

Photon Voice

v2.10

Generated by Doxygen 1.8.10

Wed Sep 25 2019 15:38:55



# 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 . . . . .	5
2.2.1.1	Codec . . . . .	5
2.3	Photon.Voice.IOS Namespace Reference . . . . .	5
2.3.1	Enumeration Type Documentation . . . . .	6
2.3.1.1	AudioSessionCategory . . . . .	6
2.3.1.2	AudioSessionCategoryOption . . . . .	6
2.3.1.3	AudioSessionMode . . . . .	7
2.4	Photon.Voice.PUN Namespace Reference . . . . .	8
2.5	Photon.Voice.Unity Namespace Reference . . . . .	8
2.6	Photon.Voice.Unity.UtilityScripts Namespace Reference . . . . .	8
2.7	POpusCodec Namespace Reference . . . . .	9
2.8	POpusCodec.Enums Namespace Reference . . . . .	9
2.8.1	Enumeration Type Documentation . . . . .	9
2.8.1.1	Bandwidth . . . . .	9
2.8.1.2	Channels . . . . .	9
2.8.1.3	Delay . . . . .	10
2.8.1.4	OpusApplicationType . . . . .	10
2.8.1.5	SignalHint . . . . .	10
<b>3</b>	<b>Class Documentation</b>	<b>11</b>
3.1	AudioClipWrapper Class Reference . . . . .	11
3.2	AudioDesc Class Reference . . . . .	11
3.3	AudioInEnumerator Class Reference . . . . .	11
3.4	AudioOutCapture Class Reference . . . . .	12
3.5	AudioSessionParameters Struct Reference . . . . .	12
3.6	AudioSessionParametersPresets Class Reference . . . . .	12
3.6.1	Member Data Documentation . . . . .	13

3.6.1.1	Game	13
3.6.1.2	VoIP	13
3.7	AudioStreamPlayer< T > Class Template Reference	13
3.8	AudioUtil Class Reference	13
3.8.1	Detailed Description	14
3.8.2	Member Function Documentation	15
3.8.2.1	Convert(float[] src, short[] dst, int dstCount)	15
3.8.2.2	Convert(short[] src, float[] dst, int dstCount)	16
3.8.2.3	ForceToStereo< T >(T[] src, T[] dst, int srcChannels)	16
3.8.2.4	Resample< T >(T[] src, T[] dst, int dstCount, int channels)	16
3.8.2.5	ResampleAndConvert(short[] src, float[] dst, int dstCount, int channels)	16
3.8.2.6	ResampleAndConvert(float[] src, short[] dst, int dstCount, int channels)	17
3.9	BufferReaderPushAdapter< T > Class Template Reference	17
3.9.1	Detailed Description	17
3.9.2	Constructor & Destructor Documentation	17
3.9.2.1	BufferReaderPushAdapter(LocalVoice localVoice, IDataReader< T > reader)	17
3.9.3	Member Function Documentation	18
3.9.3.1	Service(LocalVoice localVoice)	18
3.10	BufferReaderPushAdapterAsyncPool< T > Class Template Reference	18
3.10.1	Detailed Description	18
3.10.2	Constructor & Destructor Documentation	18
3.10.2.1	BufferReaderPushAdapterAsyncPool(LocalVoice localVoice, IDataReader< T > reader)	18
3.10.3	Member Function Documentation	18
3.10.3.1	Service(LocalVoice localVoice)	18
3.11	BufferReaderPushAdapterAsyncPoolCopy< T > Class Template Reference	19
3.11.1	Detailed Description	19
3.11.2	Constructor & Destructor Documentation	19
3.11.2.1	BufferReaderPushAdapterAsyncPoolCopy(LocalVoice localVoice, IDataReader< T > reader)	19
3.11.3	Member Function Documentation	19
3.11.3.1	Service(LocalVoice localVoice)	19
3.12	BufferReaderPushAdapterAsyncPoolFloatToShort Class Reference	20
3.12.1	Detailed Description	20
3.12.2	Constructor & Destructor Documentation	20
3.12.2.1	BufferReaderPushAdapterAsyncPoolFloatToShort(LocalVoice localVoice, IDataReader< float > reader)	20
3.12.3	Member Function Documentation	20
3.12.3.1	Service(LocalVoice localVoice)	20
3.13	BufferReaderPushAdapterBase< T > Class Template Reference	20
3.13.1	Detailed Description	21

3.13.2	Constructor & Destructor Documentation	21
3.13.2.1	BufferReaderPushAdapterBase(IDataReader< T > reader)	21
3.13.3	Member Function Documentation	21
3.13.3.1	Dispose()	21
3.13.3.2	Service(LocalVoice localVoice)	21
3.14	WebRTCAudioLib.ConfigParam Struct Reference	22
3.15	ConnectAndJoin Class Reference	22
3.16	OpusCodec.Decoder< T > Class Template Reference	22
3.16.1	Member Function Documentation	23
3.16.1.1	Input(byte[] buf)	23
3.16.1.2	Open(VoiceInfo i)	23
3.17	RawCodec.Decoder< T > Class Template Reference	23
3.17.1	Member Function Documentation	24
3.17.1.1	Input(byte[] buf)	24
3.17.1.2	Open(VoiceInfo info)	24
3.18	OpusCodec.DecoderFactory Class Reference	24
3.19	OpusCodec.DecoderFloat Class Reference	24
3.20	OpusCodec.DecoderShort Class Reference	24
3.21	OpusCodec.Encoder< T > Class Template Reference	25
3.22	RawCodec.Encoder< T > Class Template Reference	25
3.23	OpusCodec.EncoderFloat Class Reference	25
3.24	OpusCodec.EncoderShort Class Reference	26
3.25	OpusCodec.Factory Class Reference	26
3.26	FactoryPrimitiveArrayPool< T > Class Template Reference	26
3.26.1	Detailed Description	26
3.27	FactoryReusableArray< T > Class Template Reference	27
3.27.1	Detailed Description	27
3.28	Framer< T > Class Template Reference	27
3.28.1	Detailed Description	27
3.28.2	Constructor & Destructor Documentation	28
3.28.2.1	Framer(int frameSize)	28
3.28.3	Member Function Documentation	28
3.28.3.1	Count(int bufLen)	28
3.28.3.2	Frame(T[] buf)	28
3.29	IAudioDesc Interface Reference	28
3.29.1	Detailed Description	28
3.29.2	Property Documentation	29
3.29.2.1	Channels	29
3.29.2.2	Error	29
3.29.2.3	SamplingRate	29

3.30	<a href="#">IAudioOut&lt; T &gt; Interface Template Reference</a>	29
3.31	<a href="#">IAudioPusher&lt; T &gt; Interface Template Reference</a>	29
3.31.1	<a href="#">Detailed Description</a>	29
3.31.2	<a href="#">Member Function Documentation</a>	30
3.31.2.1	<a href="#">SetCallback(Action&lt; T[]&gt; callback, ObjectFactory&lt; T[], int &gt; bufferFactory)</a>	30
3.32	<a href="#">IAudioReader&lt; T &gt; Interface Template Reference</a>	30
3.32.1	<a href="#">Detailed Description</a>	30
3.33	<a href="#">IDataReader&lt; T &gt; Interface Template Reference</a>	30
3.33.1	<a href="#">Detailed Description</a>	30
3.33.2	<a href="#">Member Function Documentation</a>	30
3.33.2.1	<a href="#">Read(T[] buffer)</a>	30
3.34	<a href="#">IDecoder Interface Reference</a>	31
3.34.1	<a href="#">Detailed Description</a>	31
3.34.2	<a href="#">Member Function Documentation</a>	31
3.34.2.1	<a href="#">Input(byte[] buf)</a>	31
3.34.2.2	<a href="#">Open(VoiceInfo info)</a>	31
3.34.3	<a href="#">Property Documentation</a>	32
3.34.3.1	<a href="#">Error</a>	32
3.35	<a href="#">IDecoderQueuedOutputImageNative Interface Reference</a>	32
3.36	<a href="#">IEncoder Interface Reference</a>	32
3.36.1	<a href="#">Detailed Description</a>	32
3.36.2	<a href="#">Member Function Documentation</a>	32
3.36.2.1	<a href="#">DequeueOutput()</a>	32
3.36.3	<a href="#">Property Documentation</a>	33
3.36.3.1	<a href="#">Error</a>	33
3.36.3.2	<a href="#">Output</a>	33
3.37	<a href="#">IEncoderDirect&lt; B &gt; Interface Template Reference</a>	33
3.37.1	<a href="#">Detailed Description</a>	33
3.37.2	<a href="#">Member Function Documentation</a>	33
3.37.2.1	<a href="#">Input(B buf)</a>	33
3.38	<a href="#">AudioUtil.ILevelMeter Interface Reference</a>	33
3.38.1	<a href="#">Detailed Description</a>	34
3.38.2	<a href="#">Member Function Documentation</a>	34
3.38.2.1	<a href="#">ResetAccumAvgPeakAmp()</a>	34
3.38.3	<a href="#">Property Documentation</a>	34
3.38.3.1	<a href="#">AccumAvgPeakAmp</a>	34
3.38.3.2	<a href="#">CurrentAvgAmp</a>	34
3.38.3.3	<a href="#">CurrentPeakAmp</a>	34
3.39	<a href="#">ILocalVoiceAudio Interface Reference</a>	34
3.39.1	<a href="#">Detailed Description</a>	35

3.39.2	Member Function Documentation	35
3.39.2.1	VoiceDetectorCalibrate(int durationMs, Action< float > onCalibrated=null)	35
3.39.3	Property Documentation	35
3.39.3.1	LevelMeter	35
3.39.3.2	VoiceDetector	35
3.39.3.3	VoiceDetectorCalibrating	35
3.40	ILoggable Interface Reference	35
3.41	ILogger Interface Reference	36
3.42	ImageBufferInfo Class Reference	36
3.43	ImageBufferNative Class Reference	36
3.44	ImageBufferNativeAlloc Class Reference	36
3.45	ImageBufferNativeGCHandleSinglePlane Class Reference	37
3.46	ImageBufferNativePool< T > Class Template Reference	37
3.47	ImageInputBuf Struct Reference	37
3.48	ImageOutputBuf Struct Reference	38
3.49	IOSAudioForceToSpeaker Class Reference	38
3.50	IProcessor< T > Interface Template Reference	38
3.50.1	Detailed Description	38
3.50.2	Member Function Documentation	38
3.50.2.1	Process(T[] buf)	38
3.51	IServiceable Interface Reference	38
3.51.1	Detailed Description	39
3.51.2	Member Function Documentation	39
3.51.2.1	Service(LocalVoice localVoice)	39
3.52	ISyncAudioOut< T > Interface Template Reference	39
3.53	AudioUtil.IVoiceDetector Interface Reference	39
3.53.1	Detailed Description	40
3.53.2	Property Documentation	40
3.53.2.1	ActivityDelayMs	40
3.53.2.2	Detected	40
3.53.2.3	DetectedTime	40
3.53.2.4	On	40
3.53.2.5	Threshold	40
3.53.3	Event Documentation	40
3.53.3.1	OnDetected	40
3.54	IVoiceTransport Interface Reference	40
3.55	AudioUtil.LevelMeter< T > Class Template Reference	41
3.55.1	Detailed Description	41
3.55.2	Member Function Documentation	41
3.55.2.1	Process(T[] buf)	41

3.55.2.2	ResetAccumAvgPeakAmp()	42
3.56	AudioUtil.LevelMeterDummy Class Reference	42
3.56.1	Detailed Description	42
3.56.2	Member Function Documentation	42
3.56.2.1	ResetAccumAvgPeakAmp()	42
3.57	AudioUtil.LevelMeterFloat Class Reference	42
3.57.1	Detailed Description	43
3.57.2	Constructor & Destructor Documentation	43
3.57.2.1	LevelMeterFloat(int samplingRate, int numChannels)	43
3.58	AudioUtil.LevelMeterShort Class Reference	43
3.58.1	Detailed Description	43
3.58.2	Constructor & Destructor Documentation	43
3.58.2.1	LevelMeterShort(int samplingRate, int numChannels)	43
3.59	LoadBalancingFrontend Class Reference	44
3.60	LoadBalancingTransport Class Reference	44
3.60.1	Detailed Description	45
3.60.2	Constructor & Destructor Documentation	45
3.60.2.1	LoadBalancingTransport(ConnectionProtocol connectionProtocol=Connection← Protocol.Udp)	45
3.60.3	Member Function Documentation	45
3.60.3.1	Dispose()	45
3.60.3.2	SendDebugEchoVoicesInfo(int channelId)	45
3.60.3.3	Service()	45
3.60.4	Property Documentation	46
3.60.4.1	GlobalInterestGroup	46
3.60.4.2	VoiceClient	46
3.61	LocalVoice Class Reference	46
3.61.1	Detailed Description	47
3.61.2	Member Function Documentation	47
3.61.2.1	RemoveSelf()	47
3.61.3	Property Documentation	47
3.61.3.1	DebugEchoMode	47
3.61.3.2	Encrypt	47
3.61.3.3	FramesSent	47
3.61.3.4	FramesSentBytes	47
3.61.3.5	Info	48
3.61.3.6	InterestGroup	48
3.61.3.7	IsCurrentlyTransmitting	48
3.61.3.8	LocalUserServiceable	48
3.61.3.9	Reliable	48



3.61.3.10 TransmitEnabled . . . . .	48
3.62 LocalVoiceAudio< T > Class Template Reference . . . . .	48
3.62.1 Detailed Description . . . . .	49
3.62.2 Member Function Documentation . . . . .	49
3.62.2.1 Create(VoiceClient voiceClient, byte voiceId, IEncoder encoder, VoiceInfo voiceInfo, IAudioDesc audioSourceDesc, int channelId) . . . . .	49
3.62.2.2 VoiceDetectorCalibrate(int durationMs, Action< float > onCalibrated=null) . . . . .	49
3.62.3 Property Documentation . . . . .	50
3.62.3.1 VoiceDetectorCalibrating . . . . .	50
3.63 LocalVoiceAudioDummy Class Reference . . . . .	50
3.63.1 Detailed Description . . . . .	50
3.63.2 Member Function Documentation . . . . .	50
3.63.2.1 VoiceDetectorCalibrate(int durationMs, Action< float > onCalibrated=null) . . . . .	50
3.63.3 Member Data Documentation . . . . .	51
3.63.3.1 Dummy . . . . .	51
3.64 LocalVoiceAudioFloat Class Reference . . . . .	51
3.64.1 Detailed Description . . . . .	51
3.65 LocalVoiceAudioShort Class Reference . . . . .	51
3.65.1 Detailed Description . . . . .	51
3.66 LocalVoiceFramed< T > Class Template Reference . . . . .	51
3.66.1 Detailed Description . . . . .	52
3.66.2 Member Function Documentation . . . . .	52
3.66.2.1 AddPostProcessor(params IProcessor< T >[] processors) . . . . .	52
3.66.2.2 AddPreProcessor(params IProcessor< T >[] processors) . . . . .	52
3.66.2.3 ClearProcessors() . . . . .	52
3.66.2.4 Dispose() . . . . .	53
3.66.2.5 PushData(T[] buf) . . . . .	53
3.66.2.6 PushDataAsync(T[] buf) . . . . .	53
3.66.3 Property Documentation . . . . .	53
3.66.3.1 PushDataAsyncReady . . . . .	53
3.67 LocalVoiceFramedBase Class Reference . . . . .	53
3.67.1 Detailed Description . . . . .	53
3.67.2 Property Documentation . . . . .	53
3.67.2.1 FrameSize . . . . .	53
3.68 Logger Class Reference . . . . .	54
3.69 MicAmplifier Class Reference . . . . .	54
3.70 MicAmplifierFloat Class Reference . . . . .	54
3.71 MicAmplifierShort Class Reference . . . . .	54
3.72 MicWrapper Class Reference . . . . .	55
3.73 ObjectFactory< TType, TInfo > Interface Template Reference . . . . .	55

3.73.1 Detailed Description . . . . .	55
3.74 ObjectPool< TType, TInfo > Class Template Reference . . . . .	56
3.74.1 Detailed Description . . . . .	56
3.74.2 Constructor & Destructor Documentation . . . . .	57
3.74.2.1 ObjectPool(int capacity, string name) . . . . .	57
3.74.2.2 ObjectPool(int capacity, string name, TInfo info) . . . . .	57
3.74.3 Member Function Documentation . . . . .	57
3.74.3.1 AcquireOrCreate() . . . . .	57
3.74.3.2 AcquireOrCreate(TInfo info) . . . . .	57
3.74.3.3 Dispose() . . . . .	57
3.74.3.4 Init(TInfo info) . . . . .	57
3.74.3.5 Release(TType obj, TInfo objInfo) . . . . .	58
3.74.3.6 Release(TType obj) . . . . .	58
3.74.4 Property Documentation . . . . .	58
3.74.4.1 Info . . . . .	58
3.75 OpusCodec Class Reference . . . . .	58
3.76 OpusDecoder Class Reference . . . . .	58
3.77 OpusEncoder Class Reference . . . . .	59
3.77.1 Property Documentation . . . . .	59
3.77.1.1 EncoderDelay . . . . .	59
3.78 OpusException Class Reference . . . . .	60
3.79 WebRTCAudioLib.Param Struct Reference . . . . .	60
3.80 Recorder.PhotonVoiceCreatedParams Class Reference . . . . .	60
3.81 PhotonVoiceCreatedParams Class Reference . . . . .	60
3.82 PhotonVoiceLagSimulationGui Class Reference . . . . .	61
3.83 PhotonVoiceNetwork Class Reference . . . . .	61
3.83.1 Detailed Description . . . . .	62
3.83.2 Member Function Documentation . . . . .	62
3.83.2.1 ConnectAndJoinRoom() . . . . .	62
3.83.2.2 Disconnect() . . . . .	62
3.83.3 Member Data Documentation . . . . .	62
3.83.3.1 AutoConnectAndJoin . . . . .	62
3.83.3.2 AutoCreateSpeakerIfNotFound . . . . .	62
3.83.3.3 AutoLeaveAndDisconnect . . . . .	62
3.83.3.4 VoiceRoomNameSuffix . . . . .	62
3.83.4 Property Documentation . . . . .	62
3.83.4.1 Instance . . . . .	62
3.84 PhotonVoiceStatsGui Class Reference . . . . .	63
3.84.1 Detailed Description . . . . .	63
3.85 PhotonVoiceView Class Reference . . . . .	63

3.85.1 Detailed Description . . . . .	64
3.85.2 Member Function Documentation . . . . .	64
3.85.2.1 Init() . . . . .	64
3.85.3 Member Data Documentation . . . . .	64
3.85.3.1 AutoCreateRecorderIfNotFound . . . . .	64
3.85.3.2 SetupDebugSpeaker . . . . .	64
3.85.3.3 UsePrimaryRecorder . . . . .	64
3.85.4 Property Documentation . . . . .	64
3.85.4.1 IsRecorder . . . . .	64
3.85.4.2 IsRecording . . . . .	64
3.85.4.3 IsSetup . . . . .	64
3.85.4.4 IsSpeaker . . . . .	65
3.85.4.5 IsSpeaking . . . . .	65
3.85.4.6 RecorderInUse . . . . .	65
3.85.4.7 SpeakerInUse . . . . .	65
3.86 PrimitiveArrayPool< T > Class Template Reference . . . . .	65
3.86.1 Detailed Description . . . . .	65
3.87 RawCodec Class Reference . . . . .	65
3.88 Recorder Class Reference . . . . .	66
3.88.1 Detailed Description . . . . .	68
3.88.2 Member Function Documentation . . . . .	68
3.88.2.1 Init(VoiceClient voiceClient, object customObj=null) . . . . .	68
3.88.2.2 RestartRecording() . . . . .	68
3.88.2.3 StartRecording() . . . . .	68
3.88.2.4 StopRecording() . . . . .	68
3.88.2.5 VoiceDetectorCalibrate(int durationMs, Action< float > detectionEnded↔ Callback=null) . . . . .	68
3.88.3 Property Documentation . . . . .	68
3.88.3.1 AudioClip . . . . .	68
3.88.3.2 AudioGroup . . . . .	68
3.88.3.3 AutoStart . . . . .	69
3.88.3.4 Bitrate . . . . .	69
3.88.3.5 DebugEchoMode . . . . .	69
3.88.3.6 Encrypt . . . . .	69
3.88.3.7 FrameDuration . . . . .	69
3.88.3.8 InputFactory . . . . .	69
3.88.3.9 InterestGroup . . . . .	69
3.88.3.10 IsCurrentlyTransmitting . . . . .	69
3.88.3.11 IsInitialized . . . . .	69
3.88.3.12 IsRecording . . . . .	69

3.88.3.13 LevelMeter . . . . .	69
3.88.3.14 LoopAudioClip . . . . .	69
3.88.3.15 MicrophoneType . . . . .	70
3.88.3.16 PhotonMicrophoneDeviceId . . . . .	70
3.88.3.17 PhotonMicrophoneEnumerator . . . . .	70
3.88.3.18 ReliableMode . . . . .	70
3.88.3.19 RequiresRestart . . . . .	70
3.88.3.20 SamplingRate . . . . .	70
3.88.3.21 SourceType . . . . .	70
3.88.3.22 TransmitEnabled . . . . .	70
3.88.3.23 TypeConvert . . . . .	70
3.88.3.24 UnityMicrophoneDevice . . . . .	70
3.88.3.25 UserData . . . . .	70
3.88.3.26 VoiceDetection . . . . .	70
3.88.3.27 VoiceDetectionDelayMs . . . . .	71
3.88.3.28 VoiceDetectionThreshold . . . . .	71
3.88.3.29 VoiceDetector . . . . .	71
3.88.3.30 VoiceDetectorCalibrating . . . . .	71
3.89 RemoteVoiceInfo Class Reference . . . . .	71
3.89.1 Detailed Description . . . . .	71
3.89.2 Property Documentation . . . . .	71
3.89.2.1 ChannelId . . . . .	71
3.89.2.2 Info . . . . .	71
3.89.2.3 playerId . . . . .	71
3.89.2.4 VoiceId . . . . .	72
3.90 RemoteVoiceLink Class Reference . . . . .	72
3.91 RemoteVoiceOptions Struct Reference . . . . .	72
3.91.1 Detailed Description . . . . .	72
3.91.2 Member Function Documentation . . . . .	73
3.91.2.1 SetOutput(Action< float[]> output) . . . . .	73
3.91.3 Property Documentation . . . . .	73
3.91.3.1 Decoder . . . . .	73
3.91.3.2 OnRemoteVoiceRemoveAction . . . . .	73
3.92 AudioUtil.Resampler< T > Class Template Reference . . . . .	73
3.92.1 Detailed Description . . . . .	73
3.92.2 Constructor & Destructor Documentation . . . . .	73
3.92.2.1 Resampler(int dstSize, int channels) . . . . .	73
3.92.3 Member Function Documentation . . . . .	74
3.92.3.1 Process(T[] buf) . . . . .	74
3.93 Speaker Class Reference . . . . .	74

3.93.1 Detailed Description . . . . .	74
3.93.2 Property Documentation . . . . .	75
3.93.2.1 Actor . . . . .	75
3.93.2.2 IsLinked . . . . .	75
3.93.2.3 IsPlaying . . . . .	75
3.93.2.4 Lag . . . . .	75
3.93.2.5 OnRemoteVoiceRemoveAction . . . . .	75
3.94 TestTone Class Reference . . . . .	75
3.95 AudioUtil.ToneAudioPusher< T > Class Template Reference . . . . .	75
3.95.1 Detailed Description . . . . .	76
3.95.2 Constructor & Destructor Documentation . . . . .	76
3.95.2.1 ToneAudioPusher(int frequency=440, int bufSizeMs=100, int sampling↵ Rate=441000, int channels=2) . . . . .	76
3.95.3 Member Function Documentation . . . . .	76
3.95.3.1 SetCallback(Action< T[]> callback, ObjectFactory< T[], int > bufferFactory) . . . . .	76
3.96 ToneAudioReader Class Reference . . . . .	76
3.97 AudioUtil.ToneAudioReader< T > Class Template Reference . . . . .	76
3.97.1 Detailed Description . . . . .	77
3.97.2 Constructor & Destructor Documentation . . . . .	77
3.97.2.1 ToneAudioReader(Func< double > clockSec=null, double frequency=440, int samplingRate=441000, int channels=2) . . . . .	77
3.97.3 Member Function Documentation . . . . .	77
3.97.3.1 Read(T[] buf) . . . . .	77
3.97.4 Property Documentation . . . . .	78
3.97.4.1 Channels . . . . .	78
3.97.4.2 Error . . . . .	78
3.97.4.3 SamplingRate . . . . .	78
3.98 UnityAndroidAudioInAEC Class Reference . . . . .	78
3.99 UnityAudioOut Class Reference . . . . .	78
3.100UnsupportedCodecException Class Reference . . . . .	79
3.100.1 Detailed Description . . . . .	79
3.100.2 Constructor & Destructor Documentation . . . . .	79
3.100.2.1 UnsupportedCodecException(string info, Codec codec, ILogger logger) . . . . .	79
3.101UnsupportedSampleTypeException Class Reference . . . . .	79
3.101.1 Detailed Description . . . . .	79
3.101.2 Constructor & Destructor Documentation . . . . .	80
3.101.2.1 UnsupportedSampleTypeException(Type t) . . . . .	80
3.102OpusCodec.Util Class Reference . . . . .	80
3.103VoiceClient Class Reference . . . . .	80
3.103.1 Detailed Description . . . . .	81

3.103.2 Member Function Documentation	81
3.103.2.1 CreateLocalVoice(VoiceInfo voiceInfo, int channelId=0, IEncoder encoder=null)	81
3.103.2.2 CreateLocalVoiceAudio< T >(VoiceInfo voiceInfo, IAudioDesc audioSourceDesc, int channelId=0, IEncoder encoder=null)	81
3.103.2.3 CreateLocalVoiceAudioFromSource(VoiceInfo voiceInfo, IAudioDesc source, bool forceShort=false, int channelId=0, IEncoder encoder=null)	82
3.103.2.4 CreateLocalVoiceFramed< T >(VoiceInfo voiceInfo, int frameSize, int channelId=0, IEncoder encoder=null)	82
3.103.2.5 LocalVoicesInChannel(int channelId)	83
3.103.2.6 RemoteVoiceInfoDelegate(int channelId, int playerId, byte voiceId, VoiceInfo voiceInfo, ref RemoteVoiceOptions options)	83
3.103.2.7 RemoveLocalVoice(LocalVoice voice)	83
3.103.2.8 Service()	83
3.103.3 Property Documentation	83
3.103.3.1 DebugLostPercent	83
3.103.3.2 FramesLost	83
3.103.3.3 FramesReceived	83
3.103.3.4 FramesSent	83
3.103.3.5 FramesSentBytes	83
3.103.3.6 LocalVoices	84
3.103.3.7 OnRemoteVoiceInfoAction	84
3.103.3.8 RemoteVoiceInfos	84
3.103.3.9 RoundTripTime	84
3.103.3.10RoundTripTimeVariance	84
3.103.3.11SuppressInfoDuplicateWarning	84
3.104 VoiceComponent Class Reference	84
3.105 VoiceConnection Class Reference	84
3.105.1 Detailed Description	86
3.105.2 Member Function Documentation	86
3.105.2.1 ConnectUsingSettings(AppSettings overwriteSettings=null)	86
3.105.2.2 Dispatch()	86
3.105.3 Member Data Documentation	86
3.105.3.1 MinimalTimeScaleToDispatchInFixedUpdate	86
3.105.3.2 PrimaryRecorder	86
3.105.3.3 Settings	86
3.105.3.4 SpeakerFactory	87
3.105.4 Property Documentation	87
3.105.4.1 ClientState	87
3.105.4.2 FramesLostPercent	87
3.105.4.3 FramesLostPerSecond	87
3.105.4.4 FramesReceivedPerSecond	87

3.105.4.5 Logger	87
3.105.4.6 LogLevel	87
3.105.4.7 SpeakerPrefab	87
3.105.4.8 VoiceClient	87
3.105.5 Event Documentation	87
3.105.5.1 RemoteVoiceAdded	87
3.105.5.2 SpeakerLinked	87
3.106AudioUtil.VoiceDetector< T > Class Template Reference	88
3.106.1 Detailed Description	88
3.106.2 Member Function Documentation	88
3.106.2.1 Process(T[] buf)	88
3.106.3 Property Documentation	89
3.106.3.1 ActivityDelayMs	89
3.106.3.2 Detected	89
3.106.3.3 DetectedTime	89
3.106.3.4 On	89
3.106.3.5 Threshold	89
3.106.4 Event Documentation	89
3.106.4.1 OnDetected	89
3.107AudioUtil.VoiceDetectorCalibration< T > Class Template Reference	89
3.107.1 Detailed Description	90
3.107.2 Constructor & Destructor Documentation	90
3.107.2.1 VoiceDetectorCalibration(IVoiceDetector voiceDetector, ILevelMeter levelMeter, int samplingRate, int channels)	90
3.107.3 Member Function Documentation	90
3.107.3.1 Calibrate(int durationMs, Action< float > onCalibrated=null)	90
3.107.3.2 Process(T[] buf)	90
3.108AudioUtil.VoiceDetectorDummy Class Reference	91
3.108.1 Detailed Description	91
3.109AudioUtil.VoiceDetectorFloat Class Reference	91
3.109.1 Detailed Description	91
3.109.2 Constructor & Destructor Documentation	91
3.109.2.1 VoiceDetectorFloat(int samplingRate, int numChannels)	91
3.110AudioUtil.VoiceDetectorShort Class Reference	92
3.110.1 Detailed Description	92
3.110.2 Constructor & Destructor Documentation	92
3.110.2.1 VoiceDetectorShort(int samplingRate, int numChannels)	92
3.111VoiceEvent Class Reference	92
3.111.1 Member Data Documentation	92
3.111.1.1 Code	92

3.112VoiceInfo Struct Reference . . . . .	93
3.112.1 Detailed Description . . . . .	93
3.112.2 Member Function Documentation . . . . .	93
3.112.2.1 CreateAudio(Codec codec, int samplingRate, int channels, int frameDurationUs, object userdata=null) . . . . .	93
3.112.2.2 CreateAudioOpus(POpusCodec.Enums.SamplingRate samplingRate, int chan- nels, OpusCodec.FrameDuration frameDurationUs, int bitrate, object userdata=null) . . . . .	94
3.112.3 Property Documentation . . . . .	94
3.112.3.1 Bitrate . . . . .	94
3.112.3.2 Channels . . . . .	94
3.112.3.3 FrameDurationSamples . . . . .	94
3.112.3.4 FrameDurationUs . . . . .	94
3.112.3.5 FrameSize . . . . .	94
3.112.3.6 Height . . . . .	94
3.112.3.7 SamplingRate . . . . .	95
3.112.3.8 UserData . . . . .	95
3.112.3.9 Width . . . . .	95
3.113AudioUtil.VoiceLevelDetectCalibrate< T > Class Template Reference . . . . .	95
3.113.1 Detailed Description . . . . .	95
3.113.2 Constructor & Destructor Documentation . . . . .	95
3.113.2.1 VoiceLevelDetectCalibrate(int samplingRate, int channels) . . . . .	95
3.113.3 Member Function Documentation . . . . .	96
3.113.3.1 Calibrate(int durationMs, Action< float > onCalibrated=null) . . . . .	96
3.113.3.2 Process(T[] buf) . . . . .	96
3.113.4 Property Documentation . . . . .	96
3.113.4.1 LevelMeter . . . . .	96
3.113.4.2 VoiceDetector . . . . .	96
3.114VoiceLogger Class Reference . . . . .	96
3.115WebRtcAudioDsp Class Reference . . . . .	97
3.116WebRTCAudioLib Class Reference . . . . .	97
3.117WebRTCAudioProcessor Class Reference . . . . .	98
<b>Index</b>	<b>99</b>



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

### Classes

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

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

- class [Framer](#)  
*Utility class to re-frame audio packets.*
- interface [IAudioDesc](#)  
*Audio Source interface.*
- interface [IAudioOut](#)
- interface [IAudioPusher](#)  
*Audio Pusher interface.*
- interface [IAudioReader](#)  
*Audio Reader interface.*
- interface [IDataReader](#)  
*Interface for pulling data, in case this is more appropriate than pushing it.*
- interface [IDecoder](#)  
*Generic decoder interface.*
- interface [IDecoderQueuedOutputImageNative](#)
- interface [IEncoder](#)  
*Generic encoder interface.*
- interface [IEncoderDirect](#)  
*Interface for an encoder which consumes input data via explicit call.*
- interface [ILocalVoiceAudio](#)  
*Interface for an outgoing audio stream.*
- interface [ILogger](#)
- class [ImageBufferInfo](#)
- class [ImageBufferNative](#)
- class [ImageBufferNativeAlloc](#)
- class [ImageBufferNativeGCHandleSinglePlane](#)
- class [ImageBufferNativePool](#)
- struct [ImageInputBuf](#)
- struct [ImageOutputBuf](#)
- interface [IProcessor](#)  
*Audio Processor interface.*
- interface [IServiceable](#)  
*Interface for classes that want their [Service\(\)](#) function to be called regularly in the context of a [LocalVoice](#).*
- interface [ISyncAudioOut](#)
- interface [IVoiceTransport](#)
- class [LoadBalancingFrontend](#)
- class [LoadBalancingTransport](#)  
*Extends LoadBalancingClient with audio streaming functionality.*
- class [LocalVoice](#)  
*Represents outgoing data stream.*
- class [LocalVoiceAudio](#)  
*Outgoing audio stream.*
- class [LocalVoiceAudioDummy](#)  
*Dummy [LocalVoiceAudio](#)*
- class [LocalVoiceAudioFloat](#)  
*Specialization of [LocalVoiceAudio](#) for float audio*
- class [LocalVoiceAudioShort](#)  
*Specialization of [LocalVoiceAudio](#) for short audio*
- class [LocalVoiceFramed](#)  
*Typed re-framing [LocalVoice](#)*
- class [LocalVoiceFramedBase](#)

- Typed re-framing [LocalVoice](#)*
- interface [ObjectFactory](#)
  - Uniform interface to [ObjectPool](#)<TType, TInfo> and single reusable object.*
- class [ObjectPool](#)
  - Generic Pool to re-use objects of a certain type (TType) that optionally match a certain property or set of properties (TInfo).*
- class [OpusCodec](#)
- class **PhotonTransportProtocol**
- class [PrimitiveArrayPool](#)
  - Pool of Arrays with components of type T, with [ObjectPool](#) info being the array's size.*
- class [RawCodec](#)
- class **RemoteVoice**
- class [RemoteVoiceInfo](#)
  - Information about a remote voice (incoming stream).*
- struct [RemoteVoiceOptions](#)
  - Event Actions and other options for a remote voice (incoming stream).*
- class [UnsupportedCodecException](#)
  - Exception thrown if an unsupported codec is encountered.*
- class [UnsupportedSampleTypeException](#)
  - Exception thrown if an unsupported audio sample type is encountered.*
- class [VoiceClient](#)
  - Voice client interact with other clients on network via [IVoiceTransport](#).*
- class [VoiceEvent](#)
- struct [VoiceInfo](#)
  - Describes stream properties.*
- class [WebRTCAudioLib](#)
- class [WebRTCAudioProcessor](#)

## Enumerations

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

### 2.2.1 Enumeration Type Documentation

#### 2.2.1.1 enum **Codec** [strong]

Enum for Media Codecs supported by PhotonVoice.

