

Photon Voice

v2.24.1

Generated by Doxygen 1.8.17



<b>1 Main Page</b>	<b>1</b>
<b>2 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 AudioSampleType	6
2.2.1.2 Codec	6
2.3 Photon.Voice.IOS Namespace Reference	6
2.3.1 Enumeration Type Documentation	6
2.3.1.1 AudioSessionCategory	6
2.3.1.2 AudioSessionCategoryOption	7
2.3.1.3 AudioSessionMode	8
2.4 Photon.Voice.PUN Namespace Reference	9
2.5 Photon.Voice.PUN.UtilityScripts Namespace Reference	9
2.6 Photon.Voice.Unity Namespace Reference	9
2.7 Photon.Voice.Unity.UtilityScripts Namespace Reference	10
2.8 POpusCodec Namespace Reference	10
2.9 POpusCodec.Enums Namespace Reference	11
2.9.1 Enumeration Type Documentation	11
2.9.1.1 Bandwidth	11
2.9.1.2 Channels	11
2.9.1.3 Delay	11
2.9.1.4 OpusApplicationType	12
2.9.1.5 SignalHint	12
<b>3 Class Documentation</b>	<b>13</b>
3.1 AndroidAudioInAEC Class Reference	13
3.2 AudioClipWrapper Class Reference	13
3.3 AudioDesc Class Reference	14
3.4 AudioInChangeNotifier Class Reference	14
3.5 AudioInEnumerator Class Reference	14
3.5.1 Detailed Description	15
3.6 AudioOutCapture Class Reference	15
3.7 AudioSessionParameters Struct Reference	15
3.8 AudioSessionParametersPresets Class Reference	16
3.8.1 Member Data Documentation	16
3.8.1.1 Game	16
3.8.1.2 VoIP	16
3.9 AudioSyncBuffer< T > Class Template Reference	16
3.10 AudioUtil Class Reference	17
3.10.1 Detailed Description	18
3.10.2 Member Function Documentation	18

3.10.2.1 Convert() [1/2]	18
3.10.2.2 Convert() [2/2]	18
3.10.2.3 ForceToStereo< T >()	19
3.10.2.4 Resample< T >()	19
3.10.2.5 ResampleAndConvert() [1/2]	20
3.10.2.6 ResampleAndConvert() [2/2]	20
3.11 BufferReaderPushAdapter< T > Class Template Reference	20
3.11.1 Detailed Description	21
3.11.2 Constructor & Destructor Documentation	21
3.11.2.1 BufferReaderPushAdapter()	21
3.11.3 Member Function Documentation	21
3.11.3.1 Service()	21
3.12 BufferReaderPushAdapterAsyncPool< T > Class Template Reference	22
3.12.1 Detailed Description	22
3.12.2 Constructor & Destructor Documentation	22
3.12.2.1 BufferReaderPushAdapterAsyncPool()	22
3.12.3 Member Function Documentation	23
3.12.3.1 Service()	23
3.13 BufferReaderPushAdapterAsyncPoolCopy< T > Class Template Reference	23
3.13.1 Detailed Description	23
3.13.2 Constructor & Destructor Documentation	23
3.13.2.1 BufferReaderPushAdapterAsyncPoolCopy()	23
3.13.3 Member Function Documentation	24
3.13.3.1 Service()	24
3.14 BufferReaderPushAdapterAsyncPoolFloatToShort Class Reference	24
3.14.1 Detailed Description	24
3.14.2 Constructor & Destructor Documentation	25
3.14.2.1 BufferReaderPushAdapterAsyncPoolFloatToShort()	25
3.14.3 Member Function Documentation	25
3.14.3.1 Service()	25
3.15 BufferReaderPushAdapterAsyncPoolShortToFloat Class Reference	25
3.15.1 Detailed Description	26
3.15.2 Constructor & Destructor Documentation	26
3.15.2.1 BufferReaderPushAdapterAsyncPoolShortToFloat()	26
3.15.3 Member Function Documentation	26
3.15.3.1 Service()	26
3.16 BufferReaderPushAdapterBase< T > Class Template Reference	27
3.16.1 Detailed Description	27
3.16.2 Constructor & Destructor Documentation	27
3.16.2.1 BufferReaderPushAdapterBase()	27
3.16.3 Member Function Documentation	27
3.16.3.1 Dispose()	28

3.16.3.2 Service()	28
3.17 ConnectAndJoin Class Reference	28
3.18 RawCodec.Decoder< T > Class Template Reference	29
3.18.1 Member Function Documentation	29
3.18.1.1 Input()	29
3.18.1.2 Open()	29
3.19 OpusCodec.Decoder< T > Class Template Reference	30
3.19.1 Member Function Documentation	30
3.19.1.1 Input()	30
3.19.1.2 Open()	30
3.20 OpusCodec.DecoderFactory Class Reference	31
3.21 RawCodec.Encoder< T > Class Template Reference	31
3.22 OpusCodec.Encoder< T > Class Template Reference	31
3.23 OpusCodec.EncoderFloat Class Reference	32
3.24 OpusCodec.EncoderShort Class Reference	32
3.25 OpusCodec.Factory Class Reference	32
3.26 FactoryPrimitiveArrayPool< T > Class Template Reference	33
3.26.1 Detailed Description	33
3.27 FactoryReusableArray< T > Class Template Reference	33
3.27.1 Detailed Description	34
3.28 FrameBuffer Struct Reference	34
3.29 FrameOut< T > Class Template Reference	34
3.30 Framer< T > Class Template Reference	35
3.30.1 Detailed Description	35
3.30.2 Constructor & Destructor Documentation	35
3.30.2.1 Framer()	35
3.30.3 Member Function Documentation	35
3.30.3.1 Count()	35
3.30.3.2 Frame()	36
3.31 IAudioDesc Interface Reference	36
3.31.1 Detailed Description	36
3.31.2 Property Documentation	36
3.31.2.1 Channels	37
3.31.2.2 Error	37
3.31.2.3 SamplingRate	37
3.32 IAudioOut< T > Interface Template Reference	37
3.33 IAudioPusher< T > Interface Template Reference	37
3.33.1 Detailed Description	38
3.33.2 Member Function Documentation	38
3.33.2.1 SetCallback()	38
3.34 IAudioReader< T > Interface Template Reference	38
3.34.1 Detailed Description	38

3.35 IDataReader< T > Interface Template Reference . . . . .	39
3.35.1 Detailed Description . . . . .	39
3.35.2 Member Function Documentation . . . . .	39
3.35.2.1 Read() . . . . .	39
3.36 IDecoder Interface Reference . . . . .	39
3.36.1 Detailed Description . . . . .	40
3.36.2 Member Function Documentation . . . . .	40
3.36.2.1 Input() . . . . .	40
3.36.2.2 Open() . . . . .	40
3.36.3 Property Documentation . . . . .	41
3.36.3.1 Error . . . . .	41
3.37 IDecoderDirect< B > