataReader.Read](#) and pushing to [LocalVoice](#).*
- class [FactoryPrimitiveArrayPool](#)  
*[PrimitiveArrayPool<T>](#) as wrapped in object factory interface.*
- class [FactoryReusableArray](#)

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

- class [Framer](#)  
*Utility class to re-frame audio packets.*
- interface [IAudioDesc](#)  
*Audio Source interface.*
- interface [IAudioOut](#)
- interface [IAudioPusher](#)  
*Audio Pusher interface.*
- interface [IAudioReader](#)  
*Audio Reader interface.*
- interface [IDataReader](#)  
*Interface for pulling data, in case this is more appropriate than pushing it.*
- interface [IDecoder](#)  
*Generic media decoder interface.*
- interface [IDecoderDirect](#)  
*Interface for a media decoder that synchronously decodes data.*
- interface [IDecoderQueued](#)  
*Interface for a media decoder that feeds its data output into a separate method or callback asynchronously, or does not produce output at all.*
- interface [IDecoderQueuedOutputImageNative](#)
- interface [IEncoder](#)  
*Generic media encoder interface.*
- interface [IEncoderDataFlow](#)  
*Interface for a generic media encoder data flow.*
- interface [IEncoderDataFlowDirect](#)  
*Interface for an encoder data flow that synchronously encodes data.*
- interface [IEncoderNativeImageDirect](#)
- interface [IEncoderQueued](#)  
*Interface for an encoder data flow that returns compressed data independently (produces output on its own) or asynchronously (usually from a queue).*
- interface [ILocalVoiceAudio](#)  
*Interface for an outgoing audio stream.*
- interface [ILogger](#)
- class [ImageBufferInfo](#)
- class [ImageBufferNative](#)
- class [ImageBufferNativeAlloc](#)
- class [ImageBufferNativeGCHandleSinglePlane](#)
- class [ImageBufferNativePool](#)
- interface [IProcessor](#)  
*Audio Processor interface.*
- interface [IServiceable](#)  
*Interface for classes that want their [Service\(\)](#) function to be called regularly in the context of a [LocalVoice](#).*
- interface [ISyncAudioOut](#)
- interface [IVoiceFrontend](#)
- class [LoadBalancingFrontend](#)  
*Extends [LoadBalancingClient](#) with audio streaming functionality.*
- class [LocalVoice](#)  
*Represents outgoing data stream.*
- class [LocalVoiceAudio](#)  
*Outgoing audio stream.*
- class [LocalVoiceAudioDummy](#)

- Dummy [LocalVoiceAudio](#)*
- class [LocalVoiceAudioFloat](#)
  - Specialization of [LocalVoiceAudio](#) for float audio*
- class [LocalVoiceAudioShort](#)
  - Specialization of [LocalVoiceAudio](#) for short audio*
- class [LocalVoiceFramed](#)
  - Typed re-framing [LocalVoice](#)*
- class [LocalVoiceFramedBase](#)
  - Typed re-framing [LocalVoice](#)*
- interface [ObjectFactory](#)
  - Uniform interface to [ObjectPool](#)< TType, TInfo> and single reusable object.*
- class [ObjectPool](#)
  - Generic Pool to re-use objects of a certain type (TType) that optionally match a certain property or set of properties (TInfo).*
- class [OpusCodec](#)
- class [PrimitiveArrayPool](#)
  - Pool of Arrays with components of type T, with [ObjectPool](#) info being the array's size.*
- class **RemoteVoice**
- class [RemoteVoiceInfo](#)
  - Information about a remote voice (incoming stream).*
- struct [RemoteVoiceOptions](#)
  - Event Actions and other options for a remote voice (incoming stream).*
- class [SpeexLib](#)
- class [SpeexProcessor](#)
- class [UnsupportedCodecException](#)
  - Exception thrown if an unsupported codec is encountered.*
- class [UnsupportedSampleTypeException](#)
  - Exception thrown if an unsupported audio sample type is encountered.*
- class [VoiceClient](#)
  - Base class for [Voice](#) clients implamantations*
- class **VoiceCodec**
- class [VoiceEventCode](#)
  - PhotonVoice communication uses a single type of event, but differentiates transmission Channels by encoding a channelId into [VoiceEventCode](#).*
- struct [VoiceInfo](#)
  - Describes stream properties.*
- class [WebRTCAudioLib](#)
- class [WebRTCAudioProcessor](#)

## Enumerations

- enum **EventSubcode** : byte
- enum **EventParam** : byte
- enum [Codec](#)
  - Enum for Media Codecs supported by PhotonVoice.*
- enum **ImageFormat**
- enum **Rotation**
- enum **Flip**

## Functions

- delegate void **OnImageOutputNative** (IntPtr buf, int width, int height, int stride)

## 2.2.1 Enumeration Type Documentation

### 2.2.1.1 enum Codec [strong]

Enum for Media Codecs supported by PhotonVoice.

Transmitted in [VoiceInfo](#). Do not change the values of this Enum!

Enumerator

**AudioOpus** OPUS audio

## 2.3 Photon.Voice.PUN Namespace Reference

### Classes

- class [PhotonVoiceNetwork](#)

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

- class [PhotonVoiceView](#)

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

## 2.4 Photon.Voice.Unity Namespace Reference

### Namespaces

- namespace [UtilityScripts](#)

### Classes

- class [AudioClipWrapper](#)
- class [AudioOutCapture](#)
- interface [ILoggable](#)
- class [IOSAudioForceToSpeaker](#)
- class [Logger](#)
- class [MicWrapper](#)
- class [Recorder](#)

*Component representing outgoing audio stream in scene.*

- class [Speaker](#)

*Component representing remote audio stream in local scene.*

- class [UnityAndroidAudioInAEC](#)
- class [UnityAudioOut](#)
- class [VoiceComponent](#)
- class [VoiceConnection](#)

*Component that represents a client voice connection to [Photon](#) Servers.*

- class [VoiceLogger](#)
- class [WebRtcAudioDsp](#)



## 2.5 Photon.Voice.Unity.UtilityScripts Namespace Reference

### Classes

- class [ConnectAndJoin](#)
- class [PhotonVoiceLagSimulationGui](#)
- class [TestTone](#)
- class [ToneAudioReader](#)

## 2.6 POpusCodec Namespace Reference

### Namespaces

- namespace [Enums](#)

### Classes

- class [OpusDecoder](#)
- class [OpusEncoder](#)
- class [OpusException](#)
- class **Wrapper**

## 2.7 POpusCodec.Enums Namespace Reference

### Enumerations

- enum [Bandwidth](#) : int
- enum [Channels](#) : int
- enum **Complexity** : int
- enum [Delay](#)

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

- enum **ForceChannels** : int
- enum [OpusApplicationType](#) : int
- enum **OpusStatusCode** : int
- enum **SamplingRate** : int
- enum [SignalHint](#) : int

### 2.7.1 Enumeration Type Documentation

#### 2.7.1.1 enum **Bandwidth** : int [strong]

##### Enumerator

**Narrowband** Up to 4Khz

**Mediumband** Up to 6Khz

**Wideband** Up to 8Khz

**SuperWideband** Up to 12Khz

**Fullband** Up to 20Khz (High Definition)

### 2.7.1.2 enum Channels : int [strong]

#### Enumerator

- Mono** 1 Channel
- Stereo** 2 Channels

### 2.7.1.3 enum Delay [strong]

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

#### Enumerator

- Delay2dot5ms** 2.5ms
- Delay5ms** 5ms
- Delay10ms** 10ms
- Delay20ms** 20ms
- Delay40ms** 40ms
- Delay60ms** 60ms

### 2.7.1.4 enum OpusApplicationType : int [strong]

#### Enumerator

- Voip** Gives best quality at a given bitrate for voice signals. It enhances the input signal by high-pass filtering and emphasizing formants and harmonics. Optionally it includes in-band forward error correction to protect against packet loss. Use this mode for typical VoIP applications. Because of the enhancement, even at high bitrates the output may sound different from the input.
- Audio** Gives best quality at a given bitrate for most non-voice signals like music. Use this mode for music and mixed (music/voice) content, broadcast, and applications requiring less than 15 ms of coding delay.
- RestrictedLowDelay** Configures low-delay mode that disables the speech-optimized mode in exchange for slightly reduced delay.

### 2.7.1.5 enum SignalHint : int [strong]

#### Enumerator

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

## Chapter 3

# Class Documentation

### 3.1 SpeexProcessor.AECLatencyResultType Struct Reference

#### Public Attributes

- int **LatencyMs**
- int **LatencyDelayedMs**
- bool **PlayDetected**
- bool **PlayDelayedDetected**
- bool **RecDetected**

### 3.2 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]

### 3.3 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]

## 3.4 AudioInEnumerator Class Reference

Inherits IDisposable.

### Public Member Functions

- **AudioInEnumerator** ([Voice.ILogger](#) logger)
- void **Refresh** ()
- string **NameAtIndex** (int i)
- int **IDAtIndex** (int i)
- bool **IDIsValid** (int id)
- void **Dispose** ()

### Public Attributes

- readonly bool **IsSupported** = false

## Properties

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

## 3.5 AudioOutCapture Class Reference

Inherits MonoBehaviour.

### Events

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

## 3.6 AudioStreamPlayer Class Reference

Inherits [IAudioOut](#).

### Public Member Functions

- **AudioStreamPlayer** ([Photon.Voice.ILogger](#) logger, [Photon.Voice.ISyncAudioOut](#) audioOut, string logPrefix, bool debugInfo)
- void **Start** (int frequency, int channels, int frameSamples, int playDelayMs)
- void **Service** ()
- void **Push** (float[] frame)
- void **Stop** ()

## Properties

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

## 3.7 AudioUtil Class Reference

Collection of Audio Utility functions and classes.

## Classes

- interface [ILevelMeter](#)  
*Audio Level Metering interface.*
- interface [IVoiceDetector](#)  
*Voice Activity Detector interface.*
- class [LevelMeter](#)  
*Audio Level Meter.*
- class [LevelMeterDummy](#)  
*Dummy Audio Level Meter that doesn't actually do anything.*
- class [LevelMeterFloat](#)  
*LevelMeter specialization for float audio.*
- class [LevelMeterShort](#)  
*LevelMeter specialization for short audio.*
- class [Resampler](#)  
*Sample-rate conversion Audio Processor.*
- class [ToneAudioPusher](#)  
*IAudioPusher that provides a constant tone signal.*
- class [ToneAudioReader](#)  
*IAudioReader that provides a constant tone signal.*
- class [VoiceDetector](#)  
*Simple voice activity detector triggered by signal level.*
- class [VoiceDetectorCalibration](#)  
*Calibration Utility for Voice Detector*
- class [VoiceDetectorDummy](#)  
*Dummy VoiceDetector that doesn't actually do anything.*
- class [VoiceDetectorFloat](#)  
*VoiceDetector specialization for float audio.*
- class [VoiceDetectorShort](#)  
*VoiceDetector specialization for float audio.*
- class [VoiceLevelDetectCalibrate](#)  
*Utility Audio Processor Voice Detection Calibration.*

## Static Public Member Functions

- static void [Resample< T >](#) (T[] src, T[] dst, int dstCount, int channels)  
*Resample audio data so that the complete src buffer fits into dstCount samples in the dst buffer.*
- static void [ResampleAndConvert](#) (short[] src, float[] dst, int dstCount, int channels)  
*Resample audio data so that the complete src buffer fits into dstCount samples in the dst buffer, and convert short to float samples along the way.*
- static void [ResampleAndConvert](#) (float[] src, short[] dst, int dstCount, int channels)

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

- static void [Convert](#) (float[] src, short[] dst, int dstCount)  
*Convert audio buffer from float to short samples.*
- static void [Convert](#) (short[] src, float[] dst, int dstCount)  
*Convert audio buffer from short to float samples.*
- static void [ForceToStereo](#)< T > (T[] src, T[] dst, int srcChannels)  
*Convert audio buffer with arbitrary number of channels to stereo.*

### 3.7.1 Detailed Description

Collection of Audio Utility functions and classes.

### 3.7.2 Member Function Documentation

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

Convert audio buffer from float to short samples.

Parameters

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

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

Convert audio buffer from short to float samples.

Parameters

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

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

Convert audio buffer with arbitrary number of channels to stereo.

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

Parameters

<i>src</i>	Source buffer.
<i>dst</i>	Destination buffer.
<i>srcChannels</i>	Number of (interleaved) channels in src.

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

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

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

**Parameters**

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

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

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

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

**Parameters**

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

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

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

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

**Parameters**

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

**3.8 BufferReaderPushAdapter< T > Class Template Reference**

Simple [BufferReaderPushAdapterBase](#) implementation using a single buffer, using synchronous LocalVoice.Push↔Data

Inherits [BufferReaderPushAdapterBase< T >](#).

**Public Member Functions**

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

**Protected Attributes**

- T[] **buffer**

### 3.8.1 Detailed Description

Simple [BufferReaderPushAdapterBase](#) implementation using a single buffer, using synchronous [LocalVoice.Push↔Data](#)

### 3.8.2 Constructor & Destructor Documentation

#### 3.8.2.1 [BufferReaderPushAdapter](#) ( [LocalVoice](#) *localVoice*, [IDataReader](#)< T > *reader* )

Create a new [BufferReaderPushAdapter](#) instance

Parameters

<i>localVoice</i>	<a href="#">LocalVoice</a> instance to push data to.
<i>reader</i>	<a href="#">DataReader</a> to read from.

### 3.8.3 Member Function Documentation

#### 3.8.3.1 override void [Service](#) ( [LocalVoice](#) *localVoice* ) [virtual]

Do the actual data read/push.

Parameters

<i>localVoice</i>	<a href="#">LocalVoice</a> instance to push data to.
-------------------	--

Implements [BufferReaderPushAdapterBase](#)< T >.

## 3.9 [BufferReaderPushAdapterAsyncPool](#)< T > Class Template Reference

[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#).

Inherits [BufferReaderPushAdapterBase](#)< T >.

### Public Member Functions

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

### Additional Inherited Members

#### 3.9.1 Detailed Description

[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#).

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

Expects *localVoice* to be a [LocalVoiceFramed](#)<T> of same T.

#### 3.9.2 Constructor & Destructor Documentation



#### 3.9.2.1 `BufferedReaderPushAdapterAsyncPool ( LocalVoice localVoice, IDataReader< T > reader )`

Create a new [BufferedReaderPushAdapter](#) instance

## Parameters

<i>localVoice</i>	<a href="#">LocalVoice</a> instance to push data to.
<i>reader</i>	<a href="#">DataReader</a> to read from.

### 3.9.3 Member Function Documentation

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

Do the actual data read/push.

## Parameters

<i>localVoice</i>	<a href="#">LocalVoice</a> instance to push data to. Must be a <a href="#">LocalVoiceFramed&lt;T&gt;</a> of same T.
-------------------	---

Implements [BufferReaderPushAdapterBase< T >](#).

## 3.10 BufferReaderPushAdapterAsyncPoolCopy< T > Class Template Reference

[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#) and data copy.

Inherits [BufferReaderPushAdapterBase< T >](#).

### Public Member Functions

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

### Protected Attributes

- [T\[\]](#) **buffer**

#### 3.10.1 Detailed Description

[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#) and data copy.

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

#### 3.10.2 Constructor & Destructor Documentation

##### 3.10.2.1 BufferReaderPushAdapterAsyncPoolCopy ( [LocalVoice](#) *localVoice*, [IDataReader< T >](#) *reader* )

Create a new [BufferReaderPushAdapter](#) instance

## Parameters

<i>localVoice</i>	<a href="#">LocalVoice</a> instance to push data to.
-------------------	--

<i>reader</i>	DataReader to read from.
---------------	--------------------------

### 3.10.3 Member Function Documentation

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

Do the actual data read/push.

Parameters

<i>localVoice</i>	<a href="#">LocalVoice</a> instance to push data to. Must be a <a href="#">LocalVoiceFramed&lt;T&gt;</a> of same T.
-------------------	---

Implements [BufferReaderPushAdapterBase< T >](#).

## 3.11 BufferReaderPushAdapterAsyncPoolFloatToShort Class Reference

[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#), converting float samples to short.

Inherits [BufferReaderPushAdapterBase< float >](#).

### Public Member Functions

- [BufferReaderPushAdapterAsyncPoolFloatToShort](#) ([Voice.LocalVoice](#) localVoice, [Voice.IDataReader](#)< float > reader)  
*Create a new [BufferReaderPushAdapter](#) instance*
- override void [Service](#) ([Voice.LocalVoice](#) localVoice)  
*Do the actual data read/push.*

### Additional Inherited Members

#### 3.11.1 Detailed Description

[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#), converting float samples to short.

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

Expects localVoice to be a [LocalVoiceFramed<T>](#) of same T.

#### 3.11.2 Constructor & Destructor Documentation

##### 3.11.2.1 BufferReaderPushAdapterAsyncPoolFloatToShort ( Voice.LocalVoice localVoice, Voice.IDataReader< float > reader )

Create a new [BufferReaderPushAdapter](#) instance

Parameters

<i>localVoice</i>	<a href="#">LocalVoice</a> instance to push data to.
-------------------	--

<i>reader</i>	DataReader to read from.
---------------	--------------------------

### 3.11.3 Member Function Documentation

#### 3.11.3.1 override void Service ( Voice.LocalVoice localVoice )

Do the actual data read/push.

Parameters

<i>localVoice</i>	<a href="#">LocalVoice</a> instance to push data to. Must be a LocalVoiceFramed<T> of same T.
-------------------	---

## 3.12 BufferReaderPushAdapterBase< T > Class Template Reference

Adapter base class to move data by reading from [IDataReader.Read](#) and pushing to [LocalVoice](#).

Inherits [IServiceable](#).

Inherited by [BufferReaderPushAdapter< T >](#), [BufferReaderPushAdapterAsyncPool< T >](#), and [BufferReaderPushAdapterAsyncPoolCopy< T >](#).

### Public Member Functions

- abstract void [Service](#) ([LocalVoice](#) localVoice)  
*Do the actual data read/push.*
- [BufferReaderPushAdapterBase](#) ([IDataReader](#)< T > reader)  
*Create a new [BufferReaderPushAdapterBase](#) instance*
- void [Dispose](#) ()  
*Release resources associated with this instance.*

### Protected Attributes

- [IDataReader](#)< T > **reader**

#### 3.12.1 Detailed Description

Adapter base class to move data by reading from [IDataReader.Read](#) and pushing to [LocalVoice](#).

Use this with a [LocalVoice](#) of same T type.

#### 3.12.2 Constructor & Destructor Documentation

##### 3.12.2.1 BufferReaderPushAdapterBase ( IDataReader< T > reader )

Create a new [BufferReaderPushAdapterBase](#) instance

Parameters

<i>reader</i>	DataReader to read from.
---------------	--------------------------

### 3.12.3 Member Function Documentation

#### 3.12.3.1 void Dispose ( )

Release resources associated with this instance.

#### 3.12.3.2 abstract void Service ( LocalVoice localVoice ) [pure virtual]

Do the actual data read/push.

Parameters

<i>localVoice</i>	<a href="#">LocalVoice</a> instance to push data to.
-------------------	--

Implements [IServiceable](#).

Implemented in [BufferReaderPushAdapterAsyncPoolCopy< T >](#), [BufferReaderPushAdapterAsyncPool< T >](#), and [BufferReaderPushAdapter< T >](#).

## 3.13 WebRTCAudioLib.ConfigParam Struct Reference

### Public Attributes

- const int **AEC\_DELAY\_AGNOSTIC** = 12
- const int **AEC\_EXTENDED\_FILTER** = 13
- const int **AGC\_EXPERIMENTAL** = 53
- const int **AGC\_EXPERIMENTAL\_STARTUP\_MIN\_VOLUME** = 54
- const int **AGC\_EXPERIMENTAL\_CLIP\_LEVEL\_MIN** = 55

## 3.14 ConnectAndJoin Class Reference

Inherits MonoBehaviour, IConnectionCallbacks, and IMatchmakingCallbacks.

### Public Member Functions

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

### Public Attributes

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

## Properties

- bool **IsConnected** [get]

## 3.15 OpusCodec.Decoder Class Reference

Inherits [IDecoderDirect](#).

### Public Member Functions

- **Decoder** ([ILogger](#) logger)
- void **Open** ([VoiceInfo](#) i)  
*Open (initialize) the decoder.*
- byte[] **DecodeToByte** (byte[] buf)  
*Decode the given raw data buffer.*
- float[] **DecodeToFloat** (byte[] buf)  
*Decode the given raw data buffer to floating point audio.*
- short[] **DecodeToShort** (byte[] buf)  
*Decode the given raw data buffer to 'short' (16-bit) audio.*
- void **Dispose** ()

## Properties

- string **Error** [get]

### 3.15.1 Member Function Documentation

#### 3.15.1.1 byte[] DecodeToByte ( byte[] buf )

Decode the given raw data buffer.

##### Parameters

<i>buf</i>	Buffer of encoded (compressed) data.
------------	--------------------------------------

##### Returns

Buffer of decoded (uncompressed) data.

Implements [IDecoderDirect](#).

#### 3.15.1.2 float[] DecodeToFloat ( byte[] buf )

Decode the given raw data buffer to floating point audio.

Only sensible for audio data.

##### Parameters

<i>buf</i>	Buffer of encoded (compressed) data.
------------	--------------------------------------

##### Returns

Buffer of decoded (uncompressed) data.

Implements [IDecoderDirect](#).

## 3.15.1.3 short [] DecodeToShort ( byte[] buf )

Decode the given raw data buffer to 'short' (16-bit) audio.

Only sensible for audio data.

## Parameters

<i>buf</i>	Buffer of encoded (compressed) data.
------------	--------------------------------------

## Returns

Buffer of decoded (uncompressed) data.

Implements [IDecoderDirect](#).

## 3.15.1.4 void Open ( VoicelInfo info )

Open (initialize) the decoder.

## Parameters

<i>info</i>	Properties of the data stream to decode.
-------------	--

Implements [IDecoder](#).

## 3.16 OpusCodec.Encoder&lt; T &gt; Class Template Reference

Inherits [IEncoderDataFlowDirect< T >](#).

## Public Member Functions

- void **Dispose** ()
- abstract ArraySegment< byte > [EncodeAndGetOutput](#) (T[] buf)  
*Encode the given uncompressed media data.*

## Protected Member Functions

- **Encoder** ([VoicelInfo](#) i, [ILogger](#) logger)

## Protected Attributes

- [OpusEncoder](#) **encoder**
- bool **disposed**

## Properties

- string **Error** [get]

## 3.16.1 Member Function Documentation

## 3.16.1.1 abstract ArraySegment&lt;byte&gt; EncodeAndGetOutput ( T[] buf ) [pure virtual]

Encode the given uncompressed media data.

## Parameters

<i>buf</i>	Array containing raw (uncompressed) data (e.g. audio samples).
------------	--

## Returns

Encoded (compressed) data.

Implements [IEncoderDataFlowDirect< T >](#).

### 3.17 OpusCodec.EncoderFactory Class Reference

#### Static Public Member Functions

- static [IEncoder](#) **Create** ([VoiceInfo](#) i, [LocalVoice](#) v)

### 3.18 OpusCodec.EncoderFloat Class Reference

Inherits [OpusCodec.Encoder< float >](#).

#### Public Member Functions

- override [ArraySegment< byte >](#) **EncodeAndGetOutput** ([float\[\]](#) buf)

#### Additional Inherited Members

### 3.19 OpusCodec.EncoderShort Class Reference

Inherits [OpusCodec.Encoder< short >](#).

#### Public Member Functions

- override [ArraySegment< byte >](#) **EncodeAndGetOutput** ([short\[\]](#) buf)

#### Additional Inherited Members

### 3.20 FactoryPrimitiveArrayPool< T > Class Template Reference

[PrimitiveArrayPool<T>](#) as wrapped in object factory interface.

Inherits [ObjectFactory< T\[\], int >](#).

#### Public Member Functions

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



## Properties

- int **Info** [get]

### 3.20.1 Detailed Description

PrimitiveArrayPool<T> as wrapped in object factory interface.

#### Template Parameters

<i>T</i>	Array element type.
----------	---------------------

## 3.21 FactoryReusableArray< T > Class Template Reference

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

Inherits [ObjectFactory< T\[\], int >](#).

## Public Member Functions

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

## Properties

- int **Info** [get]

### 3.21.1 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

<i>T</i>	Array element type.
----------	---------------------

## 3.22 Framer< T > Class Template Reference

Utility class to re-frame audio packets.

## Public Member Functions

- [Framer](#) (int frameSize)  
*Create new [Framer](#) instance.*
- int [Count](#) (int bufLen)  
*Get the number of frames available after adding bufLen samples.*

- `IEnumerable< T[] > Frame (T[] buf)`  
*Append arbitrary-sized buffer and return available full frames.*

### 3.22.1 Detailed Description

Utility class to re-frame audio packets.

### 3.22.2 Constructor & Destructor Documentation

#### 3.22.2.1 `Framer ( int frameSize )`

Create new `Framer` instance.

### 3.22.3 Member Function Documentation

#### 3.22.3.1 `int Count ( int bufLen )`

Get the number of frames available after adding `bufLen` samples.

Parameters

<code>bufLen</code>	Number of samples that would be added.
---------------------	--

Returns

Number of full frames available when adding `bufLen` samples.

#### 3.22.3.2 `IEnumerable<T[]> Frame ( T[] buf )`

Append arbitrary-sized buffer and return available full frames.

Parameters

<code>buf</code>	Array of samples to add.
------------------	--------------------------

Returns

Enumerator of full frames (might be none).

## 3.23 IAudioDesc Interface Reference

Audio Source interface.

Inherits `IDisposable`.

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

### Properties

- `int SamplingRate [get]`  
*Sampling rate of the audio signal (in Hz).*
- `int Channels [get]`  
*Number of channels in the audio signal.*
- `string Error [get]`  
*If not null, audio object is in invalid state.*

### 3.23.1 Detailed Description

Audio Source interface.

### 3.23.2 Property Documentation

#### 3.23.2.1 int Channels [get]

Number of channels in the audio signal.

#### 3.23.2.2 string Error [get]

If not null, audio object is in invalid state.

#### 3.23.2.3 int SamplingRate [get]

Sampling rate of the audio signal (in Hz).

## 3.24 IAudioOut Interface Reference

Inherited by [AudioStreamPlayer](#), and [ISyncAudioOut](#).

### Public Member Functions

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

### Properties

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

## 3.25 IAudioPusher< T > Interface Template Reference

Audio Pusher interface.

Inherits [IAudioDesc](#).

Inherited by [AudioUtil.ToneAudioPusher< T >](#).

### Public Member Functions

- void **SetCallback** (Action< T[]> callback, [ObjectFactory< T\[\], int >](#) bufferFactory)  
*Set the callback function used for pushing data.*

## Additional Inherited Members

### 3.25.1 Detailed Description

Audio Pusher interface.

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

### 3.25.2 Member Function Documentation

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

Set the callback function used for pushing data.

Parameters

<i>callback</i>	Callback function to use.
<i>localVoice</i>	Outgoing audio stream, for context.

Implemented in [AudioUtil.ToneAudioPusher< T >](#).

## 3.26 IAudioReader< T > Interface Template Reference

Audio Reader interface.

Inherits [IDataReader< T >](#), and [IAudioDesc](#).

Inherited by [AudioUtil.ToneAudioReader< T >](#).

## Additional Inherited Members

### 3.26.1 Detailed Description

Audio Reader interface.

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

## 3.27 IDataReader< T > Interface Template Reference

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

Inherits [IDisposable](#).

Inherited by [IAudioReader< T >](#).

## Public Member Functions

- `bool Read (T[] buffer)`

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

### 3.27.1 Detailed Description

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

### 3.27.2 Member Function Documentation

#### 3.27.2.1 bool Read ( T[] *buffer* )

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

##### Parameters

<i>buffer</i>	Buffer to fill.
---------------	-----------------

##### Returns

True if buffer was filled successfully, false otherwise.

Implemented in [AudioUtil.ToneAudioReader< T >](#).

## 3.28 IDecoder Interface Reference

Generic media decoder interface.

Inherits IDisposable.

Inherited by [IDecoderDirect](#), and [IDecoderQueued](#).

### Public Member Functions

- void [Open](#) ([VoiceInfo](#) info)  
*Open (initialize) the decoder.*

### Properties

- string [Error](#) [get]  
*If not null, the object is in invalid state.*

### 3.28.1 Detailed Description

Generic media decoder interface.

### 3.28.2 Member Function Documentation

#### 3.28.2.1 void Open ( [VoiceInfo](#) info )

Open (initialize) the decoder.

##### Parameters

<i>info</i>	Properties of the data stream to decode.
-------------	--

Implemented in [OpusCodec.Decoder](#).

### 3.28.3 Property Documentation

#### 3.28.3.1 string Error [get]

If not null, the object is in invalid state.

## 3.29 IDecoderDirect Interface Reference

Interface for a media decoder that synchronously decodes data.

Inherits [IDecoder](#).

Inherited by [OpusCodec.Decoder](#).

### Public Member Functions

- `byte[] DecodeToByte (byte[] buf)`  
*Decode the given raw data buffer.*
- `float[] DecodeToFloat (byte[] buf)`  
*Decode the given raw data buffer to floating point audio.*
- `short[] DecodeToShort (byte[] buf)`  
*Decode the given raw data buffer to 'short' (16-bit) audio.*

### Additional Inherited Members

#### 3.29.1 Detailed Description

Interface for a media decoder that synchronously decodes data.

#### 3.29.2 Member Function Documentation

##### 3.29.2.1 `byte[] DecodeToByte ( byte[] buf )`

Decode the given raw data buffer.

###### Parameters

<i>buf</i>	Buffer of encoded (compressed) data.
------------	--------------------------------------

###### Returns

Buffer of decoded (uncompressed) data.

Implemented in [OpusCodec.Decoder](#).

##### 3.29.2.2 `float[] DecodeToFloat ( byte[] buf )`

Decode the given raw data buffer to floating point audio.

Only sensible for audio data.

###### Parameters

<i>buf</i>	Buffer of encoded (compressed) data.
------------	--------------------------------------

###### Returns

Buffer of decoded (uncompressed) data.

Implemented in [OpusCodec.Decoder](#).

## 3.29.2.3 short [] DecodeToShort ( byte[] buf )

Decode the given raw data buffer to 'short' (16-bit) audio.

Only sensible for audio data.

## Parameters

<i>buf</i>	Buffer of encoded (compressed) data.
------------	--------------------------------------

## Returns

Buffer of decoded (uncompressed) data.

Implemented in [OpusCodec.Decoder](#).

## 3.30 IDecoderQueued Interface Reference

Interface for a media decoder that feeds its data output into a separate method or callback asynchronously, or does not produce output at all.

Inherits [IDecoder](#).

Inherited by [IDecoderQueuedOutputImageNative](#).

### Public Member Functions

- void [Decode](#) (byte[] buf)  
*Decode the given raw data buffer.*

### Additional Inherited Members

#### 3.30.1 Detailed Description

Interface for a media decoder that feeds its data output into a separate method or callback asynchronously, or does not produce output at all.

#### 3.30.2 Member Function Documentation

## 3.30.2.1 void Decode ( byte[] buf )

Decode the given raw data buffer.

This function will be called also for every missing frame, with buf = null.

## Parameters

<i>buf</i>	Buffer of encoded (compressed) data.
------------	--------------------------------------

## 3.31 IDecoderQueuedOutputImageNative Interface Reference

Inherits [IDecoderQueued](#).

## Properties

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

## Additional Inherited Members

### 3.32 IEncoder Interface Reference

Generic media encoder interface.

Inherits IDisposable.

Inherited by [IEncoderDataFlow< T >](#), [IEncoderNativeImageDirect](#), and [IEncoderQueued](#).

## Properties

- string [Error](#) [get]  
*If not null, the object is in invalid state.*

#### 3.32.1 Detailed Description

Generic media encoder interface.

#### 3.32.2 Property Documentation

##### 3.32.2.1 string Error [get]

If not null, the object is in invalid state.

### 3.33 IEncoderDataFlow< T > Interface Template Reference

Interface for a generic media encoder data flow.

Inherits [IEncoder](#).

Inherited by [IEncoderDataFlowDirect< T >](#).

## Additional Inherited Members

#### 3.33.1 Detailed Description

Interface for a generic media encoder data flow.

### 3.34 IEncoderDataFlowDirect< T > Interface Template Reference

Interface for an encoder data flow that synchronously encodes data.

Inherits [IEncoderDataFlow< T >](#).

Inherited by [OpusCodec.Encoder< T >](#).



## Public Member Functions

- `ArraySegment< byte > EncodeAndGetOutput (T[] buf)`  
*Encode the given uncompressed media data.*

## Additional Inherited Members

### 3.34.1 Detailed Description

Interface for an encoder data flow that synchronously encodes data.

### 3.34.2 Member Function Documentation

#### 3.34.2.1 `ArraySegment<byte> EncodeAndGetOutput ( T[] buf )`

Encode the given uncompressed media data.

#### Parameters

<i>buf</i>	Array containing raw (uncompressed) data (e.g. audio samples).
------------	--

#### Returns

Encoded (compressed) data.

Implemented in [OpusCodec.Encoder< T >](#).

## 3.35 IEncoderNativeImageDirect Interface Reference

Inherits [IEncoder](#).

## Public Member Functions

- `IEnumerable< ArraySegment< byte > > EncodeAndGetOutput (IntPtr[] buf, int width, int height, int[] stride, ImageFormat imageFormat, Rotation rotation, Flip flip)`

## Additional Inherited Members

## 3.36 IEncoderQueued Interface Reference

Interface for an encoder data flow that returns compressed data independently (produces output on its own) or asynchronously (usually from a queue).

Inherits [IEncoder](#).

## Public Member Functions

- `IEnumerable< ArraySegment< byte > > GetOutput ()`  
*Get an Enumerable of buffers containing encoded (compressed) data.*

## Additional Inherited Members

### 3.36.1 Detailed Description

Interface for an encoder data flow that returns compressed data independently (produces output on its own) or asynchronously (usually from a queue).

### 3.36.2 Member Function Documentation

#### 3.36.2.1 `IEnumerable<ArraySegment<byte>> GetOutput ( )`

Get an Enumerable of buffers containing encoded (compressed) data.

#### Returns

Encoded (compressed) data.

## 3.37 AudioUtil.ILevelMeter Interface Reference

Audio Level Metering interface.

Inherited by [AudioUtil.LevelMeter< T >](#), and [AudioUtil.LevelMeterDummy](#).

### Public Member Functions

- void [ResetAccumAvgPeakAmp](#) ()  
*Reset [AccumAvgPeakAmp](#).*

### Properties

- float [CurrentAvgAmp](#) [get]  
*Average amplitude value over last half second.*
- float [CurrentPeakAmp](#) [get]  
*Maximum amplitude value over last half second sec.*
- float [AccumAvgPeakAmp](#) [get]  
*Average of CurrentPeakAmps since last reset.*

### 3.37.1 Detailed Description

Audio Level Metering interface.

### 3.37.2 Member Function Documentation

#### 3.37.2.1 `void ResetAccumAvgPeakAmp ( )`

Reset [AccumAvgPeakAmp](#).

Implemented in [AudioUtil.LevelMeter< T >](#), and [AudioUtil.LevelMeterDummy](#).

### 3.37.3 Property Documentation

#### 3.37.3.1 float AccumAvgPeakAmp [get]

Average of CurrentPeakAmps since last reset.

#### 3.37.3.2 float CurrentAvgAmp [get]

Average amplitude value over last half second.

#### 3.37.3.3 float CurrentPeakAmp [get]

Maximum amplitude value over last half second sec.

## 3.38 ILocalVoiceAudio Interface Reference

Interface for an outgoing audio stream.

Inherited by [LocalVoiceAudio< T >](#), and [LocalVoiceAudioDummy](#).

### Public Member Functions

- void [VoiceDetectorCalibrate](#) (int durationMs)  
*Trigger voice detector calibration process.*

### Properties

- [AudioUtil.IVoiceDetector VoiceDetector](#) [get]  
*The VoiceDetector in use.*
- [AudioUtil.ILevelMeter LevelMeter](#) [get]  
*The LevelMeter utility in use.*
- bool [VoiceDetectorCalibrating](#) [get]  
*If true, voice detector calibration is in progress.*

### 3.38.1 Detailed Description

Interface for an outgoing audio stream.

A [LocalVoice](#) always brings a LevelMeter and a VoiceDetector, which you can access using this interface.

### 3.38.2 Member Function Documentation

#### 3.38.2.1 void VoiceDetectorCalibrate ( int durationMs )

Trigger voice detector calibration process.

While calibrating, keep silence. [Voice](#) detector sets threshold based on measured background noise level.

## Parameters

<i>durationMs</i>	Duration of calibration (in milliseconds).
-------------------	--

Implemented in [LocalVoiceAudioDummy](#), and [LocalVoiceAudio< T >](#).

### 3.38.3 Property Documentation

#### 3.38.3.1 [AudioUtil.ILevelMeter](#) **LevelMeter** [get]

The LevelMeter utility in use.

#### 3.38.3.2 [AudioUtil.IVoiceDetector](#) **VoiceDetector** [get]

The VoiceDetector in use.

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

#### 3.38.3.3 **bool** [VoiceDetectorCalibrating](#) [get]

If true, voice detector calibration is in progress.

## 3.39 ILoggable Interface Reference

Inherited by [VoiceComponent](#), and [VoiceConnection](#).

### Properties

- DebugLevel **LogLevel** [get, set]
- [VoiceLogger](#) **Logger** [get]

## 3.40 ILogger Interface Reference

Inherited by [IVoiceFrontend](#), [Logger](#), and [VoiceLogger](#).

### Public Member Functions

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

## 3.41 ImageBufferInfo Class Reference

### Public Member Functions

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

## Properties

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

## 3.42 ImageBufferNative Class Reference

Inherited by [ImageBufferNativeAlloc](#), and [ImageBufferNativeGCHandleSinglePlane](#).

## Public Member Functions

- **ImageBufferNative** ([ImageBufferInfo](#) info)
- virtual void **Release** ()
- virtual void **Dispose** ()

## Properties

- [ImageBufferInfo](#) **Info** [get, protected set]
- IntPtr[] **Planes** [get, protected set]

## 3.43 ImageBufferNativeAlloc Class Reference

Inherits [ImageBufferNative](#), and [IDisposable](#).

## Public Member Functions

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

## Additional Inherited Members

## 3.44 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

### 3.45 ImageBufferNativePool< T > Class Template Reference

Inherits [ObjectPool< T, ImageBufferInfo >](#).

#### Public Member Functions

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

#### Protected Member Functions

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

## Additional Inherited Members

### 3.46 IOSAudioForceToSpeaker Class Reference

Inherits MonoBehaviour.

### 3.47 IProcessor< T > Interface Template Reference

Audio Processor interface.

Inherits IDisposable.

Inherited by [AudioUtil.LevelMeter< T >](#), [AudioUtil.Resampler< T >](#), [AudioUtil.VoiceDetector< T >](#), [AudioUtil.VoiceDetectorCalibration< T >](#), and [AudioUtil.VoiceLevelDetectCalibrate< T >](#).

#### Public Member Functions

- T[] **Process** (T[] buf)  
*Process a frame of audio data.*

#### 3.47.1 Detailed Description

Audio Processor interface.

#### 3.47.2 Member Function Documentation

##### 3.47.2.1 T[] Process ( T[] buf )

Process a frame of audio data.

## Parameters

<i>buf</i>	Buffer containing input audio data
------------	------------------------------------

## Returns

Buffer containing output audio data

Implemented in [AudioUtil.VoiceLevelDetectCalibrate< T >](#), [AudioUtil.VoiceDetector< T >](#), [AudioUtil.VoiceDetectorCalibration< T >](#), [AudioUtil.LevelMeter< T >](#), and [AudioUtil.Resampler< T >](#).

## 3.48 IServiceable Interface Reference

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

Inherited by [BufferReaderPushAdapterBase< T >](#).

### Public Member Functions

- void [Service](#) ([LocalVoice](#) localVoice)  
*Service function that should be called regularly.*

#### 3.48.1 Detailed Description

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

#### 3.48.2 Member Function Documentation

##### 3.48.2.1 void Service ( [LocalVoice](#) localVoice )

Service function that should be called regularly.

Implemented in [BufferReaderPushAdapterAsyncPoolCopy< T >](#), [BufferReaderPushAdapterAsyncPool< T >](#), [BufferReaderPushAdapter< T >](#), and [BufferReaderPushAdapterBase< T >](#).

## 3.49 ISyncAudioOut Interface Reference

Inherits [IAudioOut](#).

Inherited by [UnityAudioOut](#).

### Public Member Functions

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

### Properties

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

## 3.50 AudioUtil.IVoiceDetector Interface Reference

Voice Activity Detector interface.

Inherited by [AudioUtil.VoiceDetector< T >](#), and [AudioUtil.VoiceDetectorDummy](#).

### Properties

- bool [On](#) [get, set]  
*If true, voice detection enabled.*
- float [Threshold](#) [get, set]  
*Voice detected as soon as signal level exceeds threshold.*
- bool [Detected](#) [get]  
*If true, voice detected.*
- DateTime [DetectedTime](#) [get]  
*Last time when switched to detected state.*
- int [ActivityDelayMs](#) [get, set]  
*Keep detected state during this time after signal level dropped below threshold.*

### Events

- Action [OnDetected](#)  
*Called when switched to detected state.*

### 3.50.1 Detailed Description

Voice Activity Detector interface.

### 3.50.2 Property Documentation

#### 3.50.2.1 int ActivityDelayMs [get], [set]

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

#### 3.50.2.2 bool Detected [get]

If true, voice detected.

#### 3.50.2.3 DateTime DetectedTime [get]

Last time when switched to detected state.

#### 3.50.2.4 bool On [get], [set]

If true, voice detection enabled.

#### 3.50.2.5 float Threshold [get], [set]

Voice detected as soon as signal level exceeds threshold.



### 3.50.3 Event Documentation

#### 3.50.3.1 Action OnDetected

Called when switched to detected state.

## 3.51 IVoiceFrontend Interface Reference

Inherits [ILogger](#).

Inherited by [LoadBalancingFrontend](#).

### Public Member Functions

- int **AssignChannel** ([VoiceInfo](#) v)
- bool **IsChannelJoined** (int channelId)
- void **SendVoicesInfo** (IEnumerable< [LocalVoice](#) > voices, int channelId, int targetPlayerId)
- void **SendVoiceRemove** ([LocalVoice](#) voice, int channelId, int targetPlayerId)
- void **SendFrame** (ArraySegment< byte > data, byte evNumber, byte voicId, int channelId, [LocalVoice](#) localVoice)
- string **ChannelIdStr** (int channelId)
- string **PlayerIdStr** (int playerId)
- void **SetDebugEchoMode** ([LocalVoice](#) v)

## 3.52 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[] **buffer**
- int **prevValuesPtr**
- float **accumAvgPeakAmpSum**
- int **accumAvgPeakAmpCount**

## Properties

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

### 3.52.1 Detailed Description

Audio Level Meter.

### 3.52.2 Member Function Documentation

#### 3.52.2.1 `abstract T[] Process ( T[] buf )` [pure virtual]

Process a frame of audio data.

Parameters

<i>buf</i>	Buffer containing input audio data
------------	------------------------------------

Returns

Buffer containing output audio data

Implements [IProcessor< T >](#).

#### 3.52.2.2 `void ResetAccumAvgPeakAmp ( )`

Reset AccumAvgPeakAmp.

Implements [AudioUtil.ILevelMeter](#).

## 3.53 AudioUtil.LevelMeterDummy Class Reference

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

Inherits [AudioUtil.ILevelMeter](#).

## Public Member Functions

- void [ResetAccumAvgPeakAmp](#) ()  
*Reset AccumAvgPeakAmp.*

## Properties

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

### 3.53.1 Detailed Description

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

### 3.53.2 Member Function Documentation

#### 3.53.2.1 void ResetAccumAvgPeakAmp ( )

Reset AccumAvgPeakAmp.

Implements [AudioUtil.LevelMeter](#).

## 3.54 AudioUtil.LevelMeterFloat Class Reference

[LevelMeter](#) specialization for float audio.

Inherits [AudioUtil.LevelMeter< float >](#).

### Public Member Functions

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

### Additional Inherited Members

#### 3.54.1 Detailed Description

[LevelMeter](#) specialization for float audio.

#### 3.54.2 Constructor & Destructor Documentation

##### 3.54.2.1 LevelMeterFloat ( int *samplingRate*, int *numChannels* )

Create new [LevelMeterFloat](#) instance.

Parameters

<i>samplingRate</i>	Sampling rate of the audio signal (in Hz).
<i>numChannels</i>	Number of channels in the audio signal.

## 3.55 AudioUtil.LevelMeterShort Class Reference

[LevelMeter](#) specialization for short audio.

Inherits [AudioUtil.LevelMeter< short >](#).

### Public Member Functions

- [LevelMeterShort](#) (int samplingRate, int numChannels)  
*Create new [LevelMeterShort](#) instance.*
- override short[ ] **Process** (short[ ] buf)

## Additional Inherited Members

### 3.55.1 Detailed Description

[LevelMeter](#) specialization for short audio.

### 3.55.2 Constructor & Destructor Documentation

#### 3.55.2.1 [LevelMeterShort](#) ( int *samplingRate*, int *numChannels* )

Create new [LevelMeterShort](#) instance.

Parameters

<i>samplingRate</i>	Sampling rate of the audio signal (in Hz).
<i>numChannels</i>	Number of channels in the audio signal.

## 3.56 LoadBalancingFrontend Class Reference

Extends [LoadBalancingClient](#) with audio streaming functionality.

Inherits [LoadBalancingClient](#), [IVoiceFrontend](#), and [IDisposable](#).

### 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)
- int **AssignChannel** ([VoiceInfo](#) v)
- bool **IsChannelJoined** (int channelId)
- void **SetDebugEchoMode** ([LocalVoice](#) v)
- [LoadBalancingFrontend](#) (ConnectionProtocol connectionProtocol=[ConnectionProtocol.Udp](#))  
*Initializes a new [LoadBalancingFrontend](#).*
- 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)  
*Change audio groups listened by client. Works only while joined to a voice room.*
- void **SendVoicesInfo** (IEnumerable< [LocalVoice](#) > voices, int channelId, int targetPlayerId)
- void [SendDebugEchoVoicesInfo](#) (int channelId)  
*Send VoicesInfo events to the local player for all voices that have DebugEcho enabled.*
- void **SendVoiceRemove** ([LocalVoice](#) voice, int channelId, int targetPlayerId)
- void **SendFrame** (ArraySegment< byte > data, byte evNumber, byte voiceld, int channelId, [LocalVoice](#) localVoice)
- string **ChannelIdStr** (int channelId)
- string **PlayerIdStr** (int playerId)
- void [Dispose](#) ()  
*Releases all resources used by the [LoadBalancingFrontend](#) instance.*

### Protected Attributes

- [VoiceClient](#) **voiceClient**

## Properties

- [VoiceClient](#) [VoiceClient](#) [get]

The [VoiceClient](#) implementation associated with this [LoadBalancingFrontend](#).

- byte [GlobalAudioGroup](#) [get, set]

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

### 3.56.1 Detailed Description

Extends [LoadBalancingClient](#) with audio streaming functionality.

Use your normal [LoadBalancing](#) workflow to join a [Voice](#) room. All standard [LoadBalancing](#) features are available.

To work with audio:

- Create outgoing audio streams with [Client.CreateLocalVoice](#).
- Handle new incoming audio streams info with [OnRemoteVoiceInfoAction](#) .
- Handle incoming audio streams data with [OnAudioFrameAction](#) .
- Handle closing of incoming audio streams with .

### 3.56.2 Constructor & Destructor Documentation

#### 3.56.2.1 [LoadBalancingFrontend](#) ( [ConnectionProtocol](#) *connectionProtocol* = [ConnectionProtocol.Udp](#) )

Initializes a new [LoadBalancingFrontend](#).

Parameters

<i>connection↔ Protocol</i>	Connection protocol (UDP or TCP). <a href="#">ConnectionProtocol</a>
---------------------------------	--

### 3.56.3 Member Function Documentation

#### 3.56.3.1 virtual bool [ChangeAudioGroups](#) ( byte[] *groupsToRemove*, byte[] *groupsToAdd* ) [virtual]

Change audio groups listened by client. Works only while joined to a voice room.

[LocalVoice.Group](#) [SetGlobalAudioGroup](#)(byte)

Note the difference between passing null and [byte\[0\]](#): null won't add/remove any groups. [byte\[0\]](#) will add/remove all (existing) groups. First, removing groups is executed. This way, you could leave all groups and join only the ones provided.

Parameters

<i>groupsTo↔ Remove</i>	Groups to remove from listened. Null will not leave any. A <a href="#">byte[0]</a> will remove all.
<i>groupsToAdd</i>	Groups to add to listened. Null will not add any. A <a href="#">byte[0]</a> will add all current.

Returns

If request could be enqueued for sending

### 3.56.3.2 void Dispose ( )

Releases all resources used by the [LoadBalancingFrontend](#) instance.

### 3.56.3.3 void SendDebugEchoVoicesInfo ( int *channelId* )

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

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

#### Parameters

<i>channelId</i>	Transport Channel ID
------------------	----------------------

### 3.56.3.4 new void Service ( )

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

## 3.56.4 Property Documentation

### 3.56.4.1 byte GlobalAudioGroup [get], [set]

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

[LocalVoice.Group](#) [ChangeAudioGroups\(byte\[\], byte\[\]\)](#)

### 3.56.4.2 VoiceClient VoiceClient [get]

The [VoiceClient](#) implementation associated with this [LoadBalancingFrontend](#).

## 3.57 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** ()

### Public Attributes

- const int **DATA\_POOL\_CAPACITY** = 50

## Protected Member Functions

- void **resetNoTransmitCnt** ()

## Protected Attributes

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

## Properties

- byte [Group](#) [get, set]  
*If AudioGroup != 0, voice's data is sent only to clients listening to this group.*
- [VoiceInfo](#) [Info](#) [get]  
*Returns Info structure assigned on local voice cration.*
- bool [TransmitEnabled](#) [get, set]  
*If true, stream data broadcasted.*
- bool [IsCurrentlyTransmitting](#) [get, protected set]  
*Returns true if stream broadcasts.*
- int [FramesSent](#) [get]  
*Sent frames counter.*
- int [FramesSentBytes](#) [get]  
*Sent frames bytes counter.*
- bool [Reliable](#) [get, set]  
*Send data reliable.*
- bool [Encrypt](#) [get, set]  
*Send data encrypted.*
- object [LocalUserObject](#) [get, set]  
*Optional user object attached to [LocalVoice](#).*
- [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 frontend.*

### 3.57.1 Detailed Description

Represents outgoing data stream.

### 3.57.2 Member Function Documentation

#### 3.57.2.1 void RemoveSelf ( )

Remove this voice from it's [VoiceClient](#) (using [VoiceClient.RemoveLocalVoice](#)

.

### 3.57.3 Property Documentation

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

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

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

Send data encrypted.

#### 3.57.3.3 `int FramesSent` `[get]`

Sent frames counter.

#### 3.57.3.4 `int FramesSentBytes` `[get]`

Sent frames bytes counter.

#### 3.57.3.5 `byte Group` `[get]`, `[set]`

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

[LoadBalancingFrontend.ChangeAudioGroups\(byte\[\], byte\[\]\)](#)

#### 3.57.3.6 `VoiceInfo Info` `[get]`

Returns `Info` structure assigned on local voice cration.

#### 3.57.3.7 `bool IsCurrentlyTransmitting` `[get]`, `[protected set]`

Returns true if stream broadcasts.

#### 3.57.3.8 `object LocalUserObject` `[get]`, `[set]`

Optional user object attached to [LocalVoice](#).

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

Optional user object attached to [LocalVoice](#). its `Service()` will be called at each [VoiceClient.Service\(\)](#) call.

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

Send data reliable.

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

If true, stream data broadcasted.



## 3.58 LocalVoiceAudio< T > Class Template Reference

Outgoing audio stream.

Inherits [LocalVoiceFramed< T >](#), and [ILocalVoiceAudio](#).

### Public Member Functions

- void [VoiceDetectorCalibrate](#) (int durationMs)  
*Trigger voice detector calibration process.*

### Static Public Member Functions

- static [LocalVoiceAudio< T > Create](#) ([VoiceClient](#) voiceClient, byte voiceld, [IEncoder](#) encoder, [VoiceInfo](#) voiceInfo, 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**
- int **sourceSamplingRateHz**
- bool **resampleSource**

### Properties

- virtual [AudioUtil.IVoiceDetector](#) **VoiceDetector** [get]
- virtual [AudioUtil.ILevelMeter](#) **LevelMeter** [get]
- bool [VoiceDetectorCalibrating](#) [get]  
*True if the VoiceDetector is currently calibrating.*

### Additional Inherited Members

#### 3.58.1 Detailed Description

Outgoing audio stream.

#### 3.58.2 Member Function Documentation

- 3.58.2.1 static [LocalVoiceAudio<T> Create](#) ( [VoiceClient](#) voiceClient, byte voiceld, [IEncoder](#) encoder, [VoiceInfo](#) voiceInfo, int channelId ) [static]

Create a new LocalVoiceAudio<T> instance.

## Parameters

<i>voiceClient</i>	The <a href="#">VoiceClient</a> to use for this outgoing stream.
<i>voiceId</i>	Numeric ID for this voice.
<i>encoder</i>	Encoder to use for this voice.
<i>channelId</i>	<a href="#">Voice</a> transport channel ID to use for this voice.

## Returns

The new `LocalVoiceAudio<T>` instance.

### 3.58.2.2 void VoiceDetectorCalibrate ( int durationMs )

Trigger voice detector calibration process.

While calibrating, keep silence. [Voice](#) detector sets threshold basing on measured background noise level.

## Parameters

<i>durationMs</i>	Duration of calibration in milliseconds.
-------------------	--

Implements [ILocalVoiceAudio](#).

## 3.58.3 Property Documentation

### 3.58.3.1 bool VoiceDetectorCalibrating [get]

True if the `VoiceDetector` is currently calibrating.

## 3.59 LocalVoiceAudioDummy Class Reference

Dummy [LocalVoiceAudio](#)

Inherits [LocalVoice](#), and [ILocalVoiceAudio](#).

### Public Member Functions

- void [VoiceDetectorCalibrate](#) (int durationMs)  
*Trigger voice detector calibration process.*

### Static Public Attributes

- static [LocalVoiceAudioDummy Dummy](#) = new [LocalVoiceAudioDummy](#)()  
*A Dummy [LocalVoiceAudio](#) instance.*

### Properties

- [AudioUtil.IVoiceDetector VoiceDetector](#) [get]
- [AudioUtil.ILevelMeter LevelMeter](#) [get]
- bool [VoiceDetectorCalibrating](#) [get]

## Additional Inherited Members

### 3.59.1 Detailed Description

Dummy [LocalVoiceAudio](#)

For testing, this [LocalVoiceAudio](#) implementation features a [AudioUtil.VoiceDetectorDummy](#) and a [AudioUtil.LevelMeterDummy](#)

### 3.59.2 Member Function Documentation

#### 3.59.2.1 void VoiceDetectorCalibrate ( int *durationMs* )

Trigger voice detector calibration process.

While calibrating, keep silence. [Voice](#) detector sets threshold based on measured background noise level.

Parameters

<i>durationMs</i>	Duration of calibration (in milliseconds).
-------------------	--

Implements [ILocalVoiceAudio](#).

### 3.59.3 Member Data Documentation

#### 3.59.3.1 LocalVoiceAudioDummy Dummy = new LocalVoiceAudioDummy() [static]

A Dummy [LocalVoiceAudio](#) instance.

## 3.60 LocalVoiceAudioFloat Class Reference

Specialization of [LocalVoiceAudio](#) for float audio

Inherits [LocalVoiceAudio< float >](#).

## Additional Inherited Members

### 3.60.1 Detailed Description

Specialization of [LocalVoiceAudio](#) for float audio

## 3.61 LocalVoiceAudioShort Class Reference

Specialization of [LocalVoiceAudio](#) for short audio

Inherits [LocalVoiceAudio< short >](#).

## Additional Inherited Members

### 3.61.1 Detailed Description

Specialization of [LocalVoiceAudio](#) for short audio

## 3.62 LocalVoiceFramed< T > Class Template Reference

Typed re-framing [LocalVoice](#)

Inherits [LocalVoiceFramedBase](#).

Inherited by [LocalVoiceAudio< T >](#).

### Public Member Functions

- void [AddPostProcessor](#) (params [IProcessor< T >](#)[] processors)  
*Adds processors after any built-in processors and everything added with AddPreProcessor.*
- void [AddPreProcessor](#) (params [IProcessor< T >](#)[] processors)  
*Adds processors before built-in processors and everything added with AddPostProcessor.*
- void [ClearProcessors](#) ()  
*Clears all processors in pipeline including built-in resampling. User should add at least resampler processor after call.*
- void [PushDataAsync](#) (T[] buf)  
*Asynchronously push data into this stream.*
- void [PushData](#) (T[] buf)  
*Synchronously push data into this stream.*
- override void [Dispose](#) ()  
*Releases resources used by the VoiceFramed instance. Buffers used for asynchronous push will be disposed in encoder thread's 'finally'.*

### Properties

- [FactoryPrimitiveArrayPool< T >](#) **BufferFactory** [get]
- bool [PushDataAsyncReady](#) [get]  
*Whether this [LocalVoiceFramed](#) has capacity for more data buffers to be pushed asynchronously.*

### Additional Inherited Members

#### 3.62.1 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

<i>voiceInfo</i>	Outgoing stream parameters. Set applicable fields to read them by encoder and by receiving client when voice created.
<i>channelId</i>	Transport channel specific to frontend. Set to VoiceClient.ChannelAuto to let frontend automatically assign channel.
<i>encoder</i>	Encoder producing the stream.

#### Returns

Outgoing stream handler.

#### 3.62.2 Member Function Documentation

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

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

## Parameters

<i>processors</i>	
-------------------	--

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

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

## Parameters

<i>processors</i>	
-------------------	--

## 3.62.2.3 void ClearProcessors ( )

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

## 3.62.2.4 override void Dispose ( ) [virtual]

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

Reimplemented from [LocalVoice](#).

3.62.2.5 void PushData ( T[] *buf* )

Synchronously push data into this stream.

3.62.2.6 void PushDataAsync ( T[] *buf* )

Asynchronously push data into this stream.

## 3.62.3 Property Documentation

## 3.62.3.1 bool PushDataAsyncReady [get]

Whether this [LocalVoiceFramed](#) has capacity for more data buffers to be pushed asynchronously.

## 3.63 LocalVoiceFramedBase Class Reference

Typed re-framing [LocalVoice](#)

Inherits [LocalVoice](#).

Inherited by [LocalVoiceFramed< T >](#).

## Properties

- int [FrameSize](#) [get]

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

## Additional Inherited Members

### 3.63.1 Detailed Description

Typed re-framing [LocalVoice](#)

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

### 3.63.2 Property Documentation

#### 3.63.2.1 int FrameSize [get]

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

## 3.64 Logger Class Reference

Inherits [ILogger](#).

### Public Member Functions

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

## 3.65 MicWrapper Class Reference

Inherits [IAudioReader](#)<float>.

### Public Member Functions

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

### Properties

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

## 3.66 ObjectFactory< TType, TInfo > Interface Template Reference

Uniform interface to [ObjectPool](#)<TType, TInfo> and single reusable object.

Inherits [IDisposable](#).

## Public Member Functions

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

## Properties

- TInfo **Info** [get]

### 3.66.1 Detailed Description

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

#### Template Parameters

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

## 3.67 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 associated with this **ObjectPool***

## Protected Member Functions

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

## Protected Attributes

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

## Properties

- TInfo **Info** [get]  
*The property (info) that objects in this Pool must match.*

### 3.67.1 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

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

### 3.67.2 Constructor & Destructor Documentation

#### 3.67.2.1 **ObjectPool** ( int *capacity*, string *name* )

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

##### Parameters

<i>capacity</i>	Capacity (size) of the object pool.
<i>name</i>	Name of the object pool.

#### 3.67.2.2 **ObjectPool** ( int *capacity*, string *name*, TInfo *info* )

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

##### Parameters

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

### 3.67.3 Member Function Documentation

#### 3.67.3.1 TType **AcquireOrCreate** ( )

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

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

#### 3.67.3.2 TType **AcquireOrCreate** ( TInfo *info* )

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



## Parameters

<i>info</i>	Info structure to match, or create a new object with.
-------------	---

## 3.67.3.3 void Dispose ( )

Free resources associated with this [ObjectPool](#)

3.67.3.4 void Init ( TInfo *info* )

(Re-)Initializes this [ObjectPool](#).

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

## Parameters

<i>info</i>	Info about this Pool's objects.
-------------	---------------------------------

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

Returns object to pool.

## Parameters

<i>obj</i>	The object to return to the pool.
<i>objInfo</i>	The info structure about obj.

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

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

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

## Parameters

<i>obj</i>	The object to return to the pool.
------------	-----------------------------------

## 3.67.4 Property Documentation

## 3.67.4.1 TInfo Info [get]

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

## 3.68 OpusCodec Class Reference

## Classes

- class [Decoder](#)
- class [Encoder](#)
- class [EncoderFactory](#)
- class [EncoderFloat](#)
- class [EncoderShort](#)
- class [Util](#)

## Public Types

- enum **FrameDuration**

## 3.69 OpusDecoder Class Reference

Inherits IDisposable.

### Public Member Functions

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

### Properties

- string **Version** [get]
- [Bandwidth](#) **PreviousPacketBandwidth** [get]

## 3.70 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** ()

### Public Attributes

- const int **BitrateMax** = -1

### Properties

- SamplingRate **InputSamplingRate** [get]
- [Channels](#) **InputChannels** [get]
- string **Version** [get]
- [Delay](#) **EncoderDelay** [get, set]  
*Using a duration of less than 10 ms will prevent the encoder from using the LPC or hybrid modes.*
- int **FrameSizePerChannel** [get]
- int **Bitrate** [get, set]
- [Bandwidth](#) **MaxBandwidth** [get, set]
- Complexity **Complexity** [get, set]
- int **ExpectedPacketLossPercentage** [get, set]
- [SignalHint](#) **SignalHint** [get, set]

- ForceChannels **ForceChannels** [get, set]
- bool **UseInbandFEC** [get, set]
- bool **UseUnconstrainedVBR** [get, set]
- bool **DtxEnabled** [get, set]

### 3.70.1 Property Documentation

#### 3.70.1.1 Delay EncoderDelay [get], [set]

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

## 3.71 OpusException Class Reference

Inherits Exception.

### Public Member Functions

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

### Properties

- OpusStatusCode **StatusCodes** [get]

## 3.72 WebRTCAudioLib.Param Struct Reference

### Public Attributes

- const int **REVERSE\_STREAM\_DELAY\_MS** = 1
- const int **AEC** = 10
- const int **AEC\_SUPPRESSION\_LEVEL** = 11
- const int **AECM** = 20
- const int **AECM\_ROUTING\_MODE** = 21
- const int **AECM\_COMFORT\_NOISE** = 22
- const int **HIGH\_PASS\_FILTER** = 31
- const int **NS** = 41
- const int **NS\_LEVEL** = 42
- const int **AGC** = 51
- const int **AGC\_MODE** = 52
- const int **AGC\_COMPRESSION\_GAIN** = 56
- const int **AGC\_LIMITER** = 57
- const int **VAD** = 61
- const int **VAD\_FRAME\_SIZE\_MS** = 62
- const int **VAD\_LIKEHOOD** = 63

## 3.73 Recorder.PhotonVoiceCreatedParams Class Reference

### Properties

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

## 3.74 PhotonVoiceLagSimulationGui Class Reference

Inherits MonoBehaviour.

### Public Member Functions

- void **Start** ()
- void **OnGUI** ()

### Public Attributes

- Rect **WindowRect** = new Rect(0, 100, 120, 100)  
*Positioning rect for window.*
- int **WindowId** = 101  
*Unity GUI Window ID (must be unique or will cause issues).*
- bool **Visible** = true  
*Shows or hides GUI (does not affect settings).*

### Properties

- PhotonPeer **Peer** [get, set]  
*The peer currently in use (to set the network simulation).*

#### 3.74.1 Member Data Documentation

##### 3.74.1.1 bool Visible = true

Shows or hides GUI (does not affect settings).

##### 3.74.1.2 int WindowId = 101

Unity GUI Window ID (must be unique or will cause issues).

##### 3.74.1.3 Rect WindowRect = new Rect(0, 100, 120, 100)

Positioning rect for window.

#### 3.74.2 Property Documentation

##### 3.74.2.1 PhotonPeer Peer [get], [set]

The peer currently in use (to set the network simulation).

## 3.75 PhotonVoiceNetwork Class Reference

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

Inherits [VoiceConnection](#).

## Public Member Functions

- bool [ConnectAndJoinRoom](#) ()  
*Connect voice client to [Photon](#) servers and join a [Voice](#) room*
- void [Disconnect](#) ()  
*Disconnect voice client from all [Photon](#) servers*

## Public Attributes

- const string [VoiceRoomNameSuffix](#) = "\_voice\_"  
*Suffix for voice room names appended to [PUN](#) room names.*
- bool [AutoConnectAndJoin](#) = true  
*Auto connect voice client and join a voice room when [PUN](#) client is joined to a [PUN](#) room*
- bool [AutoLeaveAndDisconnect](#) = true  
*Auto disconnect voice client when [PUN](#) client is not joined to a [PUN](#) room*
- bool [AutoCreateSpeakerIfNotFound](#) = true  
*Auto instantiate a [GameObject](#) and attach a [Speaker](#) component to link to a remote audio stream if no candidate could be found*

## Protected Member Functions

- override void **Awake** ()
- override void **OnApplicationQuit** ()
- override void **OnDestroy** ()

## Properties

- static [PhotonVoiceNetwork Instance](#) [get, set]  
*Singleton instance for [PhotonVoiceNetwork](#)*

## Additional Inherited Members

### 3.75.1 Detailed Description

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

### 3.75.2 Member Function Documentation

#### 3.75.2.1 bool [ConnectAndJoinRoom](#) ( )

Connect voice client to [Photon](#) servers and join a [Voice](#) room

#### Returns

If true, connection command send from client

#### 3.75.2.2 void [Disconnect](#) ( )

Disconnect voice client from all [Photon](#) servers

### 3.75.3 Member Data Documentation

#### 3.75.3.1 `bool AutoConnectAndJoin = true`

Auto connect voice client and join a voice room when [PUN](#) client is joined to a [PUN](#) room

#### 3.75.3.2 `bool AutoCreateSpeakerIfNotFound = true`

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

#### 3.75.3.3 `bool AutoLeaveAndDisconnect = true`

Auto disconnect voice client when [PUN](#) client is not joined to a [PUN](#) room

#### 3.75.3.4 `const string VoiceRoomNameSuffix = "_voice_"`

Suffix for voice room names appended to [PUN](#) room names.

### 3.75.4 Property Documentation

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

Singleton instance for [PhotonVoiceNetwork](#)

## 3.76 PhotonVoiceView Class Reference

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

Inherits [VoiceComponent](#).

### Public Attributes

- `bool AutoCreateRecorderIfNotFound`  
*If true, a `Recorder` component will be added to the same `GameObject` if not found already.*
- `bool UsePrimaryRecorder`  
*If true, [PhotonVoiceNetwork.PrimaryRecorder](#) will be used by this [PhotonVoiceView](#)*
- `bool SetupDebugSpeaker`  
*If true, a `Speaker` component will be setup to be used for the `DebugEcho` mode*

### Protected Member Functions

- override void **Awake** ()

## Properties

- [Recorder RecorderInUse](#) [get, set]  
The Recorder component currently used by this [PhotonVoiceView](#)
- [Speaker SpeakerInUse](#) [get, set]  
The Speaker component currently used by this [PhotonVoiceView](#)
- bool [IsSetup](#) [get, protected set]  
If true, this [PhotonVoiceView](#) is setup and ready to be used
- bool [IsSpeaker](#) [get, protected set]  
If true, this [PhotonVoiceView](#) has a Speaker setup for playback of received audio frames from remote audio source
- bool [IsSpeaking](#) [get]  
If true, this [PhotonVoiceView](#) has a Speaker that is currently playing received audio frames from remote audio source
- bool [IsRecorder](#) [get, protected set]  
If true, this [PhotonVoiceView](#) has a Recorder setup for transmission of audio stream from local audio source
- bool [IsRecording](#) [get]  
If true, this [PhotonVoiceView](#) has a Recorder that is currently transmitting audio stream from local audio source

## Additional Inherited Members

### 3.76.1 Detailed Description

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

### 3.76.2 Member Data Documentation

#### 3.76.2.1 bool AutoCreateRecorderIfNotFound

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

#### 3.76.2.2 bool SetupDebugSpeaker

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

#### 3.76.2.3 bool UsePrimaryRecorder

If true, [PhotonVoiceNetwork.PrimaryRecorder](#) will be used by this [PhotonVoiceView](#)

### 3.76.3 Property Documentation

#### 3.76.3.1 bool IsRecorder [get], [protected set]

If true, this [PhotonVoiceView](#) has a Recorder setup for transmission of audio stream from local audio source

#### 3.76.3.2 bool IsRecording [get]

If true, this [PhotonVoiceView](#) has a Recorder that is currently transmitting audio stream from local audio source

### 3.76.3.3 `bool IsSetup` `[get], [protected set]`

If true, this [PhotonVoiceView](#) is setup and ready to be used

### 3.76.3.4 `bool IsSpeaker` `[get], [protected set]`

If true, this [PhotonVoiceView](#) has a Speaker setup for playback of received audio frames from remote audio source

### 3.76.3.5 `bool IsSpeaking` `[get]`

If true, this [PhotonVoiceView](#) has a Speaker that is currently playing received audio frames from remote audio source

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

The Recorder component currently used by this [PhotonVoiceView](#)

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

The Speaker component currently used by this [PhotonVoiceView](#)

## 3.77 `PrimitiveArrayPool< T >` Class Template Reference

Pool of Arrays with components of type T, with [ObjectPool](#) info being the array's size.

Inherits [ObjectPool< T\[\], int >](#).

### Public Member Functions

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

### Protected Member Functions

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

### Additional Inherited Members

#### 3.77.1 Detailed Description

Pool of Arrays with components of type T, with [ObjectPool](#) info being the array's size.

#### Template Parameters

<code>T</code>	Array element type.
----------------	---------------------



## 3.78 Recorder Class Reference

Component representing outgoing audio stream in scene.

Inherits [VoiceComponent](#).

### Classes

- class [PhotonVoiceCreatedParams](#)

### Public Types

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

### Public Member Functions

- void [Init](#) ([VoiceClient](#) voiceClient, object customObj=null)  
*Initializes the [Recorder](#) component to be able to transmit audio.*
- void [ReInit](#) ()  
*Reinitializes the [Recorder](#) if something has changed that requires this.*
- void [VoiceDetectorCalibrate](#) (int durationMs)  
*Trigger voice detector calibration process. While calibrating, keep silence. [Voice](#) detector sets threshold basing on measured background noise level.*

### 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.*
- bool [RequiresInit](#) [get]  
*Returns true if something has changed in the [Recorder](#) after initialization that won't take effect unless re initialized.*
- 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.*
  - bool [IsCurrentlyTransmitting](#) [get]  
*Returns true if audio stream broadcasts.*
  - [AudioUtil.ILevelMeter](#) [LevelMeter](#) [get]  
*Level meter utility.*
  - bool [VoiceDetectorCalibrating](#) [get]  
*If true, voice detector calibration is in progress.*
  - [ILocalVoiceAudio](#) [voiceAudio](#) [get]
  - InputSourceType [SourceType](#) [get, set]  
*Audio data source.*
  - MicType [MicrophoneType](#) [get, set]  
*Which microphone API to use when the Source is set to Microphone.*
  - SampleTypeConv [TypeConvert](#) [get, set]  
*Force creation of 'short' pipeline and convert audio data to short for 'float' audio sources.*
  - AudioClip [AudioClip](#) [get, set]  
*Source audio clip.*
  - bool [LoopAudioClip](#) [get, set]  
*Loop playback for audio clip sources.*
  - POpusCodec.Enums.SamplingRate [SamplingRate](#) [get, set]  
*Outgoing audio stream sampling rate.*
  - OpusCodec.FrameDuration [FrameDuration](#) [get, set]  
*Outgoing audio stream encoder delay.*
  - int [Bitrate](#) [get, set]  
*Outgoing audio stream bitrate.*

## Additional Inherited Members

### 3.78.1 Detailed Description

Component representing outgoing audio stream in scene.

### 3.78.2 Member Function Documentation

#### 3.78.2.1 void Init ( VoiceClient voiceClient, object customObj = null )

Initializes the [Recorder](#) component to be able to transmit audio.

## Parameters

<i>voiceClient</i>	The <a href="#">VoiceClient</a> to be used with this <a href="#">Recorder</a> .
<i>customObj</i>	Optional user data object to be transmitted with the voice stream info

## 3.78.2.2 void ReInit ( )

Reinitializes the [Recorder](#) if something has changed that requires this.

## 3.78.2.3 void VoiceDetectorCalibrate ( int durationMs )

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

## Parameters

<i>durationMs</i>	Duration of calibration in milliseconds.
-------------------	--

## 3.78.3 Property Documentation

## 3.78.3.1 AudioClip AudioClip [get], [set]

Source audio clip.

## 3.78.3.2 byte AudioGroup [get], [set]

Target interest group that will receive transmitted audio.

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

## 3.78.3.3 int Bitrate [get], [set]

Outgoing audio stream bitrate.

## 3.78.3.4 bool DebugEchoMode [get], [set]

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

## 3.78.3.5 bool Encrypt [get], [set]

If true, voice stream is sent encrypted.

## 3.78.3.6 OpusCodec.FrameDuration FrameDuration [get], [set]

Outgoing audio stream encoder delay.

## 3.78.3.7 Func&lt;IAudioDesc&gt; InputFactory [get], [set]

Set the method returning new [Voice.IAudioDesc](#) instance to be assigned to a new voice created with Source set to Factory

**3.78.3.8 bool IsCurrentlyTransmitting** [get]

Returns true if audio stream broadcasts.

**3.78.3.9 bool IsInitialized** [get]

If true, this [Recorder](#) has been initialized and is ready to transmit to remote clients.

**3.78.3.10 AudioUtil.ILevelMeter LevelMeter** [get]

Level meter utility.

**3.78.3.11 bool LoopAudioClip** [get], [set]

Loop playback for audio clip sources.

**3.78.3.12 MicType MicrophoneType** [get], [set]

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

**3.78.3.13 int PhotonMicrophoneDeviceId** [get], [set]

Set or get photon microphone device used for streaming.

**3.78.3.14 AudioInEnumerator PhotonMicrophoneEnumerator** [static], [get]

Enumerator for the available microphone devices gathered by the [Photon](#) plugin.

**3.78.3.15 bool ReliableMode** [get], [set]

If true, stream data sent in reliable mode.

**3.78.3.16 bool RequiresInit** [get]

Returns true if something has changed in the [Recorder](#) after initialization that won't take effect unless re initialized.

**3.78.3.17 POpusCodec.Enums.SamplingRate SamplingRate** [get], [set]

Outgoing audio stream sampling rate.

**3.78.3.18 InputSourceType SourceType** [get], [set]

Audio data source.

**3.78.3.19 bool TransmitEnabled** [get], [set]

If true, audio transmission is enabled.

**3.78.3.20 SampleTypeConv TypeConvert** [get], [set]

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

**3.78.3.21 string UnityMicrophoneDevice** [get], [set]

Set or get [Unity](#) microphone device used for streaming.

**3.78.3.22 object UserData** [get], [set]

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

**3.78.3.23 bool VoiceDetection** [get], [set]

If true, voice detection enabled.

**3.78.3.24 int VoiceDetectionDelayMs** [get], [set]

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

**3.78.3.25 float VoiceDetectionThreshold** [get], [set]

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

**3.78.3.26 AudioUtil.IVoiceDetector VoiceDetector** [get]

Returns voice activity detector for recorder's audio stream.

**3.78.3.27 bool VoiceDetectorCalibrating** [get]

If true, voice detector calibration is in progress.

## 3.79 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).*
- object [LocalUserObject](#) [get]  
*Object set by user when remote voice created.*

### 3.79.1 Detailed Description

Information about a remote voice (incoming stream).

### 3.79.2 Property Documentation

#### 3.79.2.1 `int ChannelId` [get]

ID of channel used for transmission.

#### 3.79.2.2 `VoiceInfo Info` [get]

Remote voice info.

#### 3.79.2.3 `object LocalUserObject` [get]

Object set by user when remote voice created.

#### 3.79.2.4 `int PlayerId` [get]

Player ID of voice owner.

#### 3.79.2.5 `byte VoiceId` [get]

Voice ID (unique in the room).

## 3.80 RemoteVoiceOptions Struct Reference

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

### Properties

- Action< byte[]> [OnDecodedFrameByteAction](#) [get, set]  
*Register a method to be called when new data frame received. Use it to get uncompressed data as byte[].*
- Action< float[]> [OnDecodedFrameFloatAction](#) [get, set]  
*Register a method to be called when new data frame received. Use it to get uncompressed data as float[].*
- Action< short[]> [OnDecodedFrameShortAction](#) [get, set]  
*Register a method to be called when new data frame received. Use it to get uncompressed data as short[].*
- Action [OnRemoteVoiceRemoveAction](#) [get, set]  
*Register a method to be called when the remote voice is removed.*
- object [LocalUserObject](#) [get, set]  
*User object (e.g. audio player) attached to remote voice instance for easy access.*
- [IDecoder Decoder](#) [get, set]  
*Remote voice data decoder. Use to set decoder options or override it with user decoder.*

### 3.80.1 Detailed Description

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

### 3.80.2 Property Documentation

#### 3.80.2.1 IDecoder Decoder [get], [set]

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

#### 3.80.2.2 object LocalUserObject [get], [set]

User object (e.g. audio player) attached to remote voice instance for easy access.

#### 3.80.2.3 Action<byte[]> OnDecodedFrameByteAction [get], [set]

Register a method to be called when new data frame received. Use it to get uncompressed data as byte[].

#### 3.80.2.4 Action<float[]> OnDecodedFrameFloatAction [get], [set]

Register a method to be called when new data frame received. Use it to get uncompressed data as float[].

#### 3.80.2.5 Action<short[]> OnDecodedFrameShortAction [get], [set]

Register a method to be called when new data frame received. Use it to get uncompressed data as short[].

#### 3.80.2.6 Action OnRemoteVoiceRemoveAction [get], [set]

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

## 3.81 AudioUtil.Resampler< T > Class Template Reference

Sample-rate conversion Audio Processor.

Inherits [IProcessor< T >](#).

### Public Member Functions

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

### Protected Attributes

- T[] **frameResampled**

### 3.81.1 Detailed Description

Sample-rate conversion Audio Processor.

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

### 3.81.2 Constructor & Destructor Documentation

#### 3.81.2.1 Resampler ( int *dstSize*, int *channels* )

Create a new [Resampler](#) instance.

##### Parameters

<i>dstSize</i>	Frame size of a destination frame. Determines output rate.
<i>channels</i>	Number of audio channels expected in both in- and output.

### 3.81.3 Member Function Documentation

#### 3.81.3.1 T[] Process ( T[] *buf* )

Process a frame of audio data.

##### Parameters

<i>buf</i>	Buffer containing input audio data
------------	------------------------------------

##### Returns

Buffer containing output audio data

Implements [IProcessor< T >](#).

## 3.82 Speaker Class Reference

Component representing remote audio stream in local scene.

Inherits [VoiceComponent](#).

### Public Attributes

- int **PlayDelayMs** = 200

### Protected Member Functions

- override void **Awake** ()

### Properties

- bool [IsPlaying](#) [get]  
*Is the speaker playing right now.*
- int [Lag](#) [get]  
*Smoothed difference between (jittering) stream and (clock-driven) audioOutput.*
- Action< [Speaker](#) > [OnRemoteVoiceRemoveAction](#) [get, set]  
*Register a method to be called when remote voice removed.*
- Realtime.Player [Actor](#) [get, set]  
*Per room, the connected users/players are represented with a Realtime.Player, also known as Actor.*



## Additional Inherited Members

### 3.82.1 Detailed Description

Component representing remote audio stream in local scene.

### 3.82.2 Property Documentation

#### 3.82.2.1 `Realtime.Player Actor` `[get]`, `[set]`

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

[Photon Voice](#) calls this Actor, to avoid a name-clash with the `Player` class in [Voice](#).

#### 3.82.2.2 `bool IsPlaying` `[get]`

Is the speaker playing right now.

#### 3.82.2.3 `int Lag` `[get]`

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

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

Register a method to be called when remote voice removed.

## 3.83 SpeexLib Class Reference

Inherited by [SpeexProcessor](#).

### Public Member Functions

- static `IntPtr speex_preprocess_state_init` (int frame\_size, int sampling\_rate)
- static void `speex_preprocess_state_destroy` (IntPtr st)
- static int `speex_preprocess_run` (IntPtr st, short[] x)
- static int `speex_preprocess_ctl` (IntPtr st, int request, IntPtr ptr)
- static `IntPtr speex_echo_state_init` (int frame\_size, int filter\_length)
- static `IntPtr speex_echo_state_init_mc` (int frame\_size, int filter\_length, int nb\_mic, int nb\_speakers)
- static void `speex_echo_state_destroy` (IntPtr st)
- static void `speex_echo_cancellation` (IntPtr st, short[] rec, short[] play, short[] outBuf)
- static void `speex_echo_capture` (IntPtr st, short[] rec, short[] outBuf)
- static void `speex_echo_playback` (IntPtr st, short[] play)
- static void `speex_echo_state_reset` (IntPtr st)
- static int `speex_echo_ctl` (IntPtr st, int request, IntPtr ptr)

### Static Public Member Functions

- static int `speex_preprocess_ctl` (IntPtr st, int request, ref int value)
- static int `speex_preprocess_ctl` (IntPtr st, int request, ref float value)
- static int `speex_echo_ctl` (IntPtr st, int request, ref int value)
- static int `speex_echo_ctl` (IntPtr st, int request, ref float value)

## Public Attributes

- const int [SPEEX\\_PREPROCESS\\_SET\\_DENOISE](#) = 0
- const int [SPEEX\\_PREPROCESS\\_GET\\_DENOISE](#) = 1
- const int [SPEEX\\_PREPROCESS\\_SET\\_AGC](#) = 2
- const int [SPEEX\\_PREPROCESS\\_GET\\_AGC](#) = 3
- const int [SPEEX\\_PREPROCESS\\_SET\\_VAD](#) = 4
- const int [SPEEX\\_PREPROCESS\\_GET\\_VAD](#) = 5
- const int [SPEEX\\_PREPROCESS\\_SET\\_AGC\\_LEVEL](#) = 6
- const int [SPEEX\\_PREPROCESS\\_GET\\_AGC\\_LEVEL](#) = 7
- const int [SPEEX\\_PREPROCESS\\_SET\\_DEREVERB](#) = 8
- const int [SPEEX\\_PREPROCESS\\_GET\\_DEREVERB](#) = 9
- const int [SPEEX\\_PREPROCESS\\_SET\\_DEREVERB\\_LEVEL](#) = 10
- const int [SPEEX\\_PREPROCESS\\_GET\\_DEREVERB\\_LEVEL](#) = 11
- const int [SPEEX\\_PREPROCESS\\_SET\\_DEREVERB\\_DECAY](#) = 12
- const int [SPEEX\\_PREPROCESS\\_GET\\_DEREVERB\\_DECAY](#) = 13
- const int [SPEEX\\_PREPROCESS\\_SET\\_PROB\\_START](#) = 14
- const int [SPEEX\\_PREPROCESS\\_GET\\_PROB\\_START](#) = 15
- const int [SPEEX\\_PREPROCESS\\_SET\\_PROB\\_CONTINUE](#) = 16
- const int [SPEEX\\_PREPROCESS\\_GET\\_PROB\\_CONTINUE](#) = 17
- const int [SPEEX\\_PREPROCESS\\_SET\\_NOISE\\_SUPPRESS](#) = 18
- const int [SPEEX\\_PREPROCESS\\_GET\\_NOISE\\_SUPPRESS](#) = 19
- const int [SPEEX\\_PREPROCESS\\_SET\\_ECHO\\_SUPPRESS](#) = 20
- const int [SPEEX\\_PREPROCESS\\_GET\\_ECHO\\_SUPPRESS](#) = 21
- const int [SPEEX\\_PREPROCESS\\_SET\\_ECHO\\_SUPPRESS\\_ACTIVE](#) = 22
- const int [SPEEX\\_PREPROCESS\\_GET\\_ECHO\\_SUPPRESS\\_ACTIVE](#) = 23
- const int [SPEEX\\_PREPROCESS\\_SET\\_ECHO\\_STATE](#) = 24
- const int [SPEEX\\_PREPROCESS\\_GET\\_ECHO\\_STATE](#) = 25
- const int [SPEEX\\_PREPROCESS\\_SET\\_AGC\\_INCREMENT](#) = 26
- const int [SPEEX\\_PREPROCESS\\_GET\\_AGC\\_INCREMENT](#) = 27
- const int [SPEEX\\_PREPROCESS\\_SET\\_AGC\\_DECREMENT](#) = 28
- const int [SPEEX\\_PREPROCESS\\_GET\\_AGC\\_DECREMENT](#) = 29
- const int [SPEEX\\_PREPROCESS\\_SET\\_AGC\\_MAX\\_GAIN](#) = 30
- const int [SPEEX\\_PREPROCESS\\_GET\\_AGC\\_MAX\\_GAIN](#) = 31
- const int [SPEEX\\_PREPROCESS\\_GET\\_AGC\\_LOUDNESS](#) = 33
- const int [SPEEX\\_PREPROCESS\\_GET\\_AGC\\_GAIN](#) = 35
- const int [SPEEX\\_PREPROCESS\\_GET\\_PSD\\_SIZE](#) = 37
- const int [SPEEX\\_PREPROCESS\\_GET\\_PSD](#) = 39
- const int [SPEEX\\_PREPROCESS\\_GET\\_NOISE\\_PSD\\_SIZE](#) = 41
- const int [SPEEX\\_PREPROCESS\\_GET\\_NOISE\\_PSD](#) = 43
- const int [SPEEX\\_PREPROCESS\\_GET\\_PROB](#) = 45
- const int [SPEEX\\_PREPROCESS\\_SET\\_AGC\\_TARGET](#) = 46
- const int [SPEEX\\_PREPROCESS\\_GET\\_AGC\\_TARGET](#) = 47
- const int [SPEEX\\_ECHO\\_GET\\_FRAME\\_SIZE](#) = 3
- const int [SPEEX\\_ECHO\\_SET\\_SAMPLING\\_RATE](#) = 24
- const int [SPEEX\\_ECHO\\_GET\\_SAMPLING\\_RATE](#) = 25
- const int [SPEEX\\_ECHO\\_GET\\_IMPULSE\\_RESPONSE\\_SIZE](#) = 27
- const int [SPEEX\\_ECHO\\_GET\\_IMPULSE\\_RESPONSE](#) = 29

### 3.83.1 Member Data Documentation

#### 3.83.1.1 const int [SPEEX\\_ECHO\\_GET\\_FRAME\\_SIZE](#) = 3

Obtain frame size used by the AEC

3.83.1.2 `const int SPEEX_ECHO_GET_IMPULSE_RESPONSE = 29`

Get impulse response (int32[])

3.83.1.3 `const int SPEEX_ECHO_GET_IMPULSE_RESPONSE_SIZE = 27`

Get size of impulse response (int32)

3.83.1.4 `const int SPEEX_ECHO_GET_SAMPLING_RATE = 25`

Get sampling rate

3.83.1.5 `const int SPEEX_ECHO_SET_SAMPLING_RATE = 24`

Set sampling rate

3.83.1.6 `const int SPEEX_PREPROCESS_GET_AGC = 3`

Get preprocessor Automatic Gain Control state

3.83.1.7 `const int SPEEX_PREPROCESS_GET_AGC_DECREMENT = 29`

Get maximal gain decrease in dB/second (int32)

3.83.1.8 `const int SPEEX_PREPROCESS_GET_AGC_GAIN = 35`

Get current gain (int32 percent)

3.83.1.9 `const int SPEEX_PREPROCESS_GET_AGC_INCREMENT = 27`

Get maximal gain increase in dB/second (int32)

3.83.1.10 `const int SPEEX_PREPROCESS_GET_AGC_LEVEL = 7`

Get preprocessor Automatic Gain Control level (float)

3.83.1.11 `const int SPEEX_PREPROCESS_GET_AGC_LOUDNESS = 33`

Get loudness

3.83.1.12 `const int SPEEX_PREPROCESS_GET_AGC_MAX_GAIN = 31`

Get maximal gain in dB (int32)

3.83.1.13 `const int SPEEX_PREPROCESS_GET_AGC_TARGET = 47`

Get preprocessor Automatic Gain Control level (int32)

3.83.1.14 `const int SPEEX_PREPROCESS_GET_DENOISE = 1`

Get preprocessor denoiser state

3.83.1.15 `const int SPEEX_PREPROCESS_GET_DEREVERB = 9`

Get preprocessor dereverb state

3.83.1.16 `const int SPEEX_PREPROCESS_GET_DEREVERB_DECAY = 13`

Get preprocessor dereverb decay

3.83.1.17 `const int SPEEX_PREPROCESS_GET_DEREVERB_LEVEL = 11`

Get preprocessor dereverb level

3.83.1.18 `const int SPEEX_PREPROCESS_GET_ECHO_STATE = 25`

Get the corresponding echo canceller state

3.83.1.19 `const int SPEEX_PREPROCESS_GET_ECHO_SUPPRESS = 21`

Get maximum attenuation of the residual echo in dB (negative number)

3.83.1.20 `const int SPEEX_PREPROCESS_GET_ECHO_SUPPRESS_ACTIVE = 23`

Get maximum attenuation of the residual echo in dB when near end is active (negative number)

3.83.1.21 `const int SPEEX_PREPROCESS_GET_NOISE_PSD = 43`

Get noise estimate (int32[]) of squared values)

3.83.1.22 `const int SPEEX_PREPROCESS_GET_NOISE_PSD_SIZE = 41`

Get spectrum size for noise estimate (int32)

3.83.1.23 `const int SPEEX_PREPROCESS_GET_NOISE_SUPPRESS = 19`

Get maximum attenuation of the noise in dB (negative number)

3.83.1.24 `const int SPEEX_PREPROCESS_GET_PROB = 45`

Get speech probability in last frame (int32).

3.83.1.25 `const int SPEEX_PREPROCESS_GET_PROB_CONTINUE = 17`

Get probability required for the VAD to stay in the voice state (integer percent)

3.83.1.26 `const int SPEEX_PREPROCESS_GET_PROB_START = 15`

Get probability required for the VAD to go from silence to voice

3.83.1.27 `const int SPEEX_PREPROCESS_GET_PSD = 39`

Get power spectrum (int32[] of squared values)

3.83.1.28 `const int SPEEX_PREPROCESS_GET_PSD_SIZE = 37`

Get spectrum size for power spectrum (int32)

3.83.1.29 `const int SPEEX_PREPROCESS_GET_VAD = 5`

Get preprocessor [Voice](#) Activity Detection state

3.83.1.30 `const int SPEEX_PREPROCESS_SET_AGC = 2`

Set preprocessor Automatic Gain Control state

3.83.1.31 `const int SPEEX_PREPROCESS_SET_AGC_DECREMENT = 28`

Set maximal gain decrease in dB/second (int32)

3.83.1.32 `const int SPEEX_PREPROCESS_SET_AGC_INCREMENT = 26`

Set maximal gain increase in dB/second (int32)

3.83.1.33 `const int SPEEX_PREPROCESS_SET_AGC_LEVEL = 6`

Set preprocessor Automatic Gain Control level (float)

3.83.1.34 `const int SPEEX_PREPROCESS_SET_AGC_MAX_GAIN = 30`

Set maximal gain in dB (int32)

3.83.1.35 `const int SPEEX_PREPROCESS_SET_AGC_TARGET = 46`

Set preprocessor Automatic Gain Control level (int32)

3.83.1.36 `const int SPEEX_PREPROCESS_SET_DENOISE = 0`

Set preprocessor denoiser state

3.83.1.37 `const int SPEEX_PREPROCESS_SET_DEREVERB = 8`

Set preprocessor dereverb state

3.83.1.38 `const int SPEEX_PREPROCESS_SET_DEREVERB_DECAY = 12`

Set preprocessor dereverb decay

3.83.1.39 `const int SPEEX_PREPROCESS_SET_DEREVERB_LEVEL = 10`

Set preprocessor dereverb level

3.83.1.40 `const int SPEEX_PREPROCESS_SET_ECHO_STATE = 24`

Set the corresponding echo canceller state so that residual echo suppression can be performed (NULL for no residual echo suppression)

3.83.1.41 `const int SPEEX_PREPROCESS_SET_ECHO_SUPPRESS = 20`

Set maximum attenuation of the residual echo in dB (negative number)

3.83.1.42 `const int SPEEX_PREPROCESS_SET_ECHO_SUPPRESS_ACTIVE = 22`

Set maximum attenuation of the residual echo in dB when near end is active (negative number)

3.83.1.43 `const int SPEEX_PREPROCESS_SET_NOISE_SUPPRESS = 18`

Set maximum attenuation of the noise in dB (negative number)

3.83.1.44 `const int SPEEX_PREPROCESS_SET_PROB_CONTINUE = 16`

Set probability required for the VAD to stay in the voice state (integer percent)

3.83.1.45 `const int SPEEX_PREPROCESS_SET_PROB_START = 14`

Set probability required for the VAD to go from silence to voice

3.83.1.46 `const int SPEEX_PREPROCESS_SET_VAD = 4`

Set preprocessor [Voice](#) Activity Detection state

## 3.84 SpeexProcessor Class Reference

Inherits [SpeexLib](#), and [IProcessor< short >](#).

### Classes

- struct [AECLatencyResultType](#)

## Public Member Functions

- void **ResetAEC** ()
- void **AECLatencyDetectCalibrate** ()
- **SpeexProcessor** ([ILogger](#) logger, Func< long > clockMs, int frameSize, int samplingRate, int channels, int playSamplingRate, int playChannels, int playBufSize)
- void **InitAEC** ()
- short[] **Process** (short[] buf)
- void **OnAudioOutFrame** (float[] data, int outChannels)
- void **PrintInfo** ()
- void **Dispose** ()

## Properties

- bool **AEC** [get, set]
- int **AECFilterLengthMs** [get, set]
- int **AECPlaybackDelayMs** [get, set]
- int **AECCurrentPlayDelayFrames** [get]
- bool **AECLatencyDetect** [get, set]
- [AECLatencyResultType](#) **AECLatencyResult** [get]
- bool **Denoise** [get, set]
- bool **AGC** [get, set]
- float **AGCLevel** [get, set]

## Additional Inherited Members

## 3.85 TestTone Class Reference

Inherits [MonoBehaviour](#).

## 3.86 AudioUtil.ToneAudioPusher< T > Class Template Reference

[IAudioPusher](#) that provides a constant tone signal.

Inherits [IAudioPusher< T >](#).

## Public Member Functions

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

## Properties

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

### 3.86.1 Detailed Description

[IAudioPusher](#) that provides a constant tone signal.

### 3.86.2 Constructor & Destructor Documentation

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

Create a new [ToneAudioReader](#) instance

Parameters

<i>frequency</i>	Frequency of the generated tone (in Hz).
<i>bufSizeMs</i>	Size of buffers to push (in milliseconds).
<i>samplingRate</i>	Sampling rate of the audio signal (in Hz).
<i>channels</i>	Number of channels in the audio signal.

### 3.86.3 Member Function Documentation

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

Set the callback function used for pushing data

Parameters

<i>callback</i>	Callback function to use
<i>localVoice</i>	Outgoing audio stream, for context

Implements [IAudioPusher< T >](#).

## 3.87 ToneAudioReader Class Reference

Inherits [IAudioReader< float >](#).

### Public Member Functions

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

### Properties

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

## 3.88 AudioUtil.ToneAudioReader< T > Class Template Reference

[IAudioReader](#) that provides a constant tone signal.

Inherits [IAudioReader< T >](#).



## Public Member Functions

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

Create a new [ToneAudioReader](#) instance

- void **Dispose** ()
- bool [Read](#) (T[] buf)

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

## Properties

- int [Channels](#) [get]  
Number of channels in the audio signal.
- int [SamplingRate](#) [get]  
Sampling rate of the audio signal (in Hz).
- string [Error](#) [get]  
If not null, audio object is in invalid state.

### 3.88.1 Detailed Description

[IAudioReader](#) that provides a constant tone signal.

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

### 3.88.2 Constructor & Destructor Documentation

- 3.88.2.1** [ToneAudioReader](#) ( Func< double > *clockSec* = null, double *frequency* = 440, int *samplingRate* = 441000, int *channels* = 2 )

Create a new [ToneAudioReader](#) instance

#### Parameters

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

### 3.88.3 Member Function Documentation

- 3.88.3.1** bool [Read](#) ( T[] *buffer* )

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

#### Parameters

<i>buffer</i>	Buffer to fill.
---------------	-----------------

#### Returns

True if buffer was filled successfully, false otherwise.

Implements [IDataReader< T >](#).

### 3.88.4 Property Documentation

#### 3.88.4.1 `int Channels` [get]

Number of channels in the audio signal.

#### 3.88.4.2 `string Error` [get]

If not null, audio object is in invalid state.

#### 3.88.4.3 `int SamplingRate` [get]

Sampling rate of the audio signal (in Hz).

## 3.89 UnityAndroidAudioInAEC Class Reference

Inherits [IAudioPusher< short >](#).

### Public Member Functions

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

### Properties

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

## 3.90 UnityAudioOut Class Reference

Inherits [ISyncAudioOut](#).

### Public Member Functions

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

### Public Attributes

- const int **FRAME\_POOL\_CAPACITY** = 50

## Properties

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

## 3.91 UnsupportedCodecException Class Reference

Exception thrown if an unsupported codec is encountered.

Inherits Exception.

### Public Member Functions

- [UnsupportedCodecException](#) ([Codec](#) codec, [LocalVoice](#) voice)  
*Create a new [UnsupportedCodecException](#).*

#### 3.91.1 Detailed Description

Exception thrown if an unsupported codec is encountered.

PhotonVoice currently only supports one Codec, [Codec.AudioOpus](#).

#### 3.91.2 Constructor & Destructor Documentation

##### 3.91.2.1 UnsupportedCodecException ( [Codec](#) codec, [LocalVoice](#) voice )

Create a new [UnsupportedCodecException](#).

##### Parameters

<i>codec</i>	The codec actually encountered.
<i>voice</i>	The <a href="#">LocalVoice</a> (outgoing stream) involved.

## 3.92 UnsupportedSampleTypeException Class Reference

Exception thrown if an unsupported audio sample type is encountered.

Inherits Exception.

### Public Member Functions

- [UnsupportedSampleTypeException](#) (Type t)  
*Create a new [UnsupportedSampleTypeException](#).*

#### 3.92.1 Detailed Description

Exception thrown if an unsupported audio sample type is encountered.

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

### 3.92.2 Constructor & Destructor Documentation

#### 3.92.2.1 `UnsupportedSampleTypeException` ( Type `t` )

Create a new `UnsupportedSampleTypeException`.

Parameters

<code>t</code>	The sample type actually encountered.
----------------	---------------------------------------

### 3.93 OpusCodec.Util Class Reference

### 3.94 VoiceClient Class Reference

Base class for `Voice` clients implamantations

Inherits `IDisposable`.

#### Public Member Functions

- delegate void `RemoteVoiceInfoDelegate` (int channelId, int playerId, byte voiceld, `VoiceInfo` voiceInfo, ref `RemoteVoiceOptions` options)  
*Remote voice info event delegate.*
- `IEnumerable< LocalVoice > LocalVoicesInChannel` (int channelId)  
*Iterates through copy of all local voices list of given channel.*
- void `Service` ()  
*This method dispatches all available incoming commands and then sends this client's outgoing commands. Call this method regularly (2..20 times a second).*
- `LocalVoice CreateLocalVoice` (`VoiceInfo` voiceInfo, int channelId=ChannelAuto, `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↔Id=ChannelAuto, `IEncoderDataFlow< T >` encoder=null)  
*Creates outgoing stream consuming sequence of values passed in array buffers of arbitrary length which repacked in frames of constant length for further processing and encoding.*
- `LocalVoiceAudio< T > CreateLocalVoiceAudio< T >` (`VoiceInfo` voiceInfo, int channelId=ChannelAuto, `I↔Encoder` encoder=null)  
*Creates outgoing audio stream. Adds audio specific features (e.g. resampling, level meter) to processing pipeline and to returning stream handler.*
- `Voice.LocalVoice CreateLocalVoiceAudioFromSource` (`Voice.VoiceInfo` voiceInfo, `Voice.IAudioDesc` source, bool forceShort=false, int channelId=ChannelAuto, `IEncoder` encoder=null)  
*Creates outgoing audio stream of type automatically assigned and adds procedures (callback or serviceable) for consuming given audio source data. Adds audio specific features (e.g. resampling, level meter) to processing pipeline and to returning stream handler.*
- void `RemoveLocalVoice` (`LocalVoice` voice)  
*Removes local voice (outgoing data stream).*  
Parameters  

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

#### Public Attributes

- const int `ChannelAuto` = -1

## 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.*
- `IEnumerable< object > RemoteVoiceLocalUserObjects` [get]  
*Iterates through all local objects set by user in remote voices.*

### 3.94.1 Detailed Description

Base class for [Voice](#) clients implamantations

### 3.94.2 Member Function Documentation

**3.94.2.1 [LocalVoice](#) CreateLocalVoice ( [VoiceInfo](#) voiceInfo, int channelId = ChannelAuto, [IEncoder](#) encoder = null )**

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

#### Parameters

<i>voiceInfo</i>	Outgoing stream parameters. Set applicable fields to read them by encoder and by receiving client when voice created.
<i>channelId</i>	Transport channel specific to frontend. Set to <a href="#">VoiceClient.ChannelAuto</a> to let frontend automatically assign channel.
<i>encoder</i>	Encoder producing the stream.

#### Returns

Outgoing stream handler.

**3.94.2.2 LocalVoiceAudio<T> CreateLocalVoiceAudio< T > ( VoiceInfo voiceInfo, int channelId = ChannelAuto, IEncoder encoder = null )**

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

## Template Parameters

<i>T</i>	Element type of audio array buffers.
----------	--------------------------------------

## Parameters

<i>voiceInfo</i>	Outgoing audio stream parameters. Set applicable fields to read them by encoder and by receiving client when voice created.
<i>channelId</i>	Transport channel specific to frontend. Set to VoiceClient.ChannelAuto to let frontend automatically assign channel.
<i>encoder</i>	Audio encoder. Set to null to use default Opus encoder.

## Returns

Outgoing stream handler.

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

### 3.94.2.3 Voice.LocalVoice CreateLocalVoiceAudioFromSource ( Voice.VoiceInfo *voiceInfo*, Voice.IAudioDesc *source*, bool *forceShort* = false, int *channelId* = ChannelAuto, IEncoder *encoder* = null )

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

## Parameters

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

## Returns

Outgoing stream handler.

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

### 3.94.2.4 LocalVoiceFramed<T> CreateLocalVoiceFramed< T > ( VoiceInfo *voiceInfo*, int *frameSize*, int *channelId* = ChannelAuto, IEncoderDataFlow< T > *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

<i>T</i>	Type of data consumed by outgoing stream (element type of array buffers).
----------	---

## Parameters

<i>voiceInfo</i>	Outgoing stream parameters. Set applicable fields to read them by encoder and by receiving client when voice created.
<i>channelId</i>	Transport channel specific to frontend. Set to VoiceClient.ChannelAuto to let frontend automatically assign channel.
<i>encoder</i>	Encoder compressing data stream in pipeline.

## Returns

Outgoing stream handler.

### 3.94.2.5 IEnumerable<LocalVoice> LocalVoicesInChannel ( int channelId )

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

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

Remote voice info event delegate.

### 3.94.2.7 void RemoveLocalVoice ( LocalVoice voice )

Removes local voice (outgoing data stream).

## Parameters

<i>voice</i>	Handler of outgoing stream to be removed.
--------------	---

### 3.94.2.8 void Service ( )

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

## 3.94.3 Property Documentation

### 3.94.3.1 int DebugLostPercent [get], [set]

Lost frames simulation ratio.

### 3.94.3.2 int FramesLost [get], [set]

Lost frames counter.

### 3.94.3.3 int FramesReceived [get]

Received frames counter.

### 3.94.3.4 int FramesSent [get]

Sent frames counter.



**3.94.3.5 int FramesSentBytes** [get]

Sent frames bytes counter.

**3.94.3.6 IEnumerable<LocalVoice> LocalVoices** [get]

Iterates through copy of all local voices list.

**3.94.3.7 RemoteVoiceInfoDelegate OnRemoteVoiceInfoAction** [get], [set]

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

**3.94.3.8 IEnumerable<RemoteVoiceInfo> RemoteVoiceInfos** [get]

Iterates through all remote voices infos.

**3.94.3.9 IEnumerable<object> RemoteVoiceLocalUserObjects** [get]

Iterates through all local objects set by user in remote voices.

**3.94.3.10 int RoundTripTime** [get]

Average time required voice packet to return to sender.

**3.94.3.11 int RoundTripTimeVariance** [get]

Average round trip time variation.

**3.94.3.12 bool SuppressInfoDuplicateWarning** [get], [set]

Do not log warning when duplicate info received.

## 3.95 VoiceComponent Class Reference

Inherits [MonoBehaviour](#), and [ILoggable](#).

Inherited by [PhotonVoiceView](#), [Recorder](#), [Speaker](#), and [WebRtcAudioDsp](#).

### Protected Member Functions

- virtual void **Awake** ()

### Protected Attributes

- DebugLevel **logLevel** = DebugLevel.ERROR

## Properties

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

## 3.96 VoiceConnection Class Reference

Component that represents a client voice connection to [Photon](#) Servers.

Inherits [ConnectionHandler](#), and [ILoggable](#).

Inherited by [PhotonVoiceNetwork](#).

## Public Member Functions

- bool [ConnectUsingSettings](#) (AppSettings overwriteSettings=null)  
*Connect to [Photon](#) server using [Settings](#)*

## Public Attributes

- AppSettings [Settings](#)  
*Settings to be used by this voice connection*
- [Recorder](#) [PrimaryRecorder](#)  
*Main [Recorder](#) to be used for transmission by default*
- Func< int, byte, object, [Speaker](#) > [SpeakerFactory](#)  
*Special factory to link [Speaker](#) components with incoming remote audio streams*

## Protected Member Functions

- override void **Awake** ()
- virtual void **Update** ()
- override void **OnDestroy** ()
- override void **OnApplicationQuit** ()
- void **CalcStatistics** ()

## Properties

- [VoiceLogger](#) [Logger](#) [get, protected set]  
*[Logger](#) used by this component*
- DebugLevel [LogLevel](#) [get, set]  
*Log level for this component*
- new [LoadBalancingFrontend](#) [Client](#) [get]  
*Returns underlying [Photon](#) [LoadBalancing](#) client.*
- [VoiceClient](#) [VoiceClient](#) [get]  
*Returns underlying [Photon](#) [Voice](#) client.*
- ClientState [ClientState](#) [get]  
*Returns [Photon](#) [Voice](#) client state.*
- float [FramesReceivedPerSecond](#) [get]  
*Number of frames received per second.*
- float [FramesLostPerSecond](#) [get]  
*Number of frames lost per second.*

- float [FramesLostPercent](#) [get]  
*Percentage of lost frames.*
- GameObject [SpeakerPrefab](#) [get, set]  
*Prefab that contains [Speaker](#) component to be instantiated when receiving a new remote audio source info*

## Events

- Action< [Speaker](#) > [SpeakerLinked](#)  
*Fires when a speaker has been linked to a remote audio stream*

### 3.96.1 Detailed Description

Component that represents a client voice connection to [Photon](#) Servers.

### 3.96.2 Member Function Documentation

#### 3.96.2.1 bool ConnectUsingSettings ( AppSettings *overwriteSettings* = null )

Connect to [Photon](#) server using [Settings](#)

Parameters

<i>overwriteSettings</i>	Overwrites <a href="#">Settings</a> before connecting
--------------------------	---

Returns

If true voice connection command was sent from client

### 3.96.3 Member Data Documentation

#### 3.96.3.1 Recorder PrimaryRecorder

Main [Recorder](#) to be used for transmission by default

#### 3.96.3.2 AppSettings Settings

Settings to be used by this voice connection

#### 3.96.3.3 Func<int, byte, object, Speaker> SpeakerFactory

Special factory to link [Speaker](#) components with incoming remote audio streams

### 3.96.4 Property Documentation

#### 3.96.4.1 new LoadBalancingFrontend Client [get]

Returns underlying [Photon](#) LoadBalancing client.

#### 3.96.4.2 ClientState ClientState [get]

Returns [Photon Voice](#) client state.

**3.96.4.3 float FramesLostPercent** [get]

Percentage of lost frames.

**3.96.4.4 float FramesLostPerSecond** [get]

Number of frames lost per second.

**3.96.4.5 float FramesReceivedPerSecond** [get]

Number of frames received per second.

**3.96.4.6 VoiceLogger Logger** [get], [protected set]

[Logger](#) used by this component

**3.96.4.7 DebugLevel LogLevel** [get], [set]

Log level for this component

**3.96.4.8 GameObject SpeakerPrefab** [get], [set]

Prefab that contains [Speaker](#) component to be instantiated when receiving a new remote audio source info

**3.96.4.9 VoiceClient VoiceClient** [get]

Returns underlying [Photon Voice](#) client.

**3.96.5 Event Documentation****3.96.5.1 Action<Speaker> SpeakerLinked**

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

**3.97 AudioUtil.VoiceDetector< T > Class Template Reference**

Simple voice activity detector triggered by signal level.

Inherits [IProcessor< T >](#), and [AudioUtil.IVoiceDetector](#).

**Public Member Functions**

- abstract T[] [Process](#) (T[] buf)  
*Process a frame of audio data.*
- void **Dispose** ()

## Protected Attributes

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

## Properties

- bool **On** [get, set]  
*If true, voice detection enabled.*
- float **Threshold** [get, set]  
*Voice detected as soon as signal level exceeds threshold.*
- bool **Detected** [get, protected set]  
*If true, voice detected.*
- DateTime **DetectedTime** [get]  
*Last time when switched to detected state.*
- int **ActivityDelayMs** [get, set]  
*Keep detected state during this time after signal level dropped below threshold.*

## Events

- Action **OnDetected**  
*Called when switched to detected state.*

### 3.97.1 Detailed Description

Simple voice activity detector triggered by signal level.

### 3.97.2 Member Function Documentation

#### 3.97.2.1 abstract T [] Process ( T[] buf ) [pure virtual]

Process a frame of audio data.

##### Parameters

<i>buf</i>	Buffer containing input audio data
------------	------------------------------------

##### Returns

Buffer containing output audio data

Implements [IProcessor< T >](#).

### 3.97.3 Property Documentation

#### 3.97.3.1 int ActivityDelayMs [get], [set]

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

### 3.97.3.2 bool Detected [get], [protected set]

If true, voice detected.

### 3.97.3.3 DateTime DetectedTime [get]

Last time when switched to detected state.

### 3.97.3.4 bool On [get], [set]

If true, voice detection enabled.

### 3.97.3.5 float Threshold [get], [set]

Voice detected as soon as signal level exceeds threshold.

## 3.97.4 Event Documentation

### 3.97.4.1 Action OnDetected

Called when switched to detected state.

## 3.98 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 [VoiceDetectorCalibrate](#) (int durationMs)  
*Start calibration.*
- T[] [Process](#) (T[] buf)  
*Process a frame of audio data.*
- void **Dispose** ()

### Protected Attributes

- int **voiceDetectorCalibrateCount**

### Properties

- bool **VoiceDetectorCalibrating** [get]

### 3.98.1 Detailed Description

Calibration Utility for [Voice](#) Detector

Using this audio processor, you can calibrate the [IVoiceDetector.Threshold](#).

### 3.98.2 Constructor & Destructor Documentation

#### 3.98.2.1 VoiceDetectorCalibration ( [IVoiceDetector](#) *voiceDetector*, [ILevelMeter](#) *levelMeter*, int *samplingRate*, int *channels* )

Create new [VoiceDetectorCalibration](#) instance.

Parameters

<i>voiceDetector</i>	<a href="#">Voice</a> Detector to calibrate.
<i>levelMeter</i>	Level Meter to look at for calibration.
<i>samplingRate</i>	Sampling rate of the audio signal (in Hz).
<i>numChannels</i>	Number of channels in the audio signal.

### 3.98.3 Member Function Documentation

#### 3.98.3.1 T [] Process ( T[] *buf* )

Process a frame of audio data.

Parameters

<i>buf</i>	Buffer containing input audio data
------------	------------------------------------

Returns

Buffer containing output audio data

Implements [IProcessor< T >](#).

#### 3.98.3.2 void VoiceDetectorCalibrate ( int *durationMs* )

Start calibration.

Parameters

<i>durationMs</i>	Duration of the calibration procedure (in milliseconds).
-------------------	--

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

## 3.99 AudioUtil.VoiceDetectorDummy Class Reference

Dummy [VoiceDetector](#) that doesn't actually do anything.

Inherits [AudioUtil.IVoiceDetector](#).

### Properties

- bool **On** [get, set]

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

## Additional Inherited Members

### 3.99.1 Detailed Description

Dummy [VoiceDetector](#) that doesn't actually do anything.

## 3.100 AudioUtil.VoiceDetectorFloat Class Reference

[VoiceDetector](#) specialization for float audio.

Inherits [AudioUtil.VoiceDetector< float >](#).

## Public Member Functions

- [VoiceDetectorFloat](#) (int samplingRate, int numChannels)  
Create a new [VoiceDetectorFloat](#) instance.
- override float[] **Process** (float[] buffer)

## Additional Inherited Members

### 3.100.1 Detailed Description

[VoiceDetector](#) specialization for float audio.

### 3.100.2 Constructor & Destructor Documentation

#### 3.100.2.1 VoiceDetectorFloat ( int samplingRate, int numChannels )

Create a new [VoiceDetectorFloat](#) instance.

Parameters

<i>samplingRate</i>	Sampling rate of the audio signal (in Hz).
<i>numChannels</i>	Number of channels in the audio signal.

## 3.101 AudioUtil.VoiceDetectorShort Class Reference

[VoiceDetector](#) specialization for float audio.

Inherits [AudioUtil.VoiceDetector< short >](#).



## Public Member Functions

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

## Additional Inherited Members

### 3.101.1 Detailed Description

[VoiceDetector](#) specialization for float audio.

### 3.101.2 Constructor & Destructor Documentation

#### 3.101.2.1 VoiceDetectorShort ( int *samplingRate*, int *numChannels* )

Create a new [VoiceDetectorFloat](#) instance

##### Parameters

<i>samplingRate</i>	Sampling rate of the audio signal (in Hz).
<i>numChannels</i>	Number of channels in the audio signal.

## 3.102 VoiceEventCode Class Reference

PhotonVoice communication uses a single type of event, but differentiates transmission Channels by encoding a channelId into [VoiceEventCode](#).

## Static Public Member Functions

- static byte [GetCode](#) (int channelId)  
*Get the event code for the given channel ID.*
- static bool [TryGetChannelID](#) (byte evCode, int maxChannels, out byte channelId)  
*Try to get the channel ID for the given event code.*

## Public Attributes

- const byte [Code0](#) = 201  
*Start of voice event codes range.*

### 3.102.1 Detailed Description

PhotonVoice communication uses a single type of event, but differentiates transmission Channels by encoding a channelId into [VoiceEventCode](#).

Transmission Channels are not for selective forwarding: use AudioGroups for that. Instead, they are to differentiate opus audio from (future) other codecs or media.

For this purpose, a range of event codes of length LoadBalancingPeer.ChannelCount, starting from Code0, is used.

### 3.102.2 Member Function Documentation

3.102.2.1 `static byte GetCode ( int channelID ) [static]`

Get the event code for the given channel ID.

## Parameters

<i>channelID</i>	Channel ID to get event code for.
------------------	-----------------------------------

## Returns

The corresponding event code.

3.102.2.2 `static bool TryGetChannelID ( byte evCode, int maxChannels, out byte channelID ) [static]`

Try to get the channel ID for the given event code.

## Parameters

<i>evCode</i>	Event code to find Channel ID from.
<i>maxChannels</i>	Maximum Channel ID in use.
<i>channelID</i>	(output) Channel ID found.

## Returns

True if a valid channel ID could be recovered from evCode, false otherwise.

## 3.102.3 Member Data Documentation

3.102.3.1 `const byte Code0 = 201`

Start of voice event codes range.

Change if it conflicts with other event codes used in the same [Photon](#) room.

## 3.103 VoicelInfo Struct Reference

Describes stream properties.

## Public Member Functions

- override string **ToString** ()

## Static Public Member Functions

- static [VoicelInfo CreateAudioOpus](#) (POpusCodec.Enums.SamplingRate samplingRate, int sourceSamplingRate, int channels, OpusCodec.FrameDuration frameDurationUs, int bitrate, object userdata=null)  
*Create stream info for an Opus audio stream.*

## Properties

- [Codec](#) **Codec** [get, set]
- int [SamplingRate](#) [get, set]  
*Audio sampling rate (frequency, in Hz).*
- int [SourceSamplingRate](#) [get, set]  
*Source audio sampling rate (to be resampled to SamplingRate; 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).*
- 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) size in samples.*
- int [Width](#) [get, set]  
*Video width (optional).*
- int [Height](#) [get, set]  
*Video height (optional)*

### 3.103.1 Detailed Description

Describes stream properties.

### 3.103.2 Member Function Documentation

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

Create stream info for an Opus audio stream.

Parameters

<i>samplingRate</i>	Audio sampling rate.
<i>source↔ SamplingRate</i>	Source audio sampling rate (to be resampled to <i>samplingRate</i> ; in Hz).
<i>channels</i>	Number of channels.
<i>frameDurationUs</i>	Uncompressed frame (audio packet) size in microseconds.
<i>bitrate</i>	Stream bitrate (in bits/second).
<i>userdata</i>	Optional user data. Should be serializable by <a href="#">Photon</a> .

Returns

[VoiceInfo](#) instance.

### 3.103.3 Property Documentation

- 3.103.3.1 int [Bitrate](#) [get], [set]

Target bitrate (in bits/second).

- 3.103.3.2 int [Channels](#) [get], [set]

Number of channels.

3.103.3.3 int FrameDurationSamples [get]

Uncompressed frame (data packet) size in samples.

3.103.3.4 int FrameDurationUs [get], [set]

Uncompressed frame (audio packet) size in microseconds.

3.103.3.5 int FrameSize [get]

Uncompressed frame (data packet) size in samples.

3.103.3.6 int Height [get], [set]

Video height (optional)

3.103.3.7 int SamplingRate [get], [set]

Audio sampling rate (frequency, in Hz).

3.103.3.8 int SourceSamplingRate [get], [set]

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

3.103.3.9 object UserData [get], [set]

Optional user data. Should be serializable by [Photon](#).

3.103.3.10 int Width [get], [set]

Video width (optional).

## 3.104 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)  
*Start calibration*
- T[] [Process](#) (T[] buf)  
*Process a frame of audio data.*
- void **Dispose** ()

## Properties

- [ILevelMeter Level](#) [get]  
The [LevelMeter](#) in use.
- [IVoiceDetector Detector](#) [get]  
The [VoiceDetector](#) in use

### 3.104.1 Detailed Description

Utility Audio Processor [Voice](#) Detection Calibration.

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

### 3.104.2 Constructor & Destructor Documentation

#### 3.104.2.1 VoiceLevelDetectCalibrate ( int *samplingRate*, int *channels* )

Create new [VoiceLevelDetectCalibrate](#) instance

Parameters

<i>samplingRate</i>	Sampling rate of the audio signal (in Hz).
<i>numChannels</i>	Number of channels in the audio signal.

### 3.104.3 Member Function Documentation

#### 3.104.3.1 void Calibrate ( int *durationMs* )

Start calibration

Parameters

<i>durationMs</i>	Duration of the calibration procedure (in milliseconds).
-------------------	--

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

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

Process a frame of audio data.

Parameters

<i>buf</i>	Buffer containing input audio data
------------	------------------------------------

Returns

Buffer containing output audio data

Implements [IProcessor< T >](#).

### 3.104.4 Property Documentation

#### 3.104.4.1 IVoiceDetector Detector [get]

The [VoiceDetector](#) in use

## 3.104.4.2 ILevelMeter Level [get]

The [LevelMeter](#) in use.

## 3.105 VoiceLogger Class Reference

Inherits [ILogger](#).

### Public Member Functions

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

### Properties

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

## 3.106 WebRtcAudioDsp Class Reference

Inherits [VoiceComponent](#).

### Protected Member Functions

- override void **Awake** ()

### Properties

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

### Additional Inherited Members

## 3.107 WebRTCAudioLib Class Reference

Inherited by [WebRTCAudioProcessor](#).

## Classes

- struct [ConfigParam](#)
- struct [Param](#)

## Public Member Functions

- static IntPtr **webrtc\_audio\_processor\_create** (int samplingRate, int channels, int frameSize, int rev← SamplingRate, int revChannels)
- static int **webrtc\_audio\_processor\_set\_config\_param** (IntPtr proc, int param, int v)
- static int **webrtc\_audio\_processor\_init** (IntPtr proc)
- static int **webrtc\_audio\_processor\_set\_param** (IntPtr proc, int param, int v)
- static int **webrtc\_audio\_processor\_process** (IntPtr proc, short[] buffer, int offset, out bool voiceDetected)
- static int **webrtc\_audio\_processor\_process\_reverse** (IntPtr proc, short[] buffer, int bufferSize)
- static void **webrtc\_audio\_processor\_destroy** (IntPtr proc)

## 3.108 WebRTCAudioProcessor Class Reference

Inherits [WebRTCAudioLib](#), and [IProcessor< short >](#).

## Public Member Functions

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

## Properties

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



# Index

- AccumAvgPeakAmp
    - Photon::Voice::AudioUtil::ILevelMeter, [33](#)
  - AcquireOrCreate
    - Photon::Voice::ObjectPool, [54](#)
  - ActivityDelayMs
    - Photon::Voice::AudioUtil::IVoiceDetector, [38](#)
    - Photon::Voice::AudioUtil::VoiceDetector, [91](#)
  - Actor
    - Photon::Voice::Unity::Speaker, [71](#)
  - AddPostProcessor
    - Photon::Voice::LocalVoiceFramed, [50](#)
  - AddPreProcessor
    - Photon::Voice::LocalVoiceFramed, [51](#)
  - Audio
    - POpusCodec::Enums, [8](#)
  - AudioClip
    - Photon::Voice::Unity::Recorder, [65](#)
  - AudioClipWrapper, [9](#)
  - AudioDesc, [9](#)
  - AudioGroup
    - Photon::Voice::Unity::Recorder, [65](#)
  - AudioInEnumerator, [10](#)
  - AudioOpus
    - Photon::Voice, [6](#)
  - AudioOutCapture, [10](#)
  - AudioStreamPlayer, [10](#)
  - AudioUtil, [11](#)
  - AudioUtil.ILevelMeter, [32](#)
  - AudioUtil.IVoiceDetector, [38](#)
  - AudioUtil.LevelMeter< T >, [39](#)
  - AudioUtil.LevelMeterDummy, [40](#)
  - AudioUtil.LevelMeterFloat, [41](#)
  - AudioUtil.LevelMeterShort, [41](#)
  - AudioUtil.Resampler< T >, [69](#)
  - AudioUtil.ToneAudioPusher< T >, [77](#)
  - AudioUtil.ToneAudioReader< T >, [78](#)
  - AudioUtil.VoiceDetector< T >, [90](#)
  - AudioUtil.VoiceDetectorCalibration< T >, [92](#)
  - AudioUtil.VoiceDetectorDummy, [93](#)
  - AudioUtil.VoiceDetectorFloat, [94](#)
  - AudioUtil.VoiceDetectorShort, [94](#)
  - AudioUtil.VoiceLevelDetectCalibrate< T >, [99](#)
- Auto
    - POpusCodec::Enums, [8](#)
  - AutoConnectAndJoin
    - Photon::Voice::PUN::PhotonVoiceNetwork, [60](#)
  - AutoCreateRecorderIfNotFound
    - Photon::Voice::PUN::PhotonVoiceView, [61](#)
  - AutoCreateSpeakerIfNotFound
    - Photon::Voice::PUN::PhotonVoiceNetwork, [60](#)
  - AutoLeaveAndDisconnect
    - Photon::Voice::PUN::PhotonVoiceNetwork, [60](#)
  - Bandwidth
    - POpusCodec::Enums, [7](#)
  - Bitrate
    - Photon::Voice::Unity::Recorder, [65](#)
    - Photon::Voice::VoiceInfo, [98](#)
  - BufferReaderPushAdapter
    - Photon::Voice::BufferReaderPushAdapter, [14](#)
  - BufferReaderPushAdapter< T >, [13](#)
  - BufferReaderPushAdapterAsyncPool
    - Photon::Voice::BufferReaderPushAdapterAsync↔  
Pool, [14](#)
  - BufferReaderPushAdapterAsyncPool< T >, [14](#)
  - BufferReaderPushAdapterAsyncPoolCopy
    - Photon::Voice::BufferReaderPushAdapterAsync↔  
PoolCopy, [16](#)
  - BufferReaderPushAdapterAsyncPoolCopy< T >, [16](#)
  - BufferReaderPushAdapterAsyncPoolFloatToShort, [17](#)
    - Photon::Voice::BufferReaderPushAdapterAsync↔  
PoolFloatToShort, [17](#)
  - BufferReaderPushAdapterBase
    - Photon::Voice::BufferReaderPushAdapterBase, [18](#)
  - BufferReaderPushAdapterBase< T >, [18](#)
  - Calibrate
    - Photon::Voice::AudioUtil::VoiceLevelDetect↔  
Calibrate, [100](#)
  - ChangeAudioGroups
    - Photon::Voice::LoadBalancingFrontend, [43](#)
  - ChannelId
    - Photon::Voice::RemoteVoiceInfo, [68](#)
  - Channels
    - POpusCodec::Enums, [7](#)
    - Photon::Voice::AudioUtil::ToneAudioReader, [80](#)
    - Photon::Voice::IAudioDesc, [25](#)
    - Photon::Voice::VoiceInfo, [98](#)
  - ClearProcessors
    - Photon::Voice::LocalVoiceFramed, [51](#)
  - Client
    - Photon::Voice::Unity::VoiceConnection, [89](#)
  - ClientState
    - Photon::Voice::Unity::VoiceConnection, [89](#)
  - Code0
    - Photon::Voice::VoiceEventCode, [97](#)
  - Codec
    - Photon::Voice, [6](#)
  - ConnectAndJoin, [19](#)

- ConnectAndJoinRoom
  - Photon::Voice::PUN::PhotonVoiceNetwork, 59
- ConnectUsingSettings
  - Photon::Voice::Unity::VoiceConnection, 89
- Convert
  - Photon::Voice::AudioUtil, 12
- Count
  - Photon::Voice::Framer, 24
- Create
  - Photon::Voice::LocalVoiceAudio, 47
- CreateAudioOpus
  - Photon::Voice::VoiceInfo, 98
- CreateLocalVoice
  - Photon::Voice::VoiceClient, 83
- CreateLocalVoiceAudio< T >
  - Photon::Voice::VoiceClient, 83
- CreateLocalVoiceAudioFromSource
  - Photon::Voice::VoiceClient, 85
- CreateLocalVoiceFramed< T >
  - Photon::Voice::VoiceClient, 85
- CurrentAvgAmp
  - Photon::Voice::AudioUtil::ILevelMeter, 33
- CurrentPeakAmp
  - Photon::Voice::AudioUtil::ILevelMeter, 33
- DebugEchoMode
  - Photon::Voice::LocalVoice, 46
  - Photon::Voice::Unity::Recorder, 65
- DebugLostPercent
  - Photon::Voice::VoiceClient, 86
- Decode
  - Photon::Voice::IDecoderQueued, 29
- DecodeToByte
  - Photon::Voice::IDecoderDirect, 28
  - Photon::Voice::OpusCodec::Decoder, 20
- DecodeToFloat
  - Photon::Voice::IDecoderDirect, 28
  - Photon::Voice::OpusCodec::Decoder, 20
- DecodeToShort
  - Photon::Voice::IDecoderDirect, 28
  - Photon::Voice::OpusCodec::Decoder, 20
- Decoder
  - Photon::Voice::RemoteVoiceOptions, 69
- Delay
  - POpusCodec::Enums, 8
- Delay10ms
  - POpusCodec::Enums, 8
- Delay20ms
  - POpusCodec::Enums, 8
- Delay2dot5ms
  - POpusCodec::Enums, 8
- Delay40ms
  - POpusCodec::Enums, 8
- Delay5ms
  - POpusCodec::Enums, 8
- Delay60ms
  - POpusCodec::Enums, 8
- Detected
  - Photon::Voice::AudioUtil::IVoiceDetector, 38
- Photon::Voice::AudioUtil::VoiceDetector, 91
- DetectedTime
  - Photon::Voice::AudioUtil::IVoiceDetector, 38
  - Photon::Voice::AudioUtil::VoiceDetector, 92
- Detector
  - Photon::Voice::AudioUtil::VoiceLevelDetect↔  
Calibrate, 100
- Disconnect
  - Photon::Voice::PUN::PhotonVoiceNetwork, 59
- Dispose
  - Photon::Voice::BufferReaderPushAdapterBase, 19
  - Photon::Voice::LoadBalancingFrontend, 43
  - Photon::Voice::LocalVoiceFramed, 51
  - Photon::Voice::ObjectPool, 55
- Dummy
  - Photon::Voice::LocalVoiceAudioDummy, 49
- EncodeAndGetOutput
  - Photon::Voice::IEncoderDataFlowDirect, 31
  - Photon::Voice::OpusCodec::Encoder, 21
- EncoderDelay
  - POpusCodec::OpusEncoder, 57
- Encrypt
  - Photon::Voice::LocalVoice, 46
  - Photon::Voice::Unity::Recorder, 65
- Error
  - Photon::Voice::AudioUtil::ToneAudioReader, 80
  - Photon::Voice::IAudioDesc, 25
  - Photon::Voice::IDecoder, 27
  - Photon::Voice::IEncoder, 30
- FactoryPrimitiveArrayPool< T >, 22
- FactoryReusableArray< T >, 23
- ForceToStereo< T >
  - Photon::Voice::AudioUtil, 12
- Frame
  - Photon::Voice::Framer, 24
- FrameDuration
  - Photon::Voice::Unity::Recorder, 65
- FrameDurationSamples
  - Photon::Voice::VoiceInfo, 98
- FrameDurationUs
  - Photon::Voice::VoiceInfo, 99
- FrameSize
  - Photon::Voice::LocalVoiceFramedBase, 52
  - Photon::Voice::VoiceInfo, 99
- Framer
  - Photon::Voice::Framer, 24
- Framer< T >, 23
- FramesLost
  - Photon::Voice::VoiceClient, 86
- FramesLostPerSecond
  - Photon::Voice::Unity::VoiceConnection, 90
- FramesLostPercent
  - Photon::Voice::Unity::VoiceConnection, 89
- FramesReceived
  - Photon::Voice::VoiceClient, 86
- FramesReceivedPerSecond
  - Photon::Voice::Unity::VoiceConnection, 90

- FramesSent
  - Photon::Voice::LocalVoice, [46](#)
  - Photon::Voice::VoiceClient, [86](#)
- FramesSentBytes
  - Photon::Voice::LocalVoice, [46](#)
  - Photon::Voice::VoiceClient, [86](#)
- Fullband
  - POpusCodec::Enums, [7](#)
- GetCode
  - Photon::Voice::VoiceEventCode, [96](#)
- GetOutput
  - Photon::Voice::IEncoderQueued, [32](#)
- GlobalAudioGroup
  - Photon::Voice::LoadBalancingFrontend, [44](#)
- Group
  - Photon::Voice::LocalVoice, [46](#)
- Height
  - Photon::Voice::VoiceInfo, [99](#)
- IAudioDesc, [24](#)
- IAudioOut, [25](#)
- IAudioPusher< T >, [25](#)
- IAudioReader< T >, [26](#)
- IDataReader< T >, [26](#)
- IDecoder, [27](#)
- IDecoderDirect, [28](#)
- IDecoderQueued, [29](#)
- IDecoderQueuedOutputImageNative, [29](#)
- IEncoder, [30](#)
- IEncoderDataFlow< T >, [30](#)
- IEncoderDataFlowDirect< T >, [30](#)
- IEncoderNativeImageDirect, [31](#)
- IEncoderQueued, [31](#)
- ILocalVoiceAudio, [33](#)
- ILoggable, [34](#)
- ILogger, [34](#)
- IOSAudioForceToSpeaker, [36](#)
- IProcessor< T >, [36](#)
- IServiceable, [37](#)
- ISyncAudioOut, [37](#)
- IVoiceFrontend, [39](#)
- ImageBufferInfo, [34](#)
- ImageBufferNative, [35](#)
- ImageBufferNativeAlloc, [35](#)
- ImageBufferNativeGCHandleSinglePlane, [35](#)
- ImageBufferNativePool< T >, [36](#)
- Info
  - Photon::Voice::LocalVoice, [46](#)
  - Photon::Voice::ObjectPool, [55](#)
  - Photon::Voice::RemoteVoiceInfo, [68](#)
- Init
  - Photon::Voice::ObjectPool, [55](#)
  - Photon::Voice::Unity::Recorder, [64](#)
- InputFactory
  - Photon::Voice::Unity::Recorder, [65](#)
- Instance
  - Photon::Voice::PUN::PhotonVoiceNetwork, [60](#)
- IsCurrentlyTransmitting
  - Photon::Voice::LocalVoice, [46](#)
  - Photon::Voice::Unity::Recorder, [65](#)
- IsInitialized
  - Photon::Voice::Unity::Recorder, [66](#)
- IsPlaying
  - Photon::Voice::Unity::Speaker, [71](#)
- IsRecorder
  - Photon::Voice::PUN::PhotonVoiceView, [61](#)
- IsRecording
  - Photon::Voice::PUN::PhotonVoiceView, [61](#)
- IsSetup
  - Photon::Voice::PUN::PhotonVoiceView, [61](#)
- IsSpeaker
  - Photon::Voice::PUN::PhotonVoiceView, [62](#)
- IsSpeaking
  - Photon::Voice::PUN::PhotonVoiceView, [62](#)
- Lag
  - Photon::Voice::Unity::Speaker, [71](#)
- Level
  - Photon::Voice::AudioUtil::VoiceLevelDetect←  
Calibrate, [100](#)
- LevelMeter
  - Photon::Voice::ILocalVoiceAudio, [34](#)
  - Photon::Voice::Unity::Recorder, [66](#)
- LevelMeterFloat
  - Photon::Voice::AudioUtil::LevelMeterFloat, [41](#)
- LevelMeterShort
  - Photon::Voice::AudioUtil::LevelMeterShort, [42](#)
- LoadBalancingFrontend, [42](#)
  - Photon::Voice::LoadBalancingFrontend, [43](#)
- LocalUserObject
  - Photon::Voice::LocalVoice, [46](#)
  - Photon::Voice::RemoteVoiceInfo, [68](#)
  - Photon::Voice::RemoteVoiceOptions, [69](#)
- LocalUserServiceable
  - Photon::Voice::LocalVoice, [46](#)
- LocalVoice, [44](#)
- LocalVoiceAudio< T >, [47](#)
- LocalVoiceAudioDummy, [48](#)
- LocalVoiceAudioFloat, [49](#)
- LocalVoiceAudioShort, [49](#)
- LocalVoiceFramed< T >, [50](#)
- LocalVoiceFramedBase, [51](#)
- LocalVoices
  - Photon::Voice::VoiceClient, [87](#)
- LocalVoicesInChannel
  - Photon::Voice::VoiceClient, [86](#)
- LogLevel
  - Photon::Voice::Unity::VoiceConnection, [90](#)
- Logger, [52](#)
  - Photon::Voice::Unity::VoiceConnection, [90](#)
- LoopAudioClip
  - Photon::Voice::Unity::Recorder, [66](#)
- Mediumband
  - POpusCodec::Enums, [7](#)
- MicWrapper, [52](#)

- MicrophoneType
  - Photon::Voice::Unity::Recorder, 66
- Mono
  - POpusCodec::Enums, 8
- Music
  - POpusCodec::Enums, 8
- Narrowband
  - POpusCodec::Enums, 7
- ObjectFactory< TType, TInfo >, 52
- ObjectPool
  - Photon::Voice::ObjectPool, 54
- ObjectPool< TType, TInfo >, 53
- On
  - Photon::Voice::AudioUtil::IVoiceDetector, 38
  - Photon::Voice::AudioUtil::VoiceDetector, 92
- OnDecodedFrameByteAction
  - Photon::Voice::RemoteVoiceOptions, 69
- OnDecodedFrameFloatAction
  - Photon::Voice::RemoteVoiceOptions, 69
- OnDecodedFrameShortAction
  - Photon::Voice::RemoteVoiceOptions, 69
- OnDetected
  - Photon::Voice::AudioUtil::IVoiceDetector, 39
  - Photon::Voice::AudioUtil::VoiceDetector, 92
- OnRemoteVoiceInfoAction
  - Photon::Voice::VoiceClient, 87
- OnRemoteVoiceRemoveAction
  - Photon::Voice::RemoteVoiceOptions, 69
  - Photon::Voice::Unity::Speaker, 71
- Open
  - Photon::Voice::IDecoder, 27
  - Photon::Voice::OpusCodec::Decoder, 21
- OpusApplicationType
  - POpusCodec::Enums, 8
- OpusCodec, 55
- OpusCodec.Decoder, 20
- OpusCodec.Encoder< T >, 21
- OpusCodec.EncoderFactory, 22
- OpusCodec.EncoderFloat, 22
- OpusCodec.EncoderShort, 22
- OpusCodec.Util, 82
- OpusDecoder, 56
- OpusEncoder, 56
- OpusException, 57
- POpusCodec, 7
- POpusCodec.Enums, 7
- POpusCodec::Enums
  - Audio, 8
  - Auto, 8
  - Bandwidth, 7
  - Channels, 7
  - Delay, 8
  - Delay10ms, 8
  - Delay20ms, 8
  - Delay2dot5ms, 8
  - Delay40ms, 8
  - Delay5ms, 8
  - Delay60ms, 8
  - Fullband, 7
  - Mediumband, 7
  - Mono, 8
  - Music, 8
  - Narrowband, 7
  - OpusApplicationType, 8
  - RestrictedLowDelay, 8
  - SignalHint, 8
  - Stereo, 8
  - SuperWideband, 7
  - Voice, 8
  - Voip, 8
  - Wideband, 7
- POpusCodec::OpusEncoder
  - EncoderDelay, 57
- Peer
  - Photon::Voice::Unity::UtilityScripts::PhotonVoice↔LagSimulationGui, 58
- Photon, 3
- Photon.Voice, 3
- Photon.Voice.PUN, 6
- Photon.Voice.Unity, 6
- Photon.Voice.Unity.UtilityScripts, 7
- Photon::Voice
  - AudioOpus, 6
  - Codec, 6
- Photon::Voice::AudioUtil
  - Convert, 12
  - ForceToStereo< T >, 12
  - Resample< T >, 12
  - ResampleAndConvert, 13
- Photon::Voice::AudioUtil::ILevelMeter
  - AccumAvgPeakAmp, 33
  - CurrentAvgAmp, 33
  - CurrentPeakAmp, 33
  - ResetAccumAvgPeakAmp, 32
- Photon::Voice::AudioUtil::IVoiceDetector
  - ActivityDelayMs, 38
  - Detected, 38
  - DetectedTime, 38
  - On, 38
  - OnDetected, 39
  - Threshold, 38
- Photon::Voice::AudioUtil::LevelMeter
  - Process, 40
  - ResetAccumAvgPeakAmp, 40
- Photon::Voice::AudioUtil::LevelMeterDummy
  - ResetAccumAvgPeakAmp, 41
- Photon::Voice::AudioUtil::LevelMeterFloat
  - LevelMeterFloat, 41
- Photon::Voice::AudioUtil::LevelMeterShort
  - LevelMeterShort, 42
- Photon::Voice::AudioUtil::Resampler
  - Process, 70
  - Resampler, 70
- Photon::Voice::AudioUtil::ToneAudioPusher

- SetCallback, 78
- ToneAudioPusher, 78
- Photon::Voice::AudioUtil::ToneAudioReader
  - Channels, 80
  - Error, 80
  - Read, 79
  - SamplingRate, 80
  - ToneAudioReader, 79
- Photon::Voice::AudioUtil::VoiceDetector
  - ActivityDelayMs, 91
  - Detected, 91
  - DetectedTime, 92
  - On, 92
  - OnDetected, 92
  - Process, 91
  - Threshold, 92
- Photon::Voice::AudioUtil::VoiceDetectorCalibration
  - Process, 93
  - VoiceDetectorCalibrate, 93
  - VoiceDetectorCalibration, 93
- Photon::Voice::AudioUtil::VoiceDetectorFloat
  - VoiceDetectorFloat, 94
- Photon::Voice::AudioUtil::VoiceDetectorShort
  - VoiceDetectorShort, 95
- Photon::Voice::AudioUtil::VoiceLevelDetectCalibrate
  - Calibrate, 100
  - Detector, 100
  - Level, 100
  - Process, 100
  - VoiceLevelDetectCalibrate, 100
- Photon::Voice::BufferReaderPushAdapter
  - BufferReaderPushAdapter, 14
  - Service, 14
- Photon::Voice::BufferReaderPushAdapterAsyncPool
  - BufferReaderPushAdapterAsyncPool, 14
  - Service, 16
- Photon::Voice::BufferReaderPushAdapterAsyncPool↵
  - Copy
  - BufferReaderPushAdapterAsyncPoolCopy, 16
  - Service, 17
- Photon::Voice::BufferReaderPushAdapterAsyncPool↵
  - FloatToShort
  - BufferReaderPushAdapterAsyncPoolFloatToShort, 17
  - Service, 18
- Photon::Voice::BufferReaderPushAdapterBase
  - BufferReaderPushAdapterBase, 18
  - Dispose, 19
  - Service, 19
- Photon::Voice::Framer
  - Count, 24
  - Frame, 24
  - Framer, 24
- Photon::Voice::IAudioDesc
  - Channels, 25
  - Error, 25
  - SamplingRate, 25
- Photon::Voice::IAudioPusher
  - SetCallback, 26
- Photon::Voice::IDataReader
  - Read, 27
- Photon::Voice::IDecoder
  - Error, 27
  - Open, 27
- Photon::Voice::IDecoderDirect
  - DecodeToByte, 28
  - DecodeToFloat, 28
  - DecodeToShort, 28
- Photon::Voice::IDecoderQueued
  - Decode, 29
- Photon::Voice::IEncoder
  - Error, 30
- Photon::Voice::IEncoderDataFlowDirect
  - EncodeAndGetOutput, 31
- Photon::Voice::IEncoderQueued
  - GetOutput, 32
- Photon::Voice::ILocalVoiceAudio
  - LevelMeter, 34
  - VoiceDetector, 34
  - VoiceDetectorCalibrate, 33
  - VoiceDetectorCalibrating, 34
- Photon::Voice::IProcessor
  - Process, 36
- Photon::Voice::IServiceable
  - Service, 37
- Photon::Voice::LoadBalancingFrontend
  - ChangeAudioGroups, 43
  - Dispose, 43
  - GlobalAudioGroup, 44
  - LoadBalancingFrontend, 43
  - SendDebugEchoVoicesInfo, 44
  - Service, 44
  - VoiceClient, 44
- Photon::Voice::LocalVoice
  - DebugEchoMode, 46
  - Encrypt, 46
  - FramesSent, 46
  - FramesSentBytes, 46
  - Group, 46
  - Info, 46
  - IsCurrentlyTransmitting, 46
  - LocalUserObject, 46
  - LocalUserServiceable, 46
  - Reliable, 46
  - RemoveSelf, 45
  - TransmitEnabled, 46
- Photon::Voice::LocalVoiceAudio
  - Create, 47
  - VoiceDetectorCalibrate, 48
  - VoiceDetectorCalibrating, 48
- Photon::Voice::LocalVoiceAudioDummy
  - Dummy, 49
  - VoiceDetectorCalibrate, 49
- Photon::Voice::LocalVoiceFramed
  - AddPostProcessor, 50
  - AddPreProcessor, 51

- ClearProcessors, [51](#)
- Dispose, [51](#)
- PushData, [51](#)
- PushDataAsync, [51](#)
- PushDataAsyncReady, [51](#)
- Photon::Voice::LocalVoiceFramedBase
  - FrameSize, [52](#)
- Photon::Voice::ObjectPool
  - AcquireOrCreate, [54](#)
  - Dispose, [55](#)
  - Info, [55](#)
  - Init, [55](#)
  - ObjectPool, [54](#)
  - Release, [55](#)
- Photon::Voice::OpusCodec::Decoder
  - DecodeToByte, [20](#)
  - DecodeToFloat, [20](#)
  - DecodeToShort, [20](#)
  - Open, [21](#)
- Photon::Voice::OpusCodec::Encoder
  - EncodeAndGetOutput, [21](#)
- Photon::Voice::PUN::PhotonVoiceNetwork
  - AutoConnectAndJoin, [60](#)
  - AutoCreateSpeakerIfNotFound, [60](#)
  - AutoLeaveAndDisconnect, [60](#)
  - ConnectAndJoinRoom, [59](#)
  - Disconnect, [59](#)
  - Instance, [60](#)
  - VoiceRoomNameSuffix, [60](#)
- Photon::Voice::PUN::PhotonVoiceView
  - AutoCreateRecorderIfNotFound, [61](#)
  - IsRecorder, [61](#)
  - IsRecording, [61](#)
  - IsSetup, [61](#)
  - IsSpeaker, [62](#)
  - IsSpeaking, [62](#)
  - RecorderInUse, [62](#)
  - SetupDebugSpeaker, [61](#)
  - SpeakerInUse, [62](#)
  - UsePrimaryRecorder, [61](#)
- Photon::Voice::RemoteVoiceInfo
  - ChannelId, [68](#)
  - Info, [68](#)
  - LocalUserObject, [68](#)
  - PlayerId, [68](#)
  - VoiceId, [68](#)
- Photon::Voice::RemoteVoiceOptions
  - Decoder, [69](#)
  - LocalUserObject, [69](#)
  - OnDecodedFrameByteAction, [69](#)
  - OnDecodedFrameFloatAction, [69](#)
  - OnDecodedFrameShortAction, [69](#)
  - OnRemoteVoiceRemoveAction, [69](#)
- Photon::Voice::SpeexLib
  - SPEEX\_ECHO\_GET\_FRAME\_SIZE, [72](#)
  - SPEEX\_ECHO\_GET\_IMPULSE\_RESPONSE, [72](#)
  - SPEEX\_ECHO\_GET\_IMPULSE\_RESPONSE\_↔  
SIZE, [73](#)
  - SPEEX\_ECHO\_GET\_SAMPLING\_RATE, [73](#)
  - SPEEX\_ECHO\_SET\_SAMPLING\_RATE, [73](#)
  - SPEEX\_PREPROCESS\_GET\_AGC, [73](#)
  - SPEEX\_PREPROCESS\_GET\_AGC\_DECREM↔  
ENT, [73](#)
  - SPEEX\_PREPROCESS\_GET\_AGC\_GAIN, [73](#)
  - SPEEX\_PREPROCESS\_GET\_AGC\_INCREM↔  
NT, [73](#)
  - SPEEX\_PREPROCESS\_GET\_AGC\_LEVEL, [73](#)
  - SPEEX\_PREPROCESS\_GET\_AGC\_LOUDNE↔  
SS, [73](#)
  - SPEEX\_PREPROCESS\_GET\_AGC\_MAX\_GAIN,  
[73](#)
  - SPEEX\_PREPROCESS\_GET\_AGC\_TARGET, [73](#)
  - SPEEX\_PREPROCESS\_GET\_DENOISE, [73](#)
  - SPEEX\_PREPROCESS\_GET\_DEREVERB, [74](#)
  - SPEEX\_PREPROCESS\_GET\_DEREVERB\_DE↔  
CAY, [74](#)
  - SPEEX\_PREPROCESS\_GET\_DEREVERB\_LE↔  
VEL, [74](#)
  - SPEEX\_PREPROCESS\_GET\_ECHO\_STATE, [74](#)
  - SPEEX\_PREPROCESS\_GET\_ECHO\_SUPPR↔  
ESS, [74](#)
  - SPEEX\_PREPROCESS\_GET\_ECHO\_SUPPR↔  
ESS\_ACTIVE, [74](#)
  - SPEEX\_PREPROCESS\_GET\_NOISE\_PSD, [74](#)
  - SPEEX\_PREPROCESS\_GET\_NOISE\_PSD\_SI↔  
ZE, [74](#)
  - SPEEX\_PREPROCESS\_GET\_NOISE\_SUPPR↔  
ESS, [74](#)
  - SPEEX\_PREPROCESS\_GET\_PROB, [74](#)
  - SPEEX\_PREPROCESS\_GET\_PROB\_CONTIN↔  
UE, [74](#)
  - SPEEX\_PREPROCESS\_GET\_PROB\_START, [74](#)
  - SPEEX\_PREPROCESS\_GET\_PSD, [75](#)
  - SPEEX\_PREPROCESS\_GET\_PSD\_SIZE, [75](#)
  - SPEEX\_PREPROCESS\_GET\_VAD, [75](#)
  - SPEEX\_PREPROCESS\_SET\_AGC, [75](#)
  - SPEEX\_PREPROCESS\_SET\_AGC\_DECREM↔  
ENT, [75](#)
  - SPEEX\_PREPROCESS\_SET\_AGC\_INCREM↔  
NT, [75](#)
  - SPEEX\_PREPROCESS\_SET\_AGC\_LEVEL, [75](#)
  - SPEEX\_PREPROCESS\_SET\_AGC\_MAX\_GAIN,  
[75](#)
  - SPEEX\_PREPROCESS\_SET\_AGC\_TARGET, [75](#)
  - SPEEX\_PREPROCESS\_SET\_DENOISE, [75](#)
  - SPEEX\_PREPROCESS\_SET\_DEREVERB, [75](#)
  - SPEEX\_PREPROCESS\_SET\_DEREVERB\_DE↔  
CAY, [75](#)
  - SPEEX\_PREPROCESS\_SET\_DEREVERB\_LE↔  
VEL, [76](#)
  - SPEEX\_PREPROCESS\_SET\_ECHO\_STATE, [76](#)
  - SPEEX\_PREPROCESS\_SET\_ECHO\_SUPPRE↔  
SS, [76](#)
  - SPEEX\_PREPROCESS\_SET\_ECHO\_SUPPRE↔  
SS\_ACTIVE, [76](#)
  - SPEEX\_PREPROCESS\_SET\_NOISE\_SUPPR↔



- ESS, 76
- SPEEX\_PREPROCESS\_SET\_PROB\_CONTINUE, 76
- SPEEX\_PREPROCESS\_SET\_PROB\_START, 76
- SPEEX\_PREPROCESS\_SET\_VAD, 76
- Photon::Voice::Unity::Recorder
  - AudioClip, 65
  - AudioGroup, 65
  - Bitrate, 65
  - DebugEchoMode, 65
  - Encrypt, 65
  - FrameDuration, 65
  - Init, 64
  - InputFactory, 65
  - IsCurrentlyTransmitting, 65
  - IsInitialized, 66
  - LevelMeter, 66
  - LoopAudioClip, 66
  - MicrophoneType, 66
  - PhotonMicrophoneDeviceId, 66
  - PhotonMicrophoneEnumerator, 66
  - ReInit, 65
  - ReliableMode, 66
  - RequiresInit, 66
  - SamplingRate, 66
  - SourceType, 66
  - TransmitEnabled, 66
  - TypeConvert, 66
  - UnityMicrophoneDevice, 67
  - UserData, 67
  - VoiceDetection, 67
  - VoiceDetectionDelayMs, 67
  - VoiceDetectionThreshold, 67
  - VoiceDetector, 67
  - VoiceDetectorCalibrate, 65
  - VoiceDetectorCalibrating, 67
- Photon::Voice::Unity::Speaker
  - Actor, 71
  - IsPlaying, 71
  - Lag, 71
  - OnRemoteVoiceRemoveAction, 71
- Photon::Voice::Unity::UtilityScripts::PhotonVoiceLagSimulationGui
  - Peer, 58
  - Visible, 58
  - WindowId, 58
  - WindowRect, 58
- Photon::Voice::Unity::VoiceConnection
  - Client, 89
  - ClientState, 89
  - ConnectUsingSettings, 89
  - FramesLostPerSecond, 90
  - FramesLostPercent, 89
  - FramesReceivedPerSecond, 90
  - LogLevel, 90
  - Logger, 90
  - PrimaryRecorder, 89
  - Settings, 89
  - SpeakerFactory, 89
  - SpeakerLinked, 90
  - SpeakerPrefab, 90
  - VoiceClient, 90
- Photon::Voice::UnsupportedCodecException
  - UnsupportedCodecException, 81
- Photon::Voice::UnsupportedSampleTypeException
  - UnsupportedSampleTypeException, 82
- Photon::Voice::VoiceClient
  - CreateLocalVoice, 83
  - CreateLocalVoiceAudio< T >, 83
  - CreateLocalVoiceAudioFromSource, 85
  - CreateLocalVoiceFramed< T >, 85
  - DebugLostPercent, 86
  - FramesLost, 86
  - FramesReceived, 86
  - FramesSent, 86
  - FramesSentBytes, 86
  - LocalVoices, 87
  - LocalVoicesInChannel, 86
  - OnRemoteVoiceInfoAction, 87
  - RemoteVoiceInfoDelegate, 86
  - RemoteVoiceInfos, 87
  - RemoteVoiceLocalUserObjects, 87
  - RemoveLocalVoice, 86
  - RoundTripTime, 87
  - RoundTripTimeVariance, 87
  - Service, 86
  - SuppressInfoDuplicateWarning, 87
- Photon::Voice::VoiceEventCode
  - Code0, 97
  - GetCode, 96
  - TryGetChannelId, 97
- Photon::Voice::VoiceInfo
  - Bitrate, 98
  - Channels, 98
  - CreateAudioOpus, 98
  - FrameDurationSamples, 98
  - FrameDurationUs, 99
  - FrameSize, 99
  - Height, 99
  - SamplingRate, 99
  - SourceSamplingRate, 99
  - UserData, 99
  - Width, 99
- PhotonMicrophoneDeviceId
  - Photon::Voice::Unity::Recorder, 66
- PhotonMicrophoneEnumerator
  - Photon::Voice::Unity::Recorder, 66
- PhotonVoiceLagSimulationGui, 58
- PhotonVoiceNetwork, 58
- PhotonVoiceView, 60
- PlayerId
  - Photon::Voice::RemoteVoiceInfo, 68
- PrimaryRecorder
  - Photon::Voice::Unity::VoiceConnection, 89
- PrimitiveArrayPool< T >, 62
- Process

- Photon::Voice::AudioUtil::LevelMeter, [40](#)
- Photon::Voice::AudioUtil::Resampler, [70](#)
- Photon::Voice::AudioUtil::VoiceDetector, [91](#)
- Photon::Voice::AudioUtil::VoiceDetectorCalibration, [93](#)
- Photon::Voice::AudioUtil::VoiceLevelDetect↔  
Calibrate, [100](#)
- Photon::Voice::IProcessor, [36](#)
- PushData
  - Photon::Voice::LocalVoiceFramed, [51](#)
- PushDataAsync
  - Photon::Voice::LocalVoiceFramed, [51](#)
- PushDataAsyncReady
  - Photon::Voice::LocalVoiceFramed, [51](#)
- Relnit
  - Photon::Voice::Unity::Recorder, [65](#)
- Read
  - Photon::Voice::AudioUtil::ToneAudioReader, [79](#)
  - Photon::Voice::IDataReader, [27](#)
- Recorder, [63](#)
- Recorder.PhotonVoiceCreatedParams, [57](#)
- RecorderInUse
  - Photon::Voice::PUN::PhotonVoiceView, [62](#)
- Release
  - Photon::Voice::ObjectPool, [55](#)
- Reliable
  - Photon::Voice::LocalVoice, [46](#)
- ReliableMode
  - Photon::Voice::Unity::Recorder, [66](#)
- RemoteVoiceInfo, [67](#)
- RemoteVoiceInfoDelegate
  - Photon::Voice::VoiceClient, [86](#)
- RemoteVoiceInfos
  - Photon::Voice::VoiceClient, [87](#)
- RemoteVoiceLocalUserObjects
  - Photon::Voice::VoiceClient, [87](#)
- RemoteVoiceOptions, [68](#)
- RemoveLocalVoice
  - Photon::Voice::VoiceClient, [86](#)
- RemoveSelf
  - Photon::Voice::LocalVoice, [45](#)
- RequiresInit
  - Photon::Voice::Unity::Recorder, [66](#)
- Resample< T >
  - Photon::Voice::AudioUtil, [12](#)
- ResampleAndConvert
  - Photon::Voice::AudioUtil, [13](#)
- Resampler
  - Photon::Voice::AudioUtil::Resampler, [70](#)
- ResetAccumAvgPeakAmp
  - Photon::Voice::AudioUtil::ILevelMeter, [32](#)
  - Photon::Voice::AudioUtil::LevelMeter, [40](#)
  - Photon::Voice::AudioUtil::LevelMeterDummy, [41](#)
- RestrictedLowDelay
  - POpusCodec::Enums, [8](#)
- RoundTripTime
  - Photon::Voice::VoiceClient, [87](#)
- RoundTripTimeVariance
  - Photon::Voice::VoiceClient, [87](#)
- SPEEX\_ECHO\_GET\_FRAME\_SIZE
  - Photon::Voice::SpeexLib, [72](#)
- SPEEX\_ECHO\_GET\_IMPULSE\_RESPONSE
  - Photon::Voice::SpeexLib, [72](#)
- SPEEX\_ECHO\_GET\_IMPULSE\_RESPONSE\_SIZE
  - Photon::Voice::SpeexLib, [73](#)
- SPEEX\_ECHO\_GET\_SAMPLING\_RATE
  - Photon::Voice::SpeexLib, [73](#)
- SPEEX\_ECHO\_SET\_SAMPLING\_RATE
  - Photon::Voice::SpeexLib, [73](#)
- SPEEX\_PREPROCESS\_GET\_AGC
  - Photon::Voice::SpeexLib, [73](#)
- SPEEX\_PREPROCESS\_GET\_AGC\_DECREMENT
  - Photon::Voice::SpeexLib, [73](#)
- SPEEX\_PREPROCESS\_GET\_AGC\_GAIN
  - Photon::Voice::SpeexLib, [73](#)
- SPEEX\_PREPROCESS\_GET\_AGC\_INCREMENT
  - Photon::Voice::SpeexLib, [73](#)
- SPEEX\_PREPROCESS\_GET\_AGC\_LEVEL
  - Photon::Voice::SpeexLib, [73](#)
- SPEEX\_PREPROCESS\_GET\_AGC\_LOUDNESS
  - Photon::Voice::SpeexLib, [73](#)
- SPEEX\_PREPROCESS\_GET\_AGC\_MAX\_GAIN
  - Photon::Voice::SpeexLib, [73](#)
- SPEEX\_PREPROCESS\_GET\_AGC\_TARGET
  - Photon::Voice::SpeexLib, [73](#)
- SPEEX\_PREPROCESS\_GET\_DENOISE
  - Photon::Voice::SpeexLib, [73](#)
- SPEEX\_PREPROCESS\_GET\_DEREVERB
  - Photon::Voice::SpeexLib, [74](#)
- SPEEX\_PREPROCESS\_GET\_DEREVERB\_DECAY
  - Photon::Voice::SpeexLib, [74](#)
- SPEEX\_PREPROCESS\_GET\_DEREVERB\_LEVEL
  - Photon::Voice::SpeexLib, [74](#)
- SPEEX\_PREPROCESS\_GET\_ECHO\_STATE
  - Photon::Voice::SpeexLib, [74](#)
- SPEEX\_PREPROCESS\_GET\_ECHO\_SUPPRESS
  - Photon::Voice::SpeexLib, [74](#)
- SPEEX\_PREPROCESS\_GET\_ECHO\_SUPPRESS↔  
ACTIVE
  - Photon::Voice::SpeexLib, [74](#)
- SPEEX\_PREPROCESS\_GET\_NOISE\_PSD
  - Photon::Voice::SpeexLib, [74](#)
- SPEEX\_PREPROCESS\_GET\_NOISE\_PSD\_SIZE
  - Photon::Voice::SpeexLib, [74](#)
- SPEEX\_PREPROCESS\_GET\_NOISE\_SUPPRESS
  - Photon::Voice::SpeexLib, [74](#)
- SPEEX\_PREPROCESS\_GET\_PROB
  - Photon::Voice::SpeexLib, [74](#)
- SPEEX\_PREPROCESS\_GET\_PROB\_CONTINUE
  - Photon::Voice::SpeexLib, [74](#)
- SPEEX\_PREPROCESS\_GET\_PROB\_START
  - Photon::Voice::SpeexLib, [74](#)
- SPEEX\_PREPROCESS\_GET\_PSD
  - Photon::Voice::SpeexLib, [75](#)
- SPEEX\_PREPROCESS\_GET\_PSD\_SIZE
  - Photon::Voice::SpeexLib, [75](#)



- SPEEX\_PREPROCESS\_GET\_VAD
  - Photon::Voice::SpeexLib, [75](#)
- SPEEX\_PREPROCESS\_SET\_AGC
  - Photon::Voice::SpeexLib, [75](#)
- SPEEX\_PREPROCESS\_SET\_AGC\_DECREMENT
  - Photon::Voice::SpeexLib, [75](#)
- SPEEX\_PREPROCESS\_SET\_AGC\_INCREMENT
  - Photon::Voice::SpeexLib, [75](#)
- SPEEX\_PREPROCESS\_SET\_AGC\_LEVEL
  - Photon::Voice::SpeexLib, [75](#)
- SPEEX\_PREPROCESS\_SET\_AGC\_MAX\_GAIN
  - Photon::Voice::SpeexLib, [75](#)
- SPEEX\_PREPROCESS\_SET\_AGC\_TARGET
  - Photon::Voice::SpeexLib, [75](#)
- SPEEX\_PREPROCESS\_SET\_DENOISE
  - Photon::Voice::SpeexLib, [75](#)
- SPEEX\_PREPROCESS\_SET\_DEREVERB
  - Photon::Voice::SpeexLib, [75](#)
- SPEEX\_PREPROCESS\_SET\_DEREVERB\_DECAY
  - Photon::Voice::SpeexLib, [75](#)
- SPEEX\_PREPROCESS\_SET\_DEREVERB\_LEVEL
  - Photon::Voice::SpeexLib, [76](#)
- SPEEX\_PREPROCESS\_SET\_ECHO\_STATE
  - Photon::Voice::SpeexLib, [76](#)
- SPEEX\_PREPROCESS\_SET\_ECHO\_SUPPRESS
  - Photon::Voice::SpeexLib, [76](#)
- SPEEX\_PREPROCESS\_SET\_ECHO\_SUPPRESS\_↔
  - ACTIVE
    - Photon::Voice::SpeexLib, [76](#)
- SPEEX\_PREPROCESS\_SET\_NOISE\_SUPPRESS
  - Photon::Voice::SpeexLib, [76](#)
- SPEEX\_PREPROCESS\_SET\_PROB\_CONTINUE
  - Photon::Voice::SpeexLib, [76](#)
- SPEEX\_PREPROCESS\_SET\_PROB\_START
  - Photon::Voice::SpeexLib, [76](#)
- SPEEX\_PREPROCESS\_SET\_VAD
  - Photon::Voice::SpeexLib, [76](#)
- SamplingRate
  - Photon::Voice::AudioUtil::ToneAudioReader, [80](#)
  - Photon::Voice::IAudioDesc, [25](#)
  - Photon::Voice::Unity::Recorder, [66](#)
  - Photon::Voice::VoiceInfo, [99](#)
- SendDebugEchoVoicesInfo
  - Photon::Voice::LoadBalancingFrontend, [44](#)
- Service
  - Photon::Voice::BufferReaderPushAdapter, [14](#)
  - Photon::Voice::BufferReaderPushAdapterAsync↔
    - Pool, [16](#)
  - Photon::Voice::BufferReaderPushAdapterAsync↔
    - PoolCopy, [17](#)
  - Photon::Voice::BufferReaderPushAdapterAsync↔
    - PoolFloatToShort, [18](#)
  - Photon::Voice::BufferReaderPushAdapterBase, [19](#)
  - Photon::Voice::IServiceable, [37](#)
  - Photon::Voice::LoadBalancingFrontend, [44](#)
  - Photon::Voice::VoiceClient, [86](#)
- SetCallback
  - Photon::Voice::AudioUtil::ToneAudioPusher, [78](#)
  - Photon::Voice::IAudioPusher, [26](#)
- Settings
  - Photon::Voice::Unity::VoiceConnection, [89](#)
- SetupDebugSpeaker
  - Photon::Voice::PUN::PhotonVoiceView, [61](#)
- SignalHint
  - POpusCodec::Enums, [8](#)
- SourceSamplingRate
  - Photon::Voice::VoiceInfo, [99](#)
- SourceType
  - Photon::Voice::Unity::Recorder, [66](#)
- Speaker, [70](#)
- SpeakerFactory
  - Photon::Voice::Unity::VoiceConnection, [89](#)
- SpeakerInUse
  - Photon::Voice::PUN::PhotonVoiceView, [62](#)
- SpeakerLinked
  - Photon::Voice::Unity::VoiceConnection, [90](#)
- SpeakerPrefab
  - Photon::Voice::Unity::VoiceConnection, [90](#)
- SpeexLib, [71](#)
- SpeexProcessor, [76](#)
- SpeexProcessor.AECLatencyResultType, [9](#)
- Stereo
  - POpusCodec::Enums, [8](#)
- SuperWideband
  - POpusCodec::Enums, [7](#)
- SuppressInfoDuplicateWarning
  - Photon::Voice::VoiceClient, [87](#)
- TestTone, [77](#)
- Threshold
  - Photon::Voice::AudioUtil::IVoiceDetector, [38](#)
  - Photon::Voice::AudioUtil::VoiceDetector, [92](#)
- ToneAudioPusher
  - Photon::Voice::AudioUtil::ToneAudioPusher, [78](#)
- ToneAudioReader, [78](#)
  - Photon::Voice::AudioUtil::ToneAudioReader, [79](#)
- TransmitEnabled
  - Photon::Voice::LocalVoice, [46](#)
  - Photon::Voice::Unity::Recorder, [66](#)
- TryGetChannelID
  - Photon::Voice::VoiceEventCode, [97](#)
- TypeConvert
  - Photon::Voice::Unity::Recorder, [66](#)
- UnityAndroidAudioInAEC, [80](#)
- UnityAudioOut, [80](#)
- UnityMicrophoneDevice
  - Photon::Voice::Unity::Recorder, [67](#)
- UnsupportedCodecException, [81](#)
  - Photon::Voice::UnsupportedCodecException, [81](#)
- UnsupportedSampleTypeException, [81](#)
  - Photon::Voice::UnsupportedSampleTypeException, [82](#)
- UsePrimaryRecorder
  - Photon::Voice::PUN::PhotonVoiceView, [61](#)
- UserData
  - Photon::Voice::Unity::Recorder, [67](#)

- Photon::Voice::VoiceInfo, 99
- Visible
  - Photon::Voice::Unity::UtilityScripts::PhotonVoice↔LagSimulationGui, 58
- Voice
  - POpusCodec::Enums, 8
- VoiceClient, 82
  - Photon::Voice::LoadBalancingFrontend, 44
  - Photon::Voice::Unity::VoiceConnection, 90
- VoiceComponent, 87
- VoiceConnection, 88
- VoiceDetection
  - Photon::Voice::Unity::Recorder, 67
- VoiceDetectionDelayMs
  - Photon::Voice::Unity::Recorder, 67
- VoiceDetectionThreshold
  - Photon::Voice::Unity::Recorder, 67
- VoiceDetector
  - Photon::Voice::ILocalVoiceAudio, 34
  - Photon::Voice::Unity::Recorder, 67
- VoiceDetectorCalibrate
  - Photon::Voice::AudioUtil::VoiceDetectorCalibration, 93
  - Photon::Voice::ILocalVoiceAudio, 33
  - Photon::Voice::LocalVoiceAudio, 48
  - Photon::Voice::LocalVoiceAudioDummy, 49
  - Photon::Voice::Unity::Recorder, 65
- VoiceDetectorCalibrating
  - Photon::Voice::ILocalVoiceAudio, 34
  - Photon::Voice::LocalVoiceAudio, 48
  - Photon::Voice::Unity::Recorder, 67
- VoiceDetectorCalibration
  - Photon::Voice::AudioUtil::VoiceDetectorCalibration, 93
- VoiceDetectorFloat
  - Photon::Voice::AudioUtil::VoiceDetectorFloat, 94
- VoiceDetectorShort
  - Photon::Voice::AudioUtil::VoiceDetectorShort, 95
- VoiceEventCode, 95
- VoiceId
  - Photon::Voice::RemoteVoiceInfo, 68
- VoiceInfo, 97
- VoiceLevelDetectCalibrate
  - Photon::Voice::AudioUtil::VoiceLevelDetect↔Calibrate, 100
- VoiceLogger, 101
- VoiceRoomNameSuffix
  - Photon::Voice::PUN::PhotonVoiceNetwork, 60
- Voip
  - POpusCodec::Enums, 8
- WebRTCAudioLib, 101
- WebRTCAudioLib.ConfigParam, 19
- WebRTCAudioLib.Param, 57
- WebRTCAudioProcessor, 102
- WebRtcAudioDsp, 101
- Wideband
  - POpusCodec::Enums, 7
- Width
  - Photon::Voice::VoiceInfo, 99
- WindowId
  - Photon::Voice::Unity::UtilityScripts::PhotonVoice↔LagSimulationGui, 58
- WindowRect
  - Photon::Voice::Unity::UtilityScripts::PhotonVoice↔LagSimulationGui, 58