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

Enumerator

**AudioOpus** OPUS audio

## 2.3 Photon.Voice.IOS Namespace Reference

### Classes

- struct [AudioSessionParameters](#)
- class [AudioSessionParametersPresets](#)

## Enumerations

- enum [AudioSessionCategory](#)
- enum [AudioSessionMode](#)
- enum [AudioSessionCategoryOption](#)

### 2.3.1 Enumeration Type Documentation

#### 2.3.1.1 enum `AudioSessionCategory` [strong]

##### Enumerator

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

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

**Playback** Use this category for music tracks. `API_AVAILABLE(ios(3.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);`

**Record** Use this category when recording audio. `API_AVAILABLE(ios(3.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);`

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

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

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

#### 2.3.1.2 enum `AudioSessionCategoryOption` [strong]

##### Enumerator

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

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

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

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

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

### 2.3.1.3 enum AudioSessionMode [strong]

#### Enumerator

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

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

**VideoRecording** Only valid with AVAudioSessionCategoryPlayAndRecord or AVAudioSessionCategoryRecord. Modifies the audio routing options and may engage appropriate system-supplied signal processing. API\_AVAILABLE(ios(5.0), watchos(2.0), tvos(9.0)) API\_UNAVAILABLE(macos);

**Measurement** Appropriate for applications that wish to minimize the effect of system-supplied signal processing for input and/or output audio signals. API\_AVAILABLE(ios(5.0), watchos(2.0), tvos(9.0)) API\_UNAVAILABLE(macos);

**MoviePlayback** Engages appropriate output signal processing for movie playback scenarios. Currently only applied during playback over built-in speaker. API\_AVAILABLE(ios(6.0), watchos(2.0), tvos(9.0)) API\_UNAVAILABLE(macos);

**VideoChat** Only valid with kAudioSessionCategory\_PlayAndRecord. Reduces the number of allowable audio routes to be only those that are appropriate for video chat applications. May engage appropriate system-supplied signal processing. Has the side effect of setting AVAudioSessionCategoryOptionAllowBluetooth and AVAudioSessionCategoryOptionDefaultToSpeaker. API\_AVAILABLE(ios(7.0), watchos(2.0), tvos(9.0)) API\_UNAVAILABLE(macos);

## 2.4 Photon.Voice.PUN Namespace Reference

### Classes

- class [PhotonVoiceNetwork](#)

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

- class [PhotonVoiceView](#)

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

## 2.5 Photon.Voice.Unity Namespace Reference

### Namespaces

- namespace [UtilityScripts](#)

### Classes

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

*Component representing outgoing audio stream in scene.*

- class [RemoteVoiceLink](#)
- class [Speaker](#)

*Component representing remote audio stream in local scene.*

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

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

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

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

### Classes

- class [ConnectAndJoin](#)
- class [MicAmplifier](#)
- class [MicAmplifierFloat](#)
- class [MicAmplifierShort](#)
- class [PhotonVoiceLagSimulationGui](#)
- class [PhotonVoiceStatsGui](#)



*Basic GUI to show traffic and health statistics of the connection to [Photon](#), toggled by shift+tab.*

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

## 2.7 POpusCodec Namespace Reference

### Namespaces

- namespace [Enums](#)

### Classes

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

## 2.8 POpusCodec.Enums Namespace Reference

### Enumerations

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

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

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

### 2.8.1 Enumeration Type Documentation

#### 2.8.1.1 enum **Bandwidth** : int `[strong]`

##### Enumerator

**Narrowband** Up to 4Khz  
**Mediumband** Up to 6Khz  
**Wideband** Up to 8Khz  
**SuperWideband** Up to 12Khz  
**Fullband** Up to 20Khz (High Definition)

#### 2.8.1.2 enum **Channels** : int `[strong]`

##### Enumerator

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

### 2.8.1.3 enum Delay [strong]

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

#### Enumerator

**Delay2dot5ms** 2.5ms

**Delay5ms** 5ms

**Delay10ms** 10ms

**Delay20ms** 20ms

**Delay40ms** 40ms

**Delay60ms** 60ms

### 2.8.1.4 enum OpusApplicationType : int [strong]

#### Enumerator

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

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

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

### 2.8.1.5 enum SignalHint : int [strong]

#### Enumerator

**Auto** (default)

**Voice** Bias thresholds towards choosing LPC or Hybrid modes

**Music** Bias thresholds towards choosing MDCT modes.

## Chapter 3

# Class Documentation

### 3.1 AudioClipWrapper Class Reference

Inherits [IAudioReader< float >](#).

#### Public Member Functions

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

#### Properties

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

### 3.2 AudioDesc Class Reference

Inherits [IAudioDesc](#).

#### Public Member Functions

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

#### Properties

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

### 3.3 AudioInEnumerator Class Reference

Inherits [IDisposable](#).

### Public Member Functions

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

### Public Attributes

- readonly bool **IsSupported** = false

### Properties

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

## 3.4 AudioOutCapture Class Reference

Inherits `MonoBehaviour`.

### Events

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

## 3.5 AudioSessionParameters Struct Reference

### Public Member Functions

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

### Public Attributes

- [AudioSessionCategory](#) **Category**
- [AudioSessionMode](#) **Mode**
- [AudioSessionCategoryOption](#)[] **CategoryOptions**

## 3.6 AudioSessionParametersPresets Class Reference

### Static Public Attributes

- static [AudioSessionParameters](#) **Game**
- static [AudioSessionParameters](#) **VoIP**

### 3.6.1 Member Data Documentation

#### 3.6.1.1 `AudioSessionParameters Game` [static]

Initial value:

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

#### 3.6.1.2 `AudioSessionParameters VoIP` [static]

Initial value:

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

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

## 3.7 `AudioStreamPlayer< T >` Class Template Reference

Inherits `IAudioOut< T >`.

### Public Member Functions

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

### Properties

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

## 3.8 `AudioUtil` Class Reference

Collection of Audio Utility functions and classes.

### Classes

- interface `ILevelMeter`  
*Audio Level Metering interface.*
- interface `IVoiceDetector`

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

## Static Public Member Functions

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

### 3.8.1 Detailed Description

Collection of Audio Utility functions and classes.

### 3.8.2 Member Function Documentation

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

Convert audio buffer from float to short samples.

## Parameters

<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.

**3.8.2.2** `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.

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

Convert audio buffer with arbitrary number of channels to stereo.

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

## Parameters

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

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

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

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

## Parameters

<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.

**3.8.2.5** `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.

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

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

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

#### Parameters

<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.

## 3.9 BufferReaderPushAdapter< T > Class Template Reference

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

Inherits [BufferReaderPushAdapterBase< T >](#).

### Public Member Functions

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

### Protected Attributes

- T[] **buffer**

### 3.9.1 Detailed Description

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

### 3.9.2 Constructor & Destructor Documentation

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

Create a new [BufferReaderPushAdapter](#) instance

#### Parameters

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

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

### 3.9.3 Member Function Documentation

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

Do the actual data read/push.

Parameters

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

Implements [BufferReaderPushAdapterBase< T >](#).

## 3.10 BufferReaderPushAdapterAsyncPool< T > Class Template Reference

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

Inherits [BufferReaderPushAdapterBase< T >](#).

### Public Member Functions

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

### Additional Inherited Members

#### 3.10.1 Detailed Description

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

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

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

#### 3.10.2 Constructor & Destructor Documentation

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

Create a new [BufferReaderPushAdapter](#) instance

Parameters

<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 BufferReaderPushAdapterAsyncPoolCopy< T > Class Template Reference

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

Inherits [BufferReaderPushAdapterBase< T >](#).

### Public Member Functions

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

### Protected Attributes

- **T[] buffer**

#### 3.11.1 Detailed Description

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

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

#### 3.11.2 Constructor & Destructor Documentation

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

Create a new [BufferReaderPushAdapter](#) instance

## Parameters

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

#### 3.11.3 Member Function Documentation

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

Do the actual data read/push.

## Parameters

<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 >](#).

## 3.12 BufferReaderPushAdapterAsyncPoolFloatToShort Class Reference

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

Inherits [BufferReaderPushAdapterBase< float >](#).

### Public Member Functions

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

### Additional Inherited Members

#### 3.12.1 Detailed Description

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

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

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

#### 3.12.2 Constructor & Destructor Documentation

##### 3.12.2.1 [BufferReaderPushAdapterAsyncPoolFloatToShort](#) ( [LocalVoice](#) localVoice, [IDataReader< float >](#) reader )

Create a new [BufferReaderPushAdapter](#) instance

Parameters

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

#### 3.12.3 Member Function Documentation

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

Do the actual data read/push.

Parameters

<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< float >](#).

## 3.13 BufferReaderPushAdapterBase< T > Class Template Reference

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

Inherits [IServiceable](#).

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

## Public Member Functions

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

## Protected Attributes

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

### 3.13.1 Detailed Description

Adapter base class to move data by reading from [IDataReader.Read](#) and pushing to [LocalVoice](#).  
Use this with a [LocalVoice](#) of same T type.

### 3.13.2 Constructor & Destructor Documentation

#### 3.13.2.1 [BufferReaderPushAdapterBase](#) ( [IDataReader](#)< T > *reader* )

Create a new [BufferReaderPushAdapterBase](#) instance

Parameters

<i>reader</i>	<a href="#">IDataReader</a> to read from.
---------------	---

### 3.13.3 Member Function Documentation

#### 3.13.3.1 void [Dispose](#) ( )

Release resources associated with this instance.

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

Do the actual data read/push.

Parameters

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

Implements [IServiceable](#).

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

### 3.14 WebRTCAudioLib.ConfigParam Struct Reference

#### Public Attributes

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

### 3.15 ConnectAndJoin Class Reference

Inherits MonoBehaviour, IConnectionCallbacks, and IMatchmakingCallbacks.

#### Public Member Functions

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

#### Public Attributes

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

#### Properties

- bool **IsConnected** [get]

### 3.16 OpusCodec.Decoder< T > Class Template Reference

Inherits [IDecoder](#).

## Public Member Functions

- **Decoder** (Action< T[]> output, [ILogger](#) logger)
- void [Open](#) ([VoiceInfo](#) i)  
*Open (initialize) the decoder.*
- void **Dispose** ()
- void [Input](#) (byte[] buf)  
*Consumes the given encoded data.*

## Protected Member Functions

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

## Protected Attributes

- [OpusDecoder](#) **decoder**

## Properties

- string **Error** [get]

### 3.16.1 Member Function Documentation

#### 3.16.1.1 void Input ( byte[] buf )

Consumes the given encoded data.

Implements [IDecoder](#).

#### 3.16.1.2 void Open ( VoiceInfo info )

Open (initialize) the decoder.

Parameters

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

Implements [IDecoder](#).

## 3.17 RawCodec.Decoder< T > Class Template Reference

Inherits [IDecoder](#).

## Public Member Functions

- **Decoder** (Action< T[]> output)
- void [Open](#) ([VoiceInfo](#) info)  
*Open (initialize) the decoder.*
- void [Input](#) (byte[] buf)  
*Consumes the given encoded data.*
- void **Dispose** ()

## Properties

- string **Error** [get]

### 3.17.1 Member Function Documentation

#### 3.17.1.1 void Input ( byte[] buf )

Consumes the given encoded data.

Implements [IDecoder](#).

#### 3.17.1.2 void Open ( VoiceInfo info )

Open (initialize) the decoder.

Parameters

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

Implements [IDecoder](#).

## 3.18 OpusCodec.DecoderFactory Class Reference

### Static Public Member Functions

- static [IDecoder](#) **Create**< T > ([VoiceInfo](#) i, [ILogger](#) logger)

## 3.19 OpusCodec.DecoderFloat Class Reference

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

### Public Member Functions

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

### Protected Member Functions

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

### Additional Inherited Members

## 3.20 OpusCodec.DecoderShort Class Reference

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

### Public Member Functions

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



### Protected Member Functions

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

### Additional Inherited Members

## 3.21 OpusCodec.Encoder< T > Class Template Reference

Inherits [IEncoderDirect< T\[\]>](#).

### Public Member Functions

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

### Protected Member Functions

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

### Protected Attributes

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

### Properties

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

## 3.22 RawCodec.Encoder< T > Class Template Reference

Inherits [IEncoderDirect< T\[\]>](#).

### Public Member Functions

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

### Properties

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

## 3.23 OpusCodec.EncoderFloat Class Reference

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

### Protected Member Functions

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

### Additional Inherited Members

## 3.24 OpusCodec.EncoderShort Class Reference

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

### Protected Member Functions

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

### Additional Inherited Members

## 3.25 OpusCodec.Factory Class Reference

### Static Public Member Functions

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

## 3.26 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]

### 3.26.1 Detailed Description

`PrimitiveArrayPool<T>` as wrapped in object factory interface.

## Template Parameters

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

## 3.27 FactoryReusableArray< T > Class Template Reference

Array factory returnng 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]

### 3.27.1 Detailed Description

Array factory returnng 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.
----------	---------------------

## 3.28 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.*

### 3.28.1 Detailed Description

Utility class to re-frame audio packets.

### 3.28.2 Constructor & Destructor Documentation

#### 3.28.2.1 Framer ( int *frameSize* )

Create new [Framer](#) instance.

### 3.28.3 Member Function Documentation

#### 3.28.3.1 int Count ( int *bufLen* )

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

Parameters

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

Returns

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

#### 3.28.3.2 IEnumerable<T[]> Frame ( T[] *buf* )

Append arbitrary-sized buffer and return available full frames.

Parameters

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

Returns

Enumerator of full frames (might be none).

## 3.29 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.*

#### 3.29.1 Detailed Description

Audio Source interface.

### 3.29.2 Property Documentation

#### 3.29.2.1 `int Channels` [get]

Number of channels in the audio signal.

#### 3.29.2.2 `string Error` [get]

If not null, audio object is in invalid state.

#### 3.29.2.3 `int SamplingRate` [get]

Sampling rate of the audio signal (in Hz).

## 3.30 `IAudioOut< T >` Interface Template Reference

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

### Public Member Functions

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

### Properties

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

## 3.31 `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

#### 3.31.1 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,

### 3.31.2 Member Function Documentation

#### 3.31.2.1 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 >](#).

## 3.32 IAudioReader< T > Interface Template Reference

Audio Reader interface.

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

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

### Additional Inherited Members

#### 3.32.1 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).

## 3.33 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.*

#### 3.33.1 Detailed Description

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

### 3.33.2 Member Function Documentation

#### 3.33.2.1 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 >](#).

## 3.34 IDecoder Interface Reference

Generic decoder interface.

Inherits [IDisposable](#).

Inherited by [IDecoderQueuedOutputImageNative](#), [OpusCodec.Decoder< T >](#), and [RawCodec.Decoder< T >](#).

### Public Member Functions

- void [Open](#) ([VoiceInfo](#) info)  
*Open (initialize) the decoder.*
- void [Input](#) (byte[] buf)  
*Consumes the given encoded data.*

### Properties

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

#### 3.34.1 Detailed Description

Generic decoder interface.

#### 3.34.2 Member Function Documentation

##### 3.34.2.1 void Input ( byte[] buf )

Consumes the given encoded data.

Implemented in [OpusCodec.Decoder< T >](#), and [RawCodec.Decoder< T >](#).

##### 3.34.2.2 void Open ( VoiceInfo info )

Open (initialize) the decoder.

## Parameters

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

Implemented in [OpusCodec.Decoder< T >](#), and [RawCodec.Decoder< T >](#).

### 3.34.3 Property Documentation

#### 3.34.3.1 string Error [get]

If not null, the object is in invalid state.

## 3.35 IDecoderQueuedOutputImageNative Interface Reference

Inherits [IDecoder](#).

### Properties

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

### Additional Inherited Members

## 3.36 IEncoder Interface Reference

Generic encoder interface.

Inherits [IDisposable](#).

Inherited by [IEncoderDirect< B >](#).

### Public Member Functions

- ArraySegment< byte > [DequeueOutput](#) ()  
*Returns next encoded data frame (if such output supported).*

### Properties

- string [Error](#) [get]  
*If not null, the object is in invalid state.*
- Action< ArraySegment< byte > > [Output](#) [get, set]  
*Set callback encoder calls on each encoded data frame (if such output supported).*

#### 3.36.1 Detailed Description

Generic encoder interface.

Depending on implementation, encoder should either call Output on each data frame or return next data frame in [DequeueOutput\(\)](#) call.

#### 3.36.2 Member Function Documentation

##### 3.36.2.1 ArraySegment<byte> DequeueOutput ( )

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



### 3.36.3 Property Documentation

#### 3.36.3.1 string Error [get]

If not null, the object is in invalid state.

#### 3.36.3.2 Action<ArraySegment<byte> > Output [get], [set]

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

## 3.37 IEncoderDirect< B > Interface Template Reference

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

Inherits [IEncoder](#).

### Public Member Functions

- void [Input](#) (B buf)  
*Consumes the given raw data.*

### Additional Inherited Members

#### 3.37.1 Detailed Description

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

#### 3.37.2 Member Function Documentation

##### 3.37.2.1 void Input ( B buf )

Consumes the given raw data.

##### Parameters

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

## 3.38 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.*

### 3.38.1 Detailed Description

Audio Level Metering interface.

### 3.38.2 Member Function Documentation

#### 3.38.2.1 void ResetAccumAvgPeakAmp ( )

Reset [AccumAvgPeakAmp](#).

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

### 3.38.3 Property Documentation

#### 3.38.3.1 float AccumAvgPeakAmp [get]

Average of CurrentPeakAmps since last reset.

#### 3.38.3.2 float CurrentAvgAmp [get]

Average amplitude value over last half second.

#### 3.38.3.3 float CurrentPeakAmp [get]

Maximum amplitude value over last half second sec.

## 3.39 ILocalVoiceAudio Interface Reference

Interface for an outgoing audio stream.

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

## Public Member Functions

- void [VoiceDetectorCalibrate](#) (int durationMs, Action< float > onCalibrated=null)  
*Trigger voice detector calibration process.*

## Properties

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

### 3.39.1 Detailed Description

Interface for an outgoing audio stream.

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

### 3.39.2 Member Function Documentation

3.39.2.1 void [VoiceDetectorCalibrate](#) ( int *durationMs*, Action< float > *onCalibrated* = null )

Trigger voice detector calibration process.

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

Parameters

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

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

### 3.39.3 Property Documentation

3.39.3.1 [AudioUtil.ILevelMeter LevelMeter](#) [get]

The LevelMeter utility in use.

3.39.3.2 [AudioUtil.IVoiceDetector VoiceDetector](#) [get]

The VoiceDetector in use.

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

3.39.3.3 bool [VoiceDetectorCalibrating](#) [get]

If true, voice detector calibration is in progress.

## 3.40 ILoggable Interface Reference

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

## Properties

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

### 3.41 ILogger Interface Reference

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

#### Public Member Functions

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

### 3.42 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]

### 3.43 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]

### 3.44 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

## 3.45 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

## 3.46 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

## 3.47 ImageInputBuf Struct Reference

### Public Attributes

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

### 3.48 ImageOutputBuf Struct Reference

#### Public Attributes

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

### 3.49 IOSAudioForceToSpeaker Class Reference

Inherits MonoBehaviour.

### 3.50 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.*

#### 3.50.1 Detailed Description

Audio Processor interface.

#### 3.50.2 Member Function Documentation

##### 3.50.2.1 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 >](#).

### 3.51 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.*

### 3.51.1 Detailed Description

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

### 3.51.2 Member Function Documentation

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

Service function that should be called regularly.

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

## 3.52 ISyncAudioOut< T > Interface Template Reference

Inherits [IAudioOut< T >](#).

## Public Member Functions

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

## Properties

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

## 3.53 AudioUtil.VoiceDetector 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.*

### 3.53.1 Detailed Description

[Voice](#) Activity Detector interface.

### 3.53.2 Property Documentation

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

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

#### 3.53.2.2 `bool Detected` `[get]`

If true, voice detected.

#### 3.53.2.3 `DateTime DetectedTime` `[get]`

Last time when switched to detected state.

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

If true, voice detection enabled.

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

[Voice](#) detected as soon as signal level exceeds threshold.

### 3.53.3 Event Documentation

#### 3.53.3.1 Action `OnDetected`

Called when switched to detected state.

## 3.54 IVoiceTransport Interface Reference

Inherits [ILogger](#).

Inherited by [LoadBalancingTransport](#).

## Public Member Functions

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



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

## 3.55 AudioUtil.LevelMeter< T > Class Template Reference

Audio Level Meter.

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

### Public Member Functions

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

### Protected Attributes

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

### Properties

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

#### 3.55.1 Detailed Description

Audio Level Meter.

#### 3.55.2 Member Function Documentation

##### 3.55.2.1 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 >](#).

**3.55.2.2 void ResetAccumAvgPeakAmp ( )**

Reset AccumAvgPeakAmp.

Implements [AudioUtil.ILevelMeter](#).

**3.56 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]

**3.56.1 Detailed Description**

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

**3.56.2 Member Function Documentation****3.56.2.1 void ResetAccumAvgPeakAmp ( )**

Reset AccumAvgPeakAmp.

Implements [AudioUtil.ILevelMeter](#).

**3.57 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

### 3.57.1 Detailed Description

[LevelMeter](#) specialization for float audio.

### 3.57.2 Constructor & Destructor Documentation

#### 3.57.2.1 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.

## 3.58 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

### 3.58.1 Detailed Description

[LevelMeter](#) specialization for short audio.

### 3.58.2 Constructor & Destructor Documentation

#### 3.58.2.1 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.

### 3.59 LoadBalancingFrontend Class Reference

Inherits [LoadBalancingTransport](#).

#### Additional Inherited Members

### 3.60 LoadBalancingTransport Class Reference

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

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

Inherited by [LoadBalancingFrontend](#).

#### Public Member Functions

- void **LogError** (string fmt, params object[] args)
- void **LogWarning** (string fmt, params object[] args)
- void **LogInfo** (string fmt, params object[] args)
- void **LogDebug** (string fmt, params object[] args)
- bool **IsChannelJoined** (int channelId)
- void **SetDebugEchoMode** ([LocalVoice](#) v)
- [LoadBalancingTransport](#) (ConnectionProtocol connectionProtocol=ConnectionProtocol.Udp)  
*Initializes a new [LoadBalancingTransport](#).*
- new void [Service](#) ()  
*This method dispatches all available incoming commands and then sends this client's outgoing commands. Call this method regularly (2 to 20 times a second).*
- virtual bool **ChangeAudioGroups** (byte[] groupsToRemove, byte[] groupsToAdd)
- void **SendVoicesInfo** (IEnumerable< [LocalVoice](#) > voices, int channelId, int targetPlayerId)
- void [SendDebugEchoVoicesInfo](#) (int channelId)  
*Send VoicesInfo events to the local player for all voices that have DebugEcho enabled.*
- void **SendVoiceRemove** ([LocalVoice](#) voice, int channelId, int targetPlayerId)
- void **SendFrame** (ArraySegment< byte > data, byte evNumber, byte voiceld, int channelId, [LocalVoice](#) localVoice)
- string **ChannelIdStr** (int channelId)
- string **PlayerIdStr** (int playerId)
- void [Dispose](#) ()  
*Releases all resources used by the [LoadBalancingTransport](#) instance.*

#### Protected Attributes

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

## Properties

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

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

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

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

### 3.60.1 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 .

### 3.60.2 Constructor & Destructor Documentation

#### 3.60.2.1 [LoadBalancingTransport](#) ( [ConnectionProtocol](#) *connectionProtocol* = [ConnectionProtocol.Udp](#) )

Initializes a new [LoadBalancingTransport](#).

Parameters

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

### 3.60.3 Member Function Documentation

#### 3.60.3.1 void [Dispose](#) ( )

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

#### 3.60.3.2 void [SendDebugEchoVoicesInfo](#) ( int *channelId* )

Send [VoicesInfo](#) events to the local player for all voices that have [DebugEcho](#) enabled.

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

Parameters

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

#### 3.60.3.3 new void [Service](#) ( )

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

### 3.60.4 Property Documentation

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

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

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

#### 3.60.4.2 `VoiceClient VoiceClient` [get]

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

## 3.61 LocalVoice Class Reference

Represents outgoing data stream.

Inherits `IDisposable`.

Inherited by [LocalVoiceAudioDummy](#), and [LocalVoiceFramedBase](#).

### Public Member Functions

- virtual [IEncoder](#) **CreateDefaultEncoder** ([VoiceInfo](#) info)
- void [RemoveSelf](#) ()  
*Remove this voice from it's [VoiceClient](#) (using [VoiceClient.RemoveLocalVoice](#))*
- virtual void **Dispose** ()

### Public Attributes

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

### Protected Member Functions

- void **resetNoTransmitCnt** ()

### Protected Attributes

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

### Properties

- byte **Group** [get, set]
- byte [InterestGroup](#) [get, set]  
*If InterestGroup != 0, voice's data is sent only to clients listening to this group (if supported by transport).*
- [VoiceInfo](#) **Info** [get]  
*Returns Info structure assigned on local voice cration.*
- bool [TransmitEnabled](#) [get, set]

*If true, stream data broadcasted.*

- bool [IsCurrentlyTransmitting](#) [get, protected set]

*Returns true if stream broadcasts.*

- int [FramesSent](#) [get]

*Sent frames counter.*

- int [FramesSentBytes](#) [get]

*Sent frames bytes counter.*

- bool [Reliable](#) [get, set]

*Send data reliable.*

- bool [Encrypt](#) [get, set]

*Send data encrypted.*

- [IServiceable LocalUserServiceable](#) [get, set]

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

- bool [DebugEchoMode](#) [get, set]

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

### 3.61.1 Detailed Description

Represents outgoing data stream.

### 3.61.2 Member Function Documentation

#### 3.61.2.1 void RemoveSelf ( )

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

.

### 3.61.3 Property Documentation

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

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

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

Send data encrypted.

#### 3.61.3.3 int FramesSent [get]

Sent frames counter.

#### 3.61.3.4 int FramesSentBytes [get]

Sent frames bytes counter.

### 3.61.3.5 **VoiceInfo Info** [get]

Returns Info structure assigned on local voice cration.

### 3.61.3.6 **byte InterestGroup** [get], [set]

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

### 3.61.3.7 **bool IsCurrentlyTransmitting** [get], [protected set]

Returns true if stream broadcasts.

### 3.61.3.8 **IServiceable LocalUserServiceable** [get], [set]

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

### 3.61.3.9 **bool Reliable** [get], [set]

Send data reliable.

### 3.61.3.10 **bool TransmitEnabled** [get], [set]

If true, stream data broadcasted.

## 3.62 **LocalVoiceAudio< T > Class Template Reference**

Outgoing audio stream.

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

### Public Member Functions

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

*Trigger voice detector calibration process.*

### Static Public Member Functions

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

*Create a new LocalVoiceAudio< T> instance.*

### Protected Member Functions

- void **initBuiltinProcessors** ()



## Protected Attributes

- [AudioUtil.VoiceDetector](#)< T > **voiceDetector**
- [AudioUtil.VoiceDetectorCalibration](#)< T > **voiceDetectorCalibration**
- [AudioUtil.LevelMeter](#)< T > **levelMeter**
- int **channels**
- bool **resampleSource**

## Properties

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

*True if the VoiceDetector is currently calibrating.*

## Additional Inherited Members

### 3.62.1 Detailed Description

Outgoing audio stream.

### 3.62.2 Member Function Documentation

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

Create a new [LocalVoiceAudio](#)<T> instance.

#### Parameters

<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.

- 3.62.2.2 void [VoiceDetectorCalibrate](#) ( int *durationMs*, [Action](#)< float > *onCalibrated* = null )

Trigger voice detector calibration process.

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

#### Parameters

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

Implements [ILocalVoiceAudio](#).

### 3.62.3 Property Documentation

#### 3.62.3.1 bool VoiceDetectorCalibrating [get]

True if the VoiceDetector is currently calibrating.

## 3.63 LocalVoiceAudioDummy Class Reference

Dummy [LocalVoiceAudio](#)

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

### Public Member Functions

- void [VoiceDetectorCalibrate](#) (int durationMs, Action< float > onCalibrated=null)  
*Trigger voice detector calibration process.*

### Static Public Attributes

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

### Properties

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

### Additional Inherited Members

#### 3.63.1 Detailed Description

Dummy [LocalVoiceAudio](#)

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

#### 3.63.2 Member Function Documentation

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

Trigger voice detector calibration process.

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

Parameters

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

Implements [ILocalVoiceAudio](#).

### 3.63.3 Member Data Documentation

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

A Dummy [LocalVoiceAudio](#) instance.

## 3.64 LocalVoiceAudioFloat Class Reference

Specialization of [LocalVoiceAudio](#) for float audio

Inherits [LocalVoiceAudio< float >](#).

### Additional Inherited Members

#### 3.64.1 Detailed Description

Specialization of [LocalVoiceAudio](#) for float audio

## 3.65 LocalVoiceAudioShort Class Reference

Specialization of [LocalVoiceAudio](#) for short audio

Inherits [LocalVoiceAudio< short >](#).

### Additional Inherited Members

#### 3.65.1 Detailed Description

Specialization of [LocalVoiceAudio](#) for short audio

## 3.66 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

### 3.66.1 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 transport.
<i>encoder</i>	Encoder producing the stream.

#### Returns

Outgoing stream handler.

### 3.66.2 Member Function Documentation

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

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

#### Parameters

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

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

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

#### Parameters

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

#### 3.66.2.3 void ClearProcessors ( )

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

#### 3.66.2.4 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](#).

#### 3.66.2.5 void PushData ( T[] buf )

Synchronously push data into this stream.

#### 3.66.2.6 void PushDataAsync ( T[] buf )

Asynchronously push data into this stream.

### 3.66.3 Property Documentation

#### 3.66.3.1 bool PushDataAsyncReady [get]

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

## 3.67 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

#### 3.67.1 Detailed Description

Typed re-framing [LocalVoice](#)

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

#### 3.67.2 Property Documentation

##### 3.67.2.1 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.

### 3.68 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)

### 3.69 MicAmplifier Class Reference

Inherits [VoiceComponent](#).

#### Properties

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

#### Additional Inherited Members

### 3.70 MicAmplifierFloat Class Reference

Inherits [IProcessor< float >](#).

#### Public Member Functions

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

#### Properties

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

### 3.71 MicAmplifierShort Class Reference

Inherits [IProcessor< short >](#).

#### Public Member Functions

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

## Properties

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

## 3.72 MicWrapper Class Reference

Inherits [IAudioReader< float >](#).

## Public Member Functions

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

## Properties

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

## 3.73 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]

### 3.73.1 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).

### 3.74 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.*

##### 3.74.1 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).

## 3.74.2 Constructor &amp; Destructor Documentation

3.74.2.1 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.

3.74.2.2 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.

## 3.74.3 Member Function Documentation

## 3.74.3.1 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.

3.74.3.2 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.
-------------	---

## 3.74.3.3 void Dispose ( )

Free resources associated with this [ObjectPool](#)

3.74.3.4 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.
-------------	---------------------------------

**3.74.3.5 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.

**3.74.3.6 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.
------------	-----------------------------------

**3.74.4 Property Documentation****3.74.4.1 TInfo Info [get]**

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

**3.75 OpusCodec Class Reference****Classes**

- class [Decoder](#)
- class [DecoderFactory](#)
- class [DecoderFloat](#)
- class [DecoderShort](#)
- class [Encoder](#)
- class [EncoderFloat](#)
- class [EncoderShort](#)
- class [Factory](#)
- class [Util](#)

**Public Types**

- enum **FrameDuration**

**3.76 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]

## 3.77 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]

### 3.77.1 Property Documentation

#### 3.77.1.1 [Delay](#) EncoderDelay [get], [set]

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

### 3.78 OpusException Class Reference

Inherits [Exception](#).

#### Public Member Functions

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

#### Properties

- OpusStatusCode **StatusCodes** [get]

### 3.79 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\_LIKELIHOOD** = 63

### 3.80 Recorder.PhotonVoiceCreatedParams Class Reference

Inherits [PhotonVoiceCreatedParams](#).

#### Additional Inherited Members

### 3.81 PhotonVoiceCreatedParams Class Reference

Inherited by [Recorder.PhotonVoiceCreatedParams](#).

#### Properties

- [Voice.LocalVoice](#) **Voice** [get, set]
- [Voice.IAudioDesc](#) **AudioDesc** [get, set]

## 3.82 PhotonVoiceLagSimulationGui Class Reference

Inherits MonoBehaviour.

### Public Member Functions

- void **OnEnable** ()

## 3.83 PhotonVoiceNetwork Class Reference

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

Inherits [VoiceConnection](#).

### Public Member Functions

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

### Public Attributes

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

### Protected Member Functions

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

### Properties

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

## Additional Inherited Members

### 3.83.1 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.

### 3.83.2 Member Function Documentation

#### 3.83.2.1 `bool ConnectAndJoinRoom ( )`

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

##### Returns

If true, connection command send from client

#### 3.83.2.2 `void Disconnect ( )`

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

### 3.83.3 Member Data Documentation

#### 3.83.3.1 `bool AutoConnectAndJoin = true`

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

#### 3.83.3.2 `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

#### 3.83.3.3 `bool AutoLeaveAndDisconnect = true`

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

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

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

### 3.83.4 Property Documentation

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

Singleton instance for [PhotonVoiceNetwork](#)

## 3.84 PhotonVoiceStatsGui Class Reference

Basic GUI to show traffic and health statistics of the connection to [Photon](#), toggled by shift+tab.

Inherits [MonoBehaviour](#).

### 3.84.1 Detailed Description

Basic GUI to show traffic and health statistics of the connection to [Photon](#), toggled by shift+tab.

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

## 3.85 PhotonVoiceView Class Reference

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

Inherits [VoiceComponent](#).

### Public Member Functions

- void [Init](#) ()  
*Initializes this [PhotonVoiceView](#) for [Voice](#) usage based on the [PhotonView](#), [Recorder](#) and [Speaker](#) components.*

### Public Attributes

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

### Protected Member Functions

- override void [Awake](#) ()

### Properties

- [Recorder](#) [RecorderInUse](#) [get, set]  
*The [Recorder](#) component currently used by this [PhotonVoiceView](#)*
- [Speaker](#) [SpeakerInUse](#) [get, set]  
*The [Speaker](#) component currently used by this [PhotonVoiceView](#)*
- bool [IsSetup](#) [get, 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

### 3.85.1 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.

### 3.85.2 Member Function Documentation

#### 3.85.2.1 `void Init ( )`

Initializes this [PhotonVoiceView](#) for [Voice](#) usage based on the [PhotonView](#), [Recorder](#) and [Speaker](#) components.

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

### 3.85.3 Member Data Documentation

#### 3.85.3.1 `bool AutoCreateRecorderIfNotFound`

If true, a [Recorder](#) component will be added to the same [GameObject](#) if not found already.

#### 3.85.3.2 `bool SetupDebugSpeaker`

If true, a [Speaker](#) component will be setup to be used for the [DebugEcho](#) mode

#### 3.85.3.3 `bool UsePrimaryRecorder`

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

### 3.85.4 Property Documentation

#### 3.85.4.1 `bool IsRecorder` [get], [protected set]

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

#### 3.85.4.2 `bool IsRecording` [get]

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

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

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



**3.85.4.4 bool IsSpeaker** [get], [protected set]

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

**3.85.4.5 bool IsSpeaking** [get]

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

**3.85.4.6 Recorder RecorderInUse** [get], [set]

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

**3.85.4.7 Speaker SpeakerInUse** [get], [set]

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

**3.86 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****3.86.1 Detailed Description**

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

**Template Parameters**

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

**3.87 RawCodec Class Reference****Classes**

- class [Decoder](#)
- class [Encoder](#)

## 3.88 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 **Relnit** ()
- void [RestartRecording](#) ()  
*Restarts recording if something has changed that requires this.*
- void [VoiceDetectorCalibrate](#) (int durationMs, Action< float > detectionEndedCallback=null)  
*Trigger voice detector calibration process. While calibrating, keep silence. [Voice](#) detector sets threshold basing on measured background noise level.*
- void [StartRecording](#) ()  
*Starts recording.*
- void [StopRecording](#) ()  
*Stops recording.*

### Protected Member Functions

- virtual void **SendPhotonVoiceCreatedMessage** ()

### Properties

- static [AudioInEnumerator](#) [PhotonMicrophoneEnumerator](#) [get]  
*Enumerator for the available microphone devices gathered by the [Photon](#) plugin.*
- bool [IsInitialized](#) [get]  
*If true, this [Recorder](#) has been initialized and is ready to transmit to remote clients. Otherwise call [Init](#).*
- bool **RequiresInit** [get]
- bool [RequiresRestart](#) [get, protected set]  
*Returns true if something has changed in the [Recorder](#) while recording that won't take effect unless recording is restarted using [RestartRecording](#).*
- bool [TransmitEnabled](#) [get, set]  
*If true, audio transmission is enabled.*
- bool [Encrypt](#) [get, set]  
*If true, voice stream is sent encrypted.*
- bool [DebugEchoMode](#) [get, set]  
*If true, outgoing stream routed back to client via server same way as for remote client's streams.*

- bool [ReliableMode](#) [get, set]  
*If true, stream data sent in reliable mode.*
- bool [VoiceDetection](#) [get, set]  
*If true, voice detection enabled.*
- float [VoiceDetectionThreshold](#) [get, set]  
*Voice detection threshold (0..1, where 1 is full amplitude).*
- int [VoiceDetectionDelayMs](#) [get, set]  
*Keep detected state during this time after signal level dropped below threshold. Default is 500ms*
- object [UserData](#) [get, set]  
*Custom user object to be sent in the voice stream info event.*
- Func< [IAudioDesc](#) > [InputFactory](#) [get, set]  
*Set the method returning new [Voice.IAudioDesc](#) instance to be assigned to a new voice created with Source set to Factory*
- [AudioUtil.IVoiceDetector](#) [VoiceDetector](#) [get]  
*Returns voice activity detector for recorder's audio stream.*
- string [UnityMicrophoneDevice](#) [get, set]  
*Set or get [Unity](#) microphone device used for streaming.*
- int [PhotonMicrophoneDeviceId](#) [get, set]  
*Set or get photon microphone device used for streaming.*
- byte [AudioGroup](#) [get, set]  
*Target interest group that will receive transmitted audio.*
- byte [InterestGroup](#) [get, set]  
*Target interest group that will receive transmitted audio.*
- bool [IsCurrentlyTransmitting](#) [get]  
*Returns true if audio stream broadcasts.*
- [AudioUtil.ILevelMeter](#) [LevelMeter](#) [get]  
*Level meter utility.*
- bool [VoiceDetectorCalibrating](#) [get]  
*If true, voice detector calibration is in progress.*
- [ILocalVoiceAudio](#) [voiceAudio](#) [get]
- InputSourceType [SourceType](#) [get, set]  
*Audio data source.*
- MicType [MicrophoneType](#) [get, set]  
*Which microphone API to use when the Source is set to Microphone.*
- SampleTypeConv [TypeConvert](#) [get, set]  
*Force creation of 'short' pipeline and convert audio data to short for 'float' audio sources.*
- AudioClip [AudioClip](#) [get, set]  
*Source audio clip.*
- bool [LoopAudioClip](#) [get, set]  
*Loop playback for audio clip sources.*
- POpusCodec.Enums.SamplingRate [SamplingRate](#) [get, set]  
*Outgoing audio stream sampling rate.*
- OpusCodec.FrameDuration [FrameDuration](#) [get, set]  
*Outgoing audio stream encoder delay.*
- int [Bitrate](#) [get, set]  
*Outgoing audio stream bitrate.*
- bool [IsRecording](#) [get, set]  
*Gets or sets whether this [Recorder](#) is actively recording audio to be transmitted.*
- bool [ReactOnSystemChanges](#) [get, set]
- bool [AutoStart](#) [get, set]  
*If true, automatically start recording when initialized.*

## Additional Inherited Members

### 3.88.1 Detailed Description

Component representing outgoing audio stream in scene.

### 3.88.2 Member Function Documentation

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

Initializes the Recorder component to be able to transmit audio.

Parameters

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

#### 3.88.2.2 void RestartRecording ( )

Restarts recording if something has changed that requires this.

#### 3.88.2.3 void StartRecording ( )

Starts recording.

#### 3.88.2.4 void StopRecording ( )

Stops recording.

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

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

Parameters

<i>durationMs</i>	Duration of calibration in milliseconds.
<i>detection↔ EndedCallback</i>	Callback when VAD calibration ends.

### 3.88.3 Property Documentation

#### 3.88.3.1 AudioClip AudioClip [get], [set]

Source audio clip.

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

Target interest group that will receive transmitted audio.

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

**3.88.3.3 bool AutoStart** [get], [set]

If true, automatically start recording when initialized.

**3.88.3.4 int Bitrate** [get], [set]

Outgoing audio stream bitrate.

**3.88.3.5 bool DebugEchoMode** [get], [set]

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

**3.88.3.6 bool Encrypt** [get], [set]

If true, voice stream is sent encrypted.

**3.88.3.7 OpusCodec.FrameDuration FrameDuration** [get], [set]

Outgoing audio stream encoder delay.

**3.88.3.8 Func<IAudioDesc> InputFactory** [get], [set]

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

**3.88.3.9 byte InterestGroup** [get], [set]

Target interest group that will receive transmitted audio.

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

**3.88.3.10 bool IsCurrentlyTransmitting** [get]

Returns true if audio stream broadcasts.

**3.88.3.11 bool IsInitialized** [get]

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

**3.88.3.12 bool IsRecording** [get], [set]

Gets or sets whether this [Recorder](#) is actively recording audio to be transmitted.

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

Level meter utility.

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

Loop playback for audio clip sources.

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

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

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

Set or get photon microphone device used for streaming.

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

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

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

If true, stream data sent in reliable mode.

**3.88.3.19 bool RequiresRestart** [get], [protected set]

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

Think of this as a "isDirty" flag.

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

Outgoing audio stream sampling rate.

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

Audio data source.

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

If true, audio transmission is enabled.

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

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

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

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

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

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

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

If true, voice detection enabled.

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

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

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

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

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

Returns voice activity detector for recorder's audio stream.

3.88.3.30 `bool VoiceDetectorCalibrating` `[get]`

If true, voice detector calibration is in progress.

## 3.89 RemoteVoiceInfo Class Reference

Information about a remote voice (incoming stream).

### Properties

- [VoiceInfo Info](#) `[get]`  
*Remote voice info.*
- `int ChannelId` `[get]`  
*ID of channel used for transmission.*
- `int PlayerId` `[get]`  
*Player ID of voice owner.*
- `byte VoiceId` `[get]`  
*[Voice](#) ID (unique in the room).*

### 3.89.1 Detailed Description

Information about a remote voice (incoming stream).

### 3.89.2 Property Documentation

3.89.2.1 `int ChannelId` `[get]`

ID of channel used for transmission.

3.89.2.2 `VoiceInfo Info` `[get]`

Remote voice info.

3.89.2.3 `int PlayerId` `[get]`

Player ID of voice owner.

### 3.89.2.4 byte Voiceld [get]

Voice ID (unique in the room).

## 3.90 RemoteVoiceLink Class Reference

### Public Member Functions

- **RemoteVoiceLink** ([VoiceInfo](#) info, int playerId, int voiceld, int channelId, ref [RemoteVoiceOptions](#) options)

### Properties

- [VoiceInfo](#) **Info** [get]
- int **PlayerId** [get]
- int **Voiceld** [get]
- int **ChannelId** [get]

### Events

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

## 3.91 RemoteVoiceOptions Struct Reference

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

### Public Member Functions

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

### Properties

- Action [OnRemoteVoiceRemoveAction](#) [get, set]  
*Register a method to be called when the remote voice is removed.*
- [IDecoder](#) **Decoder** [get, set]  
*Remote voice data decoder. Use to set decoder options or override it with user decoder.*
- ImageFormat **OutputImageFormat** [get, set]
- Flip **OutputImageFlip** [get, set]

### 3.91.1 Detailed Description

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



### 3.91.2 Member Function Documentation

#### 3.91.2.1 void SetOutput ( Action< float[]> output )

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

### 3.91.3 Property Documentation

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

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

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

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

## 3.92 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**

### 3.92.1 Detailed Description

Sample-rate conversion Audio Processor.

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

### 3.92.2 Constructor & Destructor Documentation

#### 3.92.2.1 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.

### 3.92.3 Member Function Documentation

#### 3.92.3.1 `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 >](#).

## 3.93 Speaker Class Reference

Component representing remote audio stream in local scene.

Inherits [VoiceComponent](#).

### Public Attributes

- int **PlayDelayMs** = 200

### Protected Member Functions

- override void **Awake** ()

### Properties

- bool [IsPlaying](#) [get]  
*Is the speaker playing right now.*
- int [Lag](#) [get]  
*Smoothed difference between (jittering) stream and (clock-driven) audioOutput.*
- Action< [Speaker](#) > [OnRemoteVoiceRemoveAction](#) [get, set]  
*Register a method to be called when remote voice removed.*
- Realtime.Player [Actor](#) [get, set]  
*Per room, the connected users/players are represented with a Realtime.Player, also known as Actor.*
- bool [IsLinked](#) [get]  
*Whether or not this [Speaker](#) has been linked to a remote voice stream.*

### Additional Inherited Members

#### 3.93.1 Detailed Description

Component representing remote audio stream in local scene.

### 3.93.2 Property Documentation

#### 3.93.2.1 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.

#### 3.93.2.2 bool IsLinked [get]

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

#### 3.93.2.3 bool IsPlaying [get]

Is the speaker playing right now.

#### 3.93.2.4 int Lag [get]

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

#### 3.93.2.5 Action<Speaker> OnRemoteVoiceRemoveAction [get], [set]

Register a method to be called when remote voice removed.

## 3.94 TestTone Class Reference

Inherits MonoBehaviour.

## 3.95 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]

### 3.95.1 Detailed Description

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

### 3.95.2 Constructor & Destructor Documentation

3.95.2.1 **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.

### 3.95.3 Member Function Documentation

3.95.3.1 **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 >](#).

## 3.96 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]

## 3.97 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.

### 3.97.1 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.

### 3.97.2 Constructor & Destructor Documentation

**3.97.2.1** [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.

### 3.97.3 Member Function Documentation

**3.97.3.1** 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 >](#).

### 3.97.4 Property Documentation

#### 3.97.4.1 `int Channels` [get]

Number of channels in the audio signal.

#### 3.97.4.2 `string Error` [get]

If not null, audio object is in invalid state.

#### 3.97.4.3 `int SamplingRate` [get]

Sampling rate of the audio signal (in Hz).

## 3.98 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]

## 3.99 UnityAudioOut Class Reference

Inherits [ISyncAudioOut< float >](#).

### Public Member Functions

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

### Public Attributes

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

## Properties

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

## 3.100 UnsupportedCodecException Class Reference

Exception thrown if an unsupported codec is encountered.

Inherits Exception.

## Public Member Functions

- [UnsupportedCodecException](#) (string info, [Codec](#) codec, [ILogger](#) logger)  
Create a new [UnsupportedCodecException](#).

### 3.100.1 Detailed Description

Exception thrown if an unsupported codec is encountered.

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

### 3.100.2 Constructor & Destructor Documentation

#### 3.100.2.1 UnsupportedCodecException ( string info, Codec codec, ILogger logger )

Create a new [UnsupportedCodecException](#).

Parameters

<i>info</i>	The info prepending standard message.
<i>codec</i>	The codec actually encountered.
<i>logger</i>	Logger.

## 3.101 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](#).

### 3.101.1 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.

### 3.101.2 Constructor & Destructor Documentation

#### 3.101.2.1 UnsupportedSampleTypeException ( Type *t* )

Create a new [UnsupportedSampleTypeException](#).

Parameters

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

## 3.102 OpusCodec.Util Class Reference

### 3.103 VoiceClient Class Reference

[Voice](#) client interact with other clients on network via [IVoiceTransport](#).

Inherits [IDisposable](#).

#### Public Member Functions

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

*Remote voice info event delegate.*

- [IEnumerable](#)< [LocalVoice](#) > [LocalVoicesInChannel](#) (int channelId)

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

- void [Service](#) ()

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

- [LocalVoice](#) [CreateLocalVoice](#) ([VoiceInfo](#) voiceInfo, int channelId=0, [IEncoder](#) encoder=null)

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

- [LocalVoiceFramed](#)< T > [CreateLocalVoiceFramed](#)< T > ([VoiceInfo](#) voiceInfo, int frameSize, int channelId=0, [IEncoder](#) encoder=null)

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

- [LocalVoiceAudio](#)< T > [CreateLocalVoiceAudio](#)< T > ([VoiceInfo](#) voiceInfo, [IAudioDesc](#) audioSourceDesc, int channelId=0, [IEncoder](#) encoder=null)

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

- [LocalVoice](#) [CreateLocalVoiceAudioFromSource](#) ([VoiceInfo](#) voiceInfo, [IAudioDesc](#) source, bool force↔ Short=false, int channelId=0, [IEncoder](#) encoder=null)

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

- void [RemoveLocalVoice](#) ([LocalVoice](#) voice)

*Removes local voice (outgoing data stream).*

Parameters

voice	Handler of outgoing stream to be removed.
-------	---

- void [Dispose](#) ()



## Properties

- int [FramesLost](#) [get, set]  
*Lost frames counter.*
- int [FramesReceived](#) [get]  
*Received frames counter.*
- int [FramesSent](#) [get]  
*Sent frames counter.*
- int [FramesSentBytes](#) [get]  
*Sent frames bytes counter.*
- int [RoundTripTime](#) [get]  
*Average time required voice packet to return to sender.*
- int [RoundTripTimeVariance](#) [get]  
*Average round trip time variation.*
- bool [SuppressInfoDuplicateWarning](#) [get, set]  
*Do not log warning when duplicate info received.*
- [RemoteVoiceInfoDelegate OnRemoteVoiceInfoAction](#) [get, set]  
*Register a method to be called when remote voice info arrived (after join or new new remote voice creation). Metod parameters: (int channelId, int playerId, byte voiceId, [VoiceInfo](#) voiceInfo, ref [RemoteVoiceOptions](#) options);*
- int [DebugLostPercent](#) [get, set]  
*Lost frames simulation ratio.*
- IEnumerable< [LocalVoice](#) > [LocalVoices](#) [get]  
*Iterates through copy of all local voices list.*
- IEnumerable< [RemoteVoiceInfo](#) > [RemoteVoiceInfos](#) [get]  
*Iterates through all remote voices infos.*

### 3.103.1 Detailed Description

[Voice](#) client interact with other clients on network via [IVoiceTransport](#).

### 3.103.2 Member Function Documentation

#### 3.103.2.1 LocalVoice CreateLocalVoice ( [VoiceInfo](#) voiceInfo, int channelId = 0, [IEncoder](#) encoder = null )

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

##### Parameters

<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 transport.
<i>encoder</i>	Encoder producing the stream.

##### Returns

Outgoing stream handler.

#### 3.103.2.2 LocalVoiceAudio<T> CreateLocalVoiceAudio< T > ( [VoiceInfo](#) voiceInfo, [IAudioDesc](#) audioSourceDesc, int channelId = 0, [IEncoder](#) encoder = null )

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

## Template Parameters

<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 transport.
<i>encoder</i>	Audio encoder. Set to null to use default Opus encoder.

## Returns

Outgoing stream handler.

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

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

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

## Parameters

<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 transport.
<i>encoder</i>	Audio encoder. Set to null to use default Opus encoder.

## Returns

Outgoing stream handler.

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

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

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

## Template Parameters

<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>frameSize</i>	Size of buffer <a href="#">LocalVoiceFramed</a> repacks input data stream to.
<i>channelId</i>	Transport channel specific to transport.
<i>encoder</i>	Encoder compressing data stream in pipeline.

**Returns**

Outgoing stream handler.

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

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

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

Remote voice info event delegate.

**3.103.2.7 void RemoveLocalVoice ( LocalVoice voice )**

Removes local voice (outgoing data stream).

**Parameters**

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

**3.103.2.8 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).

**3.103.3 Property Documentation****3.103.3.1 int DebugLostPercent [get], [set]**

Lost frames simulation ratio.

**3.103.3.2 int FramesLost [get], [set]**

Lost frames counter.

**3.103.3.3 int FramesReceived [get]**

Received frames counter.

**3.103.3.4 int FramesSent [get]**

Sent frames counter.

**3.103.3.5 int FramesSentBytes [get]**

Sent frames bytes counter.

### 3.103.3.6 IEnumerable<LocalVoice> LocalVoices [get]

Iterates through copy of all local voices list.

### 3.103.3.7 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);

### 3.103.3.8 IEnumerable<RemoteVoiceInfo> RemoteVoiceInfos [get]

Iterates through all remote voices infos.

### 3.103.3.9 int RoundTripTime [get]

Average time required voice packet to return to sender.

### 3.103.3.10 int RoundTripTimeVariance [get]

Average round trip time variation.

### 3.103.3.11 bool SuppressInfoDuplicateWarning [get], [set]

Do not log warning when duplicate info received.

## 3.104 VoiceComponent Class Reference

Inherits MonoBehaviour, and [ILoggable](#).

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

### Protected Member Functions

- virtual void **Awake** ()

### Protected Attributes

- DebugLevel **LogLevel** = DebugLevel.ERROR

### Properties

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

## 3.105 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*
- float [MinimalTimeScaleToDispatchInFixedUpdate](#) = -1f  
*Configures the minimal Time.timeScale at which [Voice](#) client will dispatch incoming messages within LateUpdate.*

## Protected Member Functions

- override void **Awake** ()
- virtual void **Update** ()
- virtual void **FixedUpdate** ()
- void [Dispatch](#) ()  
*Dispatches incoming network messages for [Voice](#) client. Called in FixedUpdate or LateUpdate.*
- override void **OnDisable** ()
- virtual void **OnDestroy** ()
- virtual [Speaker](#) **SimpleSpeakerFactory** (int playerId, byte voiceld, object userData)
- virtual void **OnVoiceStateChanged** ([ClientState](#) fromState, [ClientState](#) toState)
- void **CalcStatistics** ()
- void **LinkSpeaker** ([Speaker](#) speaker, [RemoteVoiceLink](#) remoteVoice)

## Protected Attributes

- List< [RemoteVoiceLink](#) > **cachedRemoteVoices** = new List<[RemoteVoiceLink](#)>()

## Properties

- [VoiceLogger](#) [Logger](#) [get, protected set]  
*[Logger](#) used by this component*
- DebugLevel [LogLevel](#) [get, set]  
*Log level for this component*
- new [LoadBalancingTransport](#) **Client** [get]
- [VoiceClient](#) [VoiceClient](#) [get]  
*Returns underlying [Photon Voice](#) client.*
- ClientState [ClientState](#) [get]  
*Returns [Photon Voice](#) client state.*
- float [FramesReceivedPerSecond](#) [get]  
*Number of frames received per second.*
- float [FramesLostPerSecond](#) [get]  
*Number of frames lost per second.*
- float [FramesLostPercent](#) [get]  
*Percentage of lost frames.*
- GameObject [SpeakerPrefab](#) [get, set]  
*Prefab that contains [Speaker](#) component to be instantiated when receiving a new remote audio source info*

## Events

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

### 3.105.1 Detailed Description

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

### 3.105.2 Member Function Documentation

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

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

##### Parameters

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

##### Returns

If true voice connection command was sent from client

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

Dispatches incoming network messages for [Voice](#) client. Called in FixedUpdate or LateUpdate.

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

Without dispatching messages, [Voice](#) client won't change state and does not handle updates.

### 3.105.3 Member Data Documentation

#### 3.105.3.1 `float MinimalTimeScaleToDispatchInFixedUpdate = -1f`

Configures the minimal Time.timeScale at which [Voice](#) client will dispatch incoming messages within LateUpdate.

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

#### 3.105.3.2 `Recorder PrimaryRecorder`

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

#### 3.105.3.3 `AppSettings Settings`

Settings to be used by this voice connection

### 3.105.3.4 Func<int, byte, object, [Speaker](#)> SpeakerFactory

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

## 3.105.4 Property Documentation

### 3.105.4.1 ClientState ClientState [get]

Returns [Photon Voice](#) client state.

### 3.105.4.2 float FramesLostPercent [get]

Percentage of lost frames.

### 3.105.4.3 float FramesLostPerSecond [get]

Number of frames lost per second.

### 3.105.4.4 float FramesReceivedPerSecond [get]

Number of frames received per second.

### 3.105.4.5 VoiceLogger Logger [get], [protected set]

[Logger](#) used by this component

### 3.105.4.6 DebugLevel LogLevel [get], [set]

Log level for this component

### 3.105.4.7 GameObject SpeakerPrefab [get], [set]

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

### 3.105.4.8 VoiceClient VoiceClient [get]

Returns underlying [Photon Voice](#) client.

## 3.105.5 Event Documentation

### 3.105.5.1 Action<[RemoteVoiceLink](#)> RemoteVoiceAdded

Fires when a remote voice stream is added

### 3.105.5.2 Action<[Speaker](#)> SpeakerLinked

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

### 3.106 AudioUtil.VoiceDetector< T > Class Template Reference

Simple voice activity detector triggered by signal level.

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

#### Public Member Functions

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

#### Protected Attributes

- float **norm**
- float **threshold**
- int **activityDelay**
- int **autoSilenceCounter** = 0
- int **valuesCountPerSec**
- int **activityDelayValuesCount**

#### Properties

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

#### Events

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

#### 3.106.1 Detailed Description

Simple voice activity detector triggered by signal level.

#### 3.106.2 Member Function Documentation

##### 3.106.2.1 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 >](#).

## 3.106.3 Property Documentation

3.106.3.1 `int ActivityDelayMs` [\[get\]](#), [\[set\]](#)

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

3.106.3.2 `bool Detected` [\[get\]](#), [\[protected set\]](#)

If true, voice detected.

3.106.3.3 `DateTime DetectedTime` [\[get\]](#)

Last time when switched to detected state.

3.106.3.4 `bool On` [\[get\]](#), [\[set\]](#)

If true, voice detection enabled.

3.106.3.5 `float Threshold` [\[get\]](#), [\[set\]](#)

[Voice](#) detected as soon as signal level exceeds threshold.

## 3.106.4 Event Documentation

3.106.4.1 `Action OnDetected`

Called when switched to detected state.

## 3.107 AudioUtil.VoiceDetectorCalibration&lt; T &gt; Class Template Reference

Calibration Utility for [Voice](#) Detector

Inherits [IProcessor< T >](#).

## Public Member Functions

- [VoiceDetectorCalibration](#) ([IVoiceDetector](#) voiceDetector, [ILevelMeter](#) levelMeter, int samplingRate, int channels)  
Create new [VoiceDetectorCalibration](#) instance.
- void [Calibrate](#) (int durationMs, Action< float > onCalibrated=null)  
Start calibration.

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

### Protected Attributes

- `int calibrateCount`

### Properties

- `bool IsCalibrating` [get]

### 3.107.1 Detailed Description

Calibration Utility for [Voice](#) Detector

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

### 3.107.2 Constructor & Destructor Documentation

#### 3.107.2.1 `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.

### 3.107.3 Member Function Documentation

#### 3.107.3.1 `void Calibrate ( int durationMs, Action< float > onCalibrated = null )`

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.

#### 3.107.3.2 `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 >](#).

## 3.108 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**

### 3.108.1 Detailed Description

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

## 3.109 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**

### 3.109.1 Detailed Description

[VoiceDetector](#) specialization for float audio.

### 3.109.2 Constructor & Destructor Documentation

#### 3.109.2.1 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.

### 3.110 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

##### 3.110.1 Detailed Description

[VoiceDetector](#) specialization for float audio.

##### 3.110.2 Constructor & Destructor Documentation

###### 3.110.2.1 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.

### 3.111 VoiceEvent Class Reference

#### Public Attributes

- const byte [Code](#) = 202  
*Single event used for voice communications.*

##### 3.111.1 Member Data Documentation

###### 3.111.1.1 const byte Code = 202

Single event used for voice communications.

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

## 3.112 VoicelInfo Struct Reference

Describes stream properties.

### Public Member Functions

- override string **ToString** ()

### Static Public Member Functions

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

### Properties

- [Codec](#) **Codec** [get, set]
- int [SamplingRate](#) [get, set]  
*Audio sampling rate (frequency, in Hz).*
- int [Channels](#) [get, set]  
*Source audio sampling rate (to be resampled to SamplingRate; in Hz).*
- int [FrameDurationUs](#) [get, set]  
*Uncompressed frame (audio packet) size in microseconds.*
- int [Bitrate](#) [get, set]  
*Target bitrate (in bits/second).*
- object [UserData](#) [get, set]  
*Optional user data. Should be serializable by [Photon](#).*
- int [FrameDurationSamples](#) [get]  
*Uncompressed frame (data packet) size in samples.*
- int [FrameSize](#) [get]  
*Uncompressed frame (data packet) array size.*
- int [Width](#) [get, set]  
*Video width (optional).*
- int [Height](#) [get, set]  
*Video height (optional)*

### 3.112.1 Detailed Description

Describes stream properties.

### 3.112.2 Member Function Documentation

- 3.112.2.1 **static VoicelInfo CreateAudio** ( **Codec** codec, int *samplingRate*, int *channels*, int *frameDurationUs*, object *userdata* = null ) [static]

Create stream info for an Opus audio stream.

## Parameters

<i>samplingRate</i>	Audio sampling rate.
<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.

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

Create stream info for an Opus audio stream.

## Parameters

<i>samplingRate</i>	Audio sampling rate.
<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.

**3.112.3 Property Documentation**

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

Target bitrate (in bits/second).

**3.112.3.2** `int Channels [get], [set]`

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

**3.112.3.3** `int FrameDurationSamples [get]`

Uncompressed frame (data packet) size in samples.

**3.112.3.4** `int FrameDurationUs [get], [set]`

Uncompressed frame (audio packet) size in microseconds.

**3.112.3.5** `int FrameSize [get]`

Uncompressed frame (data packet) array size.

**3.112.3.6** `int Height [get], [set]`

Video height (optional)

3.112.3.7 int **SamplingRate** [get], [set]

Audio sampling rate (frequency, in Hz).

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

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

3.112.3.9 int **Width** [get], [set]

Video width (optional).

## 3.113 AudioUtil.VoiceLevelDetectCalibrate< T > Class Template Reference

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

Inherits [IProcessor< T >](#).

### Public Member Functions

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

### Properties

- [ILevelMeter](#) **LevelMeter** [get]  
*The [LevelMeter](#) in use.*
- [IVoiceDetector](#) **VoiceDetector** [get]  
*The [VoiceDetector](#) in use*
- bool **IsCalibrating** [get]

### 3.113.1 Detailed Description

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

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

### 3.113.2 Constructor & Destructor Documentation

3.113.2.1 **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.

## 3.113.3 Member Function Documentation

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

Start calibration

## Parameters

<i>durationMs</i>	Duration of the calibration procedure (in milliseconds).
<i>onCalibrated</i>	Called when calibration is complete. Parameter is new threshold value.

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

3.113.3.2 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 >](#).

## 3.113.4 Property Documentation

## 3.113.4.1 ILevelMeter LevelMeter [get]

The [LevelMeter](#) in use.

## 3.113.4.2 IVoiceDetector VoiceDetector [get]

The [VoiceDetector](#) in use

## 3.114 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]

## 3.115 WebRtcAudioDsp Class Reference

Inherits [VoiceComponent](#).

## Protected Member Functions

- override void **Awake** ()

## Properties

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

## Additional Inherited Members

## 3.116 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)

### 3.117 WebRTCAudioProcessor Class Reference

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

#### Public Member Functions

- **WebRTCAudioProcessor** ([ILogger](#) logger, int frameSize, int samplingRate, int channels, int reverseSamplingRate, 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, [34](#)
- AcquireOrCreate
  - Photon::Voice::ObjectPool, [57](#)
- ActivityDelayMs
  - Photon::Voice::AudioUtil::IVoiceDetector, [40](#)
  - Photon::Voice::AudioUtil::VoiceDetector, [89](#)
- Actor
  - Photon::Voice::Unity::Speaker, [75](#)
- AddPostProcessor
  - Photon::Voice::LocalVoiceFramed, [52](#)
- AddPreProcessor
  - Photon::Voice::LocalVoiceFramed, [52](#)
- AllowBluetooth
  - Photon::Voice::IOS, [7](#)
- Ambient
  - Photon::Voice::IOS, [6](#)
- Audio
  - POpusCodec::Enums, [10](#)
- AudioClip
  - Photon::Voice::Unity::Recorder, [68](#)
- AudioClipWrapper, [11](#)
- AudioDesc, [11](#)
- AudioGroup
  - Photon::Voice::Unity::Recorder, [68](#)
- AudioInEnumerator, [11](#)
- AudioOpus
  - Photon::Voice, [5](#)
- AudioOutCapture, [12](#)
- AudioProcessing
  - Photon::Voice::IOS, [6](#)
- AudioSessionCategory
  - Photon::Voice::IOS, [6](#)
- AudioSessionCategoryOption
  - Photon::Voice::IOS, [6](#)
- AudioSessionMode
  - Photon::Voice::IOS, [7](#)
- AudioSessionParameters, [12](#)
- AudioSessionParametersPresets, [12](#)
- AudioStreamPlayer< T >, [13](#)
- AudioUtil, [13](#)
- AudioUtil.ILevelMeter, [33](#)
- AudioUtil.IVoiceDetector, [39](#)
- AudioUtil.LevelMeter< T >, [41](#)
- AudioUtil.LevelMeterDummy, [42](#)
- AudioUtil.LevelMeterFloat, [42](#)
- AudioUtil.LevelMeterShort, [43](#)
- AudioUtil.Resampler< T >, [73](#)
- AudioUtil.ToneAudioPusher< T >, [75](#)
- AudioUtil.ToneAudioReader< T >, [76](#)
- AudioUtil.VoiceDetector< T >, [88](#)
- AudioUtil.VoiceDetectorCalibration< T >, [89](#)
- AudioUtil.VoiceDetectorDummy, [91](#)
- AudioUtil.VoiceDetectorFloat, [91](#)
- AudioUtil.VoiceDetectorShort, [92](#)
- AudioUtil.VoiceLevelDetectCalibrate< T >, [95](#)
- Auto
  - POpusCodec::Enums, [10](#)
- AutoConnectAndJoin
  - Photon::Voice::PUN::PhotonVoiceNetwork, [62](#)
- AutoCreateRecorderIfNotFound
  - Photon::Voice::PUN::PhotonVoiceView, [64](#)
- AutoCreateSpeakerIfNotFound
  - Photon::Voice::PUN::PhotonVoiceNetwork, [62](#)
- AutoLeaveAndDisconnect
  - Photon::Voice::PUN::PhotonVoiceNetwork, [62](#)
- AutoStart
  - Photon::Voice::Unity::Recorder, [68](#)
- Bandwidth
  - POpusCodec::Enums, [9](#)
- Bitrate
  - Photon::Voice::Unity::Recorder, [69](#)
  - Photon::Voice::VoiceInfo, [94](#)
- BufferReaderPushAdapter
  - Photon::Voice::BufferReaderPushAdapter, [17](#)
- BufferReaderPushAdapter< T >, [17](#)
- BufferReaderPushAdapterAsyncPool
  - Photon::Voice::BufferReaderPushAdapterAsync↔  
Pool, [18](#)
- BufferReaderPushAdapterAsyncPool< T >, [18](#)
- BufferReaderPushAdapterAsyncPoolCopy
  - Photon::Voice::BufferReaderPushAdapterAsync↔  
PoolCopy, [19](#)
- BufferReaderPushAdapterAsyncPoolCopy< T >, [19](#)
- BufferReaderPushAdapterAsyncPoolFloatToShort, [20](#)
- Photon::Voice::BufferReaderPushAdapterAsync↔  
PoolFloatToShort, [20](#)
- BufferReaderPushAdapterBase
  - Photon::Voice::BufferReaderPushAdapterBase, [21](#)
- BufferReaderPushAdapterBase< T >, [20](#)
- Calibrate
  - Photon::Voice::AudioUtil::VoiceDetectorCalibration,  
[90](#)
  - Photon::Voice::AudioUtil::VoiceLevelDetect↔  
Calibrate, [96](#)
- ChannelId
  - Photon::Voice::RemoteVoiceInfo, [71](#)

- Channels
  - POpusCodec::Enums, 9
  - Photon::Voice::AudioUtil::ToneAudioReader, 78
  - Photon::Voice::IAudioDesc, 29
  - Photon::Voice::VoiceInfo, 94
- ClearProcessors
  - Photon::Voice::LocalVoiceFramed, 52
- ClientState
  - Photon::Voice::Unity::VoiceConnection, 87
- Code
  - Photon::Voice::VoiceEvent, 92
- Codec
  - Photon::Voice, 5
- ConnectAndJoin, 22
- ConnectAndJoinRoom
  - Photon::Voice::PUN::PhotonVoiceNetwork, 62
- ConnectUsingSettings
  - Photon::Voice::Unity::VoiceConnection, 86
- Convert
  - Photon::Voice::AudioUtil, 15, 16
- Count
  - Photon::Voice::Framer, 28
- Create
  - Photon::Voice::LocalVoiceAudio, 49
- CreateAudio
  - Photon::Voice::VoiceInfo, 93
- CreateAudioOpus
  - Photon::Voice::VoiceInfo, 94
- CreateLocalVoice
  - Photon::Voice::VoiceClient, 81
- CreateLocalVoiceAudio< T >
  - Photon::Voice::VoiceClient, 81
- CreateLocalVoiceAudioFromSource
  - Photon::Voice::VoiceClient, 82
- CreateLocalVoiceFramed< T >
  - Photon::Voice::VoiceClient, 82
- CurrentAvgAmp
  - Photon::Voice::AudioUtil::ILevelMeter, 34
- CurrentPeakAmp
  - Photon::Voice::AudioUtil::ILevelMeter, 34
- DebugEchoMode
  - Photon::Voice::LocalVoice, 47
  - Photon::Voice::Unity::Recorder, 69
- DebugLostPercent
  - Photon::Voice::VoiceClient, 83
- Decoder
  - Photon::Voice::RemoteVoiceOptions, 73
- Default
  - Photon::Voice::IOS, 7
- DefaultToSpeaker
  - Photon::Voice::IOS, 7
- Delay
  - POpusCodec::Enums, 9
- Delay10ms
  - POpusCodec::Enums, 10
- Delay20ms
  - POpusCodec::Enums, 10
- Delay2dot5ms
  - POpusCodec::Enums, 10
- Delay40ms
  - POpusCodec::Enums, 10
- Delay5ms
  - POpusCodec::Enums, 10
- Delay60ms
  - POpusCodec::Enums, 10
- DequeueOutput
  - Photon::Voice::IEncoder, 32
- Detected
  - Photon::Voice::AudioUtil::IVoiceDetector, 40
  - Photon::Voice::AudioUtil::VoiceDetector, 89
- DetectedTime
  - Photon::Voice::AudioUtil::IVoiceDetector, 40
  - Photon::Voice::AudioUtil::VoiceDetector, 89
- Disconnect
  - Photon::Voice::PUN::PhotonVoiceNetwork, 62
- Dispatch
  - Photon::Voice::Unity::VoiceConnection, 86
- Dispose
  - Photon::Voice::BufferReaderPushAdapterBase, 21
  - Photon::Voice::LoadBalancingTransport, 45
  - Photon::Voice::LocalVoiceFramed, 52
  - Photon::Voice::ObjectPool, 57
- DuckOthers
  - Photon::Voice::IOS, 6
- Dummy
  - Photon::Voice::LocalVoiceAudioDummy, 51
- EncoderDelay
  - POpusCodec::OpusEncoder, 59
- Encrypt
  - Photon::Voice::LocalVoice, 47
  - Photon::Voice::Unity::Recorder, 69
- Error
  - Photon::Voice::AudioUtil::ToneAudioReader, 78
  - Photon::Voice::IAudioDesc, 29
  - Photon::Voice::IDecoder, 32
  - Photon::Voice::IEncoder, 33
- FactoryPrimitiveArrayPool< T >, 26
- FactoryReusableArray< T >, 27
- ForceToStereo< T >
  - Photon::Voice::AudioUtil, 16
- Frame
  - Photon::Voice::Framer, 28
- FrameDuration
  - Photon::Voice::Unity::Recorder, 69
- FrameDurationSamples
  - Photon::Voice::VoiceInfo, 94
- FrameDurationUs
  - Photon::Voice::VoiceInfo, 94
- FrameSize
  - Photon::Voice::LocalVoiceFramedBase, 53
  - Photon::Voice::VoiceInfo, 94
- Framer
  - Photon::Voice::Framer, 28
- Framer< T >, 27
- FramesLost

- Photon::Voice::VoiceClient, 83
- FramesLostPerSecond
  - Photon::Voice::Unity::VoiceConnection, 87
- FramesLostPercent
  - Photon::Voice::Unity::VoiceConnection, 87
- FramesReceived
  - Photon::Voice::VoiceClient, 83
- FramesReceivedPerSecond
  - Photon::Voice::Unity::VoiceConnection, 87
- FramesSent
  - Photon::Voice::LocalVoice, 47
  - Photon::Voice::VoiceClient, 83
- FramesSentBytes
  - Photon::Voice::LocalVoice, 47
  - Photon::Voice::VoiceClient, 83
- Fullband
  - POpusCodec::Enums, 9
- Game
  - Photon::Voice::IOS::AudioSessionParameters←Presets, 13
- GlobalInterestGroup
  - Photon::Voice::LoadBalancingTransport, 46
- Height
  - Photon::Voice::VoiceInfo, 94
- IAudioDesc, 28
- IAudioOut< T >, 29
- IAudioPusher< T >, 29
- IAudioReader< T >, 30
- IDataReader< T >, 30
- IDecoder, 31
- IDecoderQueuedOutputImageNative, 32
- IEncoder, 32
- IEncoderDirect< B >, 33
- ILocalVoiceAudio, 34
- ILoggable, 35
- ILogger, 36
- IOSAudioForceToSpeaker, 38
- IProcessor< T >, 38
- IServiceable, 38
- ISyncAudioOut< T >, 39
- IVoiceTransport, 40
- ImageBufferInfo, 36
- ImageBufferNative, 36
- ImageBufferNativeAlloc, 36
- ImageBufferNativeGCHandleSinglePlane, 37
- ImageBufferNativePool< T >, 37
- ImageInputBuf, 37
- ImageOutputBuf, 38
- Info
  - Photon::Voice::LocalVoice, 47
  - Photon::Voice::ObjectPool, 58
  - Photon::Voice::RemoteVoiceInfo, 71
- Init
  - Photon::Voice::ObjectPool, 57
  - Photon::Voice::PUN::PhotonVoiceView, 64
  - Photon::Voice::Unity::Recorder, 68
- Input
  - Photon::Voice::IDecoder, 31
  - Photon::Voice::IEncoderDirect, 33
  - Photon::Voice::OpusCodec::Decoder, 23
  - Photon::Voice::RawCodec::Decoder, 24
- InputFactory
  - Photon::Voice::Unity::Recorder, 69
- Instance
  - Photon::Voice::PUN::PhotonVoiceNetwork, 62
- InterestGroup
  - Photon::Voice::LocalVoice, 48
  - Photon::Voice::Unity::Recorder, 69
- IsCurrentlyTransmitting
  - Photon::Voice::LocalVoice, 48
  - Photon::Voice::Unity::Recorder, 69
- IsInitialized
  - Photon::Voice::Unity::Recorder, 69
- IsLinked
  - Photon::Voice::Unity::Speaker, 75
- IsPlaying
  - Photon::Voice::Unity::Speaker, 75
- IsRecorder
  - Photon::Voice::PUN::PhotonVoiceView, 64
- IsRecording
  - Photon::Voice::PUN::PhotonVoiceView, 64
  - Photon::Voice::Unity::Recorder, 69
- IsSetup
  - Photon::Voice::PUN::PhotonVoiceView, 64
- IsSpeaker
  - Photon::Voice::PUN::PhotonVoiceView, 64
- IsSpeaking
  - Photon::Voice::PUN::PhotonVoiceView, 65
- Lag
  - Photon::Voice::Unity::Speaker, 75
- LevelMeter
  - Photon::Voice::AudioUtil::VoiceLevelDetect←Calibrate, 96
  - Photon::Voice::ILocalVoiceAudio, 35
  - Photon::Voice::Unity::Recorder, 69
- LevelMeterFloat
  - Photon::Voice::AudioUtil::LevelMeterFloat, 43
- LevelMeterShort
  - Photon::Voice::AudioUtil::LevelMeterShort, 43
- LoadBalancingFrontend, 44
- LoadBalancingTransport, 44
  - Photon::Voice::LoadBalancingTransport, 45
- LocalUserServiceable
  - Photon::Voice::LocalVoice, 48
- LocalVoice, 46
- LocalVoiceAudio< T >, 48
- LocalVoiceAudioDummy, 50
- LocalVoiceAudioFloat, 51
- LocalVoiceAudioShort, 51
- LocalVoiceFramed< T >, 51
- LocalVoiceFramedBase, 53
- LocalVoices
  - Photon::Voice::VoiceClient, 83
- LocalVoicesInChannel

- Photon::Voice::VoiceClient, [83](#)
- LogLevel
  - Photon::Voice::Unity::VoiceConnection, [87](#)
- Logger, [54](#)
  - Photon::Voice::Unity::VoiceConnection, [87](#)
- LoopAudioClip
  - Photon::Voice::Unity::Recorder, [69](#)
- Measurement
  - Photon::Voice::IOS, [7](#)
- Mediumband
  - POpusCodec::Enums, [9](#)
- MicAmplifier, [54](#)
- MicAmplifierFloat, [54](#)
- MicAmplifierShort, [54](#)
- MicWrapper, [55](#)
- MicrophoneType
  - Photon::Voice::Unity::Recorder, [69](#)
- MinimalTimeScaleToDispatchInFixedUpdate
  - Photon::Voice::Unity::VoiceConnection, [86](#)
- MixWithOthers
  - Photon::Voice::IOS, [6](#)
- Mono
  - POpusCodec::Enums, [9](#)
- MoviePlayback
  - Photon::Voice::IOS, [7](#)
- MultiRoute
  - Photon::Voice::IOS, [6](#)
- Music
  - POpusCodec::Enums, [10](#)
- Narrowband
  - POpusCodec::Enums, [9](#)
- ObjectFactory< TType, TInfo >, [55](#)
- ObjectPool
  - Photon::Voice::ObjectPool, [57](#)
- ObjectPool< TType, TInfo >, [56](#)
- On
  - Photon::Voice::AudioUtil::IVoiceDetector, [40](#)
  - Photon::Voice::AudioUtil::VoiceDetector, [89](#)
- OnDetected
  - Photon::Voice::AudioUtil::IVoiceDetector, [40](#)
  - Photon::Voice::AudioUtil::VoiceDetector, [89](#)
- OnRemoteVoiceInfoAction
  - Photon::Voice::VoiceClient, [84](#)
- OnRemoteVoiceRemoveAction
  - Photon::Voice::RemoteVoiceOptions, [73](#)
  - Photon::Voice::Unity::Speaker, [75](#)
- Open
  - Photon::Voice::IDecoder, [31](#)
  - Photon::Voice::OpusCodec::Decoder, [23](#)
  - Photon::Voice::RawCodec::Decoder, [24](#)
- OpusApplicationType
  - POpusCodec::Enums, [10](#)
- OpusCodec, [58](#)
- OpusCodec.Decoder< T >, [22](#)
- OpusCodec.DecoderFactory, [24](#)
- OpusCodec.DecoderFloat, [24](#)
- OpusCodec.DecoderShort, [24](#)
- OpusCodec.Encoder< T >, [25](#)
- OpusCodec.EncoderFloat, [25](#)
- OpusCodec.EncoderShort, [26](#)
- OpusCodec.Factory, [26](#)
- OpusCodec.Util, [80](#)
- OpusDecoder, [58](#)
- OpusEncoder, [59](#)
- OpusException, [60](#)
- Output
  - Photon::Voice::LEncoder, [33](#)
- POpusCodec, [9](#)
- POpusCodec.Enums, [9](#)
- POpusCodec::Enums
  - Audio, [10](#)
  - Auto, [10](#)
  - Bandwidth, [9](#)
  - Channels, [9](#)
  - Delay, [9](#)
  - Delay10ms, [10](#)
  - Delay20ms, [10](#)
  - Delay2dot5ms, [10](#)
  - Delay40ms, [10](#)
  - Delay5ms, [10](#)
  - Delay60ms, [10](#)
  - Fullband, [9](#)
  - Mediumband, [9](#)
  - Mono, [9](#)
  - Music, [10](#)
  - Narrowband, [9](#)
  - OpusApplicationType, [10](#)
  - RestrictedLowDelay, [10](#)
  - SignalHint, [10](#)
  - Stereo, [9](#)
  - SuperWideband, [9](#)
  - Voice, [10](#)
  - Voip, [10](#)
  - Wideband, [9](#)
- POpusCodec::OpusEncoder
  - EncoderDelay, [59](#)
- Photon, [3](#)
- Photon.Voice, [3](#)
- Photon.Voice.IOS, [5](#)
- Photon.Voice.PUN, [8](#)
- Photon.Voice.Unity, [8](#)
- Photon.Voice.Unity.UtilityScripts, [8](#)
- Photon::Voice
  - AudioOpus, [5](#)
  - Codec, [5](#)
- Photon::Voice::AudioUtil
  - Convert, [15](#), [16](#)
  - ForceToStereo< T >, [16](#)
  - Resample< T >, [16](#)
  - ResampleAndConvert, [16](#), [17](#)
- Photon::Voice::AudioUtil::ILevelMeter
  - AccumAvgPeakAmp, [34](#)
  - CurrentAvgAmp, [34](#)
  - CurrentPeakAmp, [34](#)

- ResetAccumAvgPeakAmp, [34](#)
- Photon::Voice::AudioUtil::IVoiceDetector
  - ActivityDelayMs, [40](#)
  - Detected, [40](#)
  - DetectedTime, [40](#)
  - On, [40](#)
  - OnDetected, [40](#)
  - Threshold, [40](#)
- Photon::Voice::AudioUtil::LevelMeter
  - Process, [41](#)
  - ResetAccumAvgPeakAmp, [42](#)
- Photon::Voice::AudioUtil::LevelMeterDummy
  - ResetAccumAvgPeakAmp, [42](#)
- Photon::Voice::AudioUtil::LevelMeterFloat
  - LevelMeterFloat, [43](#)
- Photon::Voice::AudioUtil::LevelMeterShort
  - LevelMeterShort, [43](#)
- Photon::Voice::AudioUtil::Resampler
  - Process, [74](#)
  - Resampler, [73](#)
- Photon::Voice::AudioUtil::ToneAudioPusher
  - SetCallback, [76](#)
  - ToneAudioPusher, [76](#)
- Photon::Voice::AudioUtil::ToneAudioReader
  - Channels, [78](#)
  - Error, [78](#)
  - Read, [77](#)
  - SamplingRate, [78](#)
  - ToneAudioReader, [77](#)
- Photon::Voice::AudioUtil::VoiceDetector
  - ActivityDelayMs, [89](#)
  - Detected, [89](#)
  - DetectedTime, [89](#)
  - On, [89](#)
  - OnDetected, [89](#)
  - Process, [88](#)
  - Threshold, [89](#)
- Photon::Voice::AudioUtil::VoiceDetectorCalibration
  - Calibrate, [90](#)
  - Process, [90](#)
  - VoiceDetectorCalibration, [90](#)
- Photon::Voice::AudioUtil::VoiceDetectorFloat
  - VoiceDetectorFloat, [91](#)
- Photon::Voice::AudioUtil::VoiceDetectorShort
  - VoiceDetectorShort, [92](#)
- Photon::Voice::AudioUtil::VoiceLevelDetectCalibrate
  - Calibrate, [96](#)
  - LevelMeter, [96](#)
  - Process, [96](#)
  - VoiceDetector, [96](#)
  - VoiceLevelDetectCalibrate, [95](#)
- Photon::Voice::BufferReaderPushAdapter
  - BufferReaderPushAdapter, [17](#)
  - Service, [18](#)
- Photon::Voice::BufferReaderPushAdapterAsyncPool
  - BufferReaderPushAdapterAsyncPool, [18](#)
  - Service, [18](#)
- Photon::Voice::BufferReaderPushAdapterAsyncPool↔
  - Copy
    - BufferReaderPushAdapterAsyncPoolCopy, [19](#)
  - Service, [19](#)
- Photon::Voice::BufferReaderPushAdapterAsyncPool↔
  - FloatToShort
    - BufferReaderPushAdapterAsyncPoolFloatToShort, [20](#)
  - Service, [20](#)
- Photon::Voice::BufferReaderPushAdapterBase
  - BufferReaderPushAdapterBase, [21](#)
  - Dispose, [21](#)
  - Service, [21](#)
- Photon::Voice::Framer
  - Count, [28](#)
  - Frame, [28](#)
  - Framer, [28](#)
- Photon::Voice::IAudioDesc
  - Channels, [29](#)
  - Error, [29](#)
  - SamplingRate, [29](#)
- Photon::Voice::IAudioPusher
  - SetCallback, [30](#)
- Photon::Voice::IDataReader
  - Read, [30](#)
- Photon::Voice::IDecoder
  - Error, [32](#)
  - Input, [31](#)
  - Open, [31](#)
- Photon::Voice::IEncoder
  - DequeueOutput, [32](#)
  - Error, [33](#)
  - Output, [33](#)
- Photon::Voice::IEncoderDirect
  - Input, [33](#)
- Photon::Voice::ILocalVoiceAudio
  - LevelMeter, [35](#)
  - VoiceDetector, [35](#)
  - VoiceDetectorCalibrate, [35](#)
  - VoiceDetectorCalibrating, [35](#)
- Photon::Voice::IOS
  - AllowBluetooth, [7](#)
  - Ambient, [6](#)
  - AudioProcessing, [6](#)
  - AudioSessionCategory, [6](#)
  - AudioSessionCategoryOption, [6](#)
  - AudioSessionMode, [7](#)
  - Default, [7](#)
  - DefaultToSpeaker, [7](#)
  - DuckOthers, [6](#)
  - Measurement, [7](#)
  - MixWithOthers, [6](#)
  - MoviePlayback, [7](#)
  - MultiRoute, [6](#)
  - PlayAndRecord, [6](#)
  - Playback, [6](#)
  - Record, [6](#)
  - SoloAmbient, [6](#)

- VideoChat, [7](#)
- VideoRecording, [7](#)
- VoiceChat, [7](#)
- Photon::Voice::IOS::AudioSessionParametersPresets
  - Game, [13](#)
  - VoIP, [13](#)
- Photon::Voice::IPProcessor
  - Process, [38](#)
- Photon::Voice::IServiceable
  - Service, [39](#)
- Photon::Voice::LoadBalancingTransport
  - Dispose, [45](#)
  - GlobalInterestGroup, [46](#)
  - LoadBalancingTransport, [45](#)
  - SendDebugEchoVoicesInfo, [45](#)
  - Service, [45](#)
  - VoiceClient, [46](#)
- Photon::Voice::LocalVoice
  - DebugEchoMode, [47](#)
  - Encrypt, [47](#)
  - FramesSent, [47](#)
  - FramesSentBytes, [47](#)
  - Info, [47](#)
  - InterestGroup, [48](#)
  - IsCurrentlyTransmitting, [48](#)
  - LocalUserServiceable, [48](#)
  - Reliable, [48](#)
  - RemoveSelf, [47](#)
  - TransmitEnabled, [48](#)
- Photon::Voice::LocalVoiceAudio
  - Create, [49](#)
  - VoiceDetectorCalibrate, [49](#)
  - VoiceDetectorCalibrating, [50](#)
- Photon::Voice::LocalVoiceAudioDummy
  - Dummy, [51](#)
  - VoiceDetectorCalibrate, [50](#)
- Photon::Voice::LocalVoiceFramed
  - AddPostProcessor, [52](#)
  - AddPreProcessor, [52](#)
  - ClearProcessors, [52](#)
  - Dispose, [52](#)
  - PushData, [53](#)
  - PushDataAsync, [53](#)
  - PushDataAsyncReady, [53](#)
- Photon::Voice::LocalVoiceFramedBase
  - FrameSize, [53](#)
- Photon::Voice::ObjectPool
  - AcquireOrCreate, [57](#)
  - Dispose, [57](#)
  - Info, [58](#)
  - Init, [57](#)
  - ObjectPool, [57](#)
  - Release, [58](#)
- Photon::Voice::OpusCodec::Decoder
  - Input, [23](#)
  - Open, [23](#)
- Photon::Voice::PUN::PhotonVoiceNetwork
  - AutoConnectAndJoin, [62](#)
  - AutoCreateSpeakerIfNotFound, [62](#)
  - AutoLeaveAndDisconnect, [62](#)
  - ConnectAndJoinRoom, [62](#)
  - Disconnect, [62](#)
  - Instance, [62](#)
  - VoiceRoomNameSuffix, [62](#)
- Photon::Voice::PUN::PhotonVoiceView
  - AutoCreateRecorderIfNotFound, [64](#)
  - Init, [64](#)
  - IsRecorder, [64](#)
  - IsRecording, [64](#)
  - IsSetup, [64](#)
  - IsSpeaker, [64](#)
  - IsSpeaking, [65](#)
  - RecorderInUse, [65](#)
  - SetupDebugSpeaker, [64](#)
  - SpeakerInUse, [65](#)
  - UsePrimaryRecorder, [64](#)
- Photon::Voice::RawCodec::Decoder
  - Input, [24](#)
  - Open, [24](#)
- Photon::Voice::RemoteVoiceInfo
  - ChannelId, [71](#)
  - Info, [71](#)
  - PlayerId, [71](#)
  - VoiceId, [71](#)
- Photon::Voice::RemoteVoiceOptions
  - Decoder, [73](#)
  - OnRemoteVoiceRemoveAction, [73](#)
  - SetOutput, [73](#)
- Photon::Voice::Unity::Recorder
  - AudioClip, [68](#)
  - AudioGroup, [68](#)
  - AutoStart, [68](#)
  - Bitrate, [69](#)
  - DebugEchoMode, [69](#)
  - Encrypt, [69](#)
  - FrameDuration, [69](#)
  - Init, [68](#)
  - InputFactory, [69](#)
  - InterestGroup, [69](#)
  - IsCurrentlyTransmitting, [69](#)
  - IsInitialized, [69](#)
  - IsRecording, [69](#)
  - LevelMeter, [69](#)
  - LoopAudioClip, [69](#)
  - MicrophoneType, [69](#)
  - PhotonMicrophoneDeviceId, [70](#)
  - PhotonMicrophoneEnumerator, [70](#)
  - ReliableMode, [70](#)
  - RequiresRestart, [70](#)
  - RestartRecording, [68](#)
  - SamplingRate, [70](#)
  - SourceType, [70](#)
  - StartRecording, [68](#)
  - StopRecording, [68](#)
  - TransmitEnabled, [70](#)
  - TypeConvert, [70](#)



- UnityMicrophoneDevice, 70
- UserData, 70
- VoiceDetection, 70
- VoiceDetectionDelayMs, 70
- VoiceDetectionThreshold, 71
- VoiceDetector, 71
- VoiceDetectorCalibrate, 68
- VoiceDetectorCalibrating, 71
- Photon::Voice::Unity::Speaker
  - Actor, 75
  - IsLinked, 75
  - IsPlaying, 75
  - Lag, 75
  - OnRemoteVoiceRemoveAction, 75
- Photon::Voice::Unity::VoiceConnection
  - ClientState, 87
  - ConnectUsingSettings, 86
  - Dispatch, 86
  - FramesLostPerSecond, 87
  - FramesLostPercent, 87
  - FramesReceivedPerSecond, 87
  - LogLevel, 87
  - Logger, 87
  - MinimalTimeScaleToDispatchInFixedUpdate, 86
  - PrimaryRecorder, 86
  - RemoteVoiceAdded, 87
  - Settings, 86
  - SpeakerFactory, 86
  - SpeakerLinked, 87
  - SpeakerPrefab, 87
  - VoiceClient, 87
- Photon::Voice::UnsupportedCodecException
  - UnsupportedCodecException, 79
- Photon::Voice::UnsupportedSampleTypeException
  - UnsupportedSampleTypeException, 80
- Photon::Voice::VoiceClient
  - CreateLocalVoice, 81
  - CreateLocalVoiceAudio< T >, 81
  - CreateLocalVoiceAudioFromSource, 82
  - CreateLocalVoiceFramed< T >, 82
  - DebugLostPercent, 83
  - FramesLost, 83
  - FramesReceived, 83
  - FramesSent, 83
  - FramesSentBytes, 83
  - LocalVoices, 83
  - LocalVoicesInChannel, 83
  - OnRemoteVoiceInfoAction, 84
  - RemoteVoiceInfoDelegate, 83
  - RemoteVoiceInfos, 84
  - RemoveLocalVoice, 83
  - RoundTripTime, 84
  - RoundTripTimeVariance, 84
  - Service, 83
  - SuppressInfoDuplicateWarning, 84
- Photon::Voice::VoiceEvent
  - Code, 92
- Photon::Voice::VoiceInfo
  - Bitrate, 94
  - Channels, 94
  - CreateAudio, 93
  - CreateAudioOpus, 94
  - FrameDurationSamples, 94
  - FrameDurationUs, 94
  - FrameSize, 94
  - Height, 94
  - SamplingRate, 94
  - UserData, 95
  - Width, 95
- PhotonMicrophoneDeviceId
  - Photon::Voice::Unity::Recorder, 70
- PhotonMicrophoneEnumerator
  - Photon::Voice::Unity::Recorder, 70
- PhotonVoiceCreatedParams, 60
- PhotonVoiceLagSimulationGui, 61
- PhotonVoiceNetwork, 61
- PhotonVoiceStatsGui, 63
- PhotonVoiceView, 63
- PlayAndRecord
  - Photon::Voice::IOS, 6
- Playback
  - Photon::Voice::IOS, 6
- PlayerId
  - Photon::Voice::RemoteVoiceInfo, 71
- PrimaryRecorder
  - Photon::Voice::Unity::VoiceConnection, 86
- PrimitiveArrayPool< T >, 65
- Process
  - Photon::Voice::AudioUtil::LevelMeter, 41
  - Photon::Voice::AudioUtil::Resampler, 74
  - Photon::Voice::AudioUtil::VoiceDetector, 88
  - Photon::Voice::AudioUtil::VoiceDetectorCalibration, 90
  - Photon::Voice::AudioUtil::VoiceLevelDetect← Calibrate, 96
  - Photon::Voice::IProcessor, 38
- PushData
  - Photon::Voice::LocalVoiceFramed, 53
- PushDataAsync
  - Photon::Voice::LocalVoiceFramed, 53
- PushDataAsyncReady
  - Photon::Voice::LocalVoiceFramed, 53
- RawCodec, 65
- RawCodec.Decoder< T >, 23
- RawCodec.Encoder< T >, 25
- Read
  - Photon::Voice::AudioUtil::ToneAudioReader, 77
  - Photon::Voice::IDataReader, 30
- Record
  - Photon::Voice::IOS, 6
- Recorder, 66
- Recorder.PhotonVoiceCreatedParams, 60
- RecorderInUse
  - Photon::Voice::PUN::PhotonVoiceView, 65
- Release
  - Photon::Voice::ObjectPool, 58

- Reliable
  - Photon::Voice::LocalVoice, [48](#)
- ReliableMode
  - Photon::Voice::Unity::Recorder, [70](#)
- RemoteVoiceAdded
  - Photon::Voice::Unity::VoiceConnection, [87](#)
- RemoteVoiceInfo, [71](#)
- RemoteVoiceInfoDelegate
  - Photon::Voice::VoiceClient, [83](#)
- RemoteVoiceInfos
  - Photon::Voice::VoiceClient, [84](#)
- RemoteVoiceLink, [72](#)
- RemoteVoiceOptions, [72](#)
- RemoveLocalVoice
  - Photon::Voice::VoiceClient, [83](#)
- RemoveSelf
  - Photon::Voice::LocalVoice, [47](#)
- RequiresRestart
  - Photon::Voice::Unity::Recorder, [70](#)
- Resample< T >
  - Photon::Voice::AudioUtil, [16](#)
- ResampleAndConvert
  - Photon::Voice::AudioUtil, [16](#), [17](#)
- Resampler
  - Photon::Voice::AudioUtil::Resampler, [73](#)
- ResetAccumAvgPeakAmp
  - Photon::Voice::AudioUtil::ILevelMeter, [34](#)
  - Photon::Voice::AudioUtil::LevelMeter, [42](#)
  - Photon::Voice::AudioUtil::LevelMeterDummy, [42](#)
- RestartRecording
  - Photon::Voice::Unity::Recorder, [68](#)
- RestrictedLowDelay
  - POpusCodec::Enums, [10](#)
- RoundTripTime
  - Photon::Voice::VoiceClient, [84](#)
- RoundTripTimeVariance
  - Photon::Voice::VoiceClient, [84](#)
- SamplingRate
  - Photon::Voice::AudioUtil::ToneAudioReader, [78](#)
  - Photon::Voice::IAudioDesc, [29](#)
  - Photon::Voice::Unity::Recorder, [70](#)
  - Photon::Voice::VoiceInfo, [94](#)
- SendDebugEchoVoicesInfo
  - Photon::Voice::LoadBalancingTransport, [45](#)
- Service
  - Photon::Voice::BufferReaderPushAdapter, [18](#)
  - Photon::Voice::BufferReaderPushAdapterAsync↔
    - Pool, [18](#)
  - Photon::Voice::BufferReaderPushAdapterAsync↔
    - PoolCopy, [19](#)
  - Photon::Voice::BufferReaderPushAdapterAsync↔
    - PoolFloatToShort, [20](#)
  - Photon::Voice::BufferReaderPushAdapterBase, [21](#)
  - Photon::Voice::IServiceable, [39](#)
  - Photon::Voice::LoadBalancingTransport, [45](#)
  - Photon::Voice::VoiceClient, [83](#)
- SetCallback
  - Photon::Voice::AudioUtil::ToneAudioPusher, [76](#)
  - Photon::Voice::IAudioPusher, [30](#)
- SetOutput
  - Photon::Voice::RemoteVoiceOptions, [73](#)
- Settings
  - Photon::Voice::Unity::VoiceConnection, [86](#)
- SetupDebugSpeaker
  - Photon::Voice::PUN::PhotonVoiceView, [64](#)
- SignalHint
  - POpusCodec::Enums, [10](#)
- SoloAmbient
  - Photon::Voice::IOS, [6](#)
- SourceType
  - Photon::Voice::Unity::Recorder, [70](#)
- Speaker, [74](#)
- SpeakerFactory
  - Photon::Voice::Unity::VoiceConnection, [86](#)
- SpeakerInUse
  - Photon::Voice::PUN::PhotonVoiceView, [65](#)
- SpeakerLinked
  - Photon::Voice::Unity::VoiceConnection, [87](#)
- SpeakerPrefab
  - Photon::Voice::Unity::VoiceConnection, [87](#)
- StartRecording
  - Photon::Voice::Unity::Recorder, [68](#)
- Stereo
  - POpusCodec::Enums, [9](#)
- StopRecording
  - Photon::Voice::Unity::Recorder, [68](#)
- SuperWideband
  - POpusCodec::Enums, [9](#)
- SuppressInfoDuplicateWarning
  - Photon::Voice::VoiceClient, [84](#)
- TestTone, [75](#)
- Threshold
  - Photon::Voice::AudioUtil::IVoiceDetector, [40](#)
  - Photon::Voice::AudioUtil::VoiceDetector, [89](#)
- ToneAudioPusher
  - Photon::Voice::AudioUtil::ToneAudioPusher, [76](#)
- ToneAudioReader, [76](#)
  - Photon::Voice::AudioUtil::ToneAudioReader, [77](#)
- TransmitEnabled
  - Photon::Voice::LocalVoice, [48](#)
  - Photon::Voice::Unity::Recorder, [70](#)
- TypeConvert
  - Photon::Voice::Unity::Recorder, [70](#)
- UnityAndroidAudioInAEC, [78](#)
- UnityAudioOut, [78](#)
- UnityMicrophoneDevice
  - Photon::Voice::Unity::Recorder, [70](#)
- UnsupportedCodecException, [79](#)
  - Photon::Voice::UnsupportedCodecException, [79](#)
- UnsupportedSampleTypeException, [79](#)
  - Photon::Voice::UnsupportedSampleTypeException, [80](#)
- UsePrimaryRecorder
  - Photon::Voice::PUN::PhotonVoiceView, [64](#)
- UserData

- Photon::Voice::Unity::Recorder, [70](#)
- Photon::Voice::VoiceInfo, [95](#)
- VideoChat
  - Photon::Voice::IOS, [7](#)
- VideoRecording
  - Photon::Voice::IOS, [7](#)
- VoIP
  - Photon::Voice::IOS::AudioSessionParameters↔  
Presets, [13](#)
- Voice
  - POpusCodec::Enums, [10](#)
- VoiceChat
  - Photon::Voice::IOS, [7](#)
- VoiceClient, [80](#)
  - Photon::Voice::LoadBalancingTransport, [46](#)
  - Photon::Voice::Unity::VoiceConnection, [87](#)
- VoiceComponent, [84](#)
- VoiceConnection, [84](#)
- VoiceDetection
  - Photon::Voice::Unity::Recorder, [70](#)
- VoiceDetectionDelayMs
  - Photon::Voice::Unity::Recorder, [70](#)
- VoiceDetectionThreshold
  - Photon::Voice::Unity::Recorder, [71](#)
- VoiceDetector
  - Photon::Voice::AudioUtil::VoiceLevelDetect↔  
Calibrate, [96](#)
  - Photon::Voice::ILocalVoiceAudio, [35](#)
  - Photon::Voice::Unity::Recorder, [71](#)
- VoiceDetectorCalibrate
  - Photon::Voice::ILocalVoiceAudio, [35](#)
  - Photon::Voice::LocalVoiceAudio, [49](#)
  - Photon::Voice::LocalVoiceAudioDummy, [50](#)
  - Photon::Voice::Unity::Recorder, [68](#)
- VoiceDetectorCalibrating
  - Photon::Voice::ILocalVoiceAudio, [35](#)
  - Photon::Voice::LocalVoiceAudio, [50](#)
  - Photon::Voice::Unity::Recorder, [71](#)
- VoiceDetectorCalibration
  - Photon::Voice::AudioUtil::VoiceDetectorCalibration,  
[90](#)
- VoiceDetectorFloat
  - Photon::Voice::AudioUtil::VoiceDetectorFloat, [91](#)
- VoiceDetectorShort
  - Photon::Voice::AudioUtil::VoiceDetectorShort, [92](#)
- VoiceEvent, [92](#)
- VoiceId
  - Photon::Voice::RemoteVoiceInfo, [71](#)
- VoiceInfo, [93](#)
- VoiceLevelDetectCalibrate
  - Photon::Voice::AudioUtil::VoiceLevelDetect↔  
Calibrate, [95](#)
- VoiceLogger, [96](#)
- VoiceRoomNameSuffix
  - Photon::Voice::PUN::PhotonVoiceNetwork, [62](#)
- Voip
  - POpusCodec::Enums, [10](#)
- WebRTCAudioLib, [97](#)
- WebRTCAudioLib.ConfigParam, [22](#)
- WebRTCAudioLib.Param, [60](#)
- WebRTCAudioProcessor, [98](#)
- WebRtcAudioDsp, [97](#)
- Wideband
  - POpusCodec::Enums, [9](#)
- Width
  - Photon::Voice::VoiceInfo, [95](#)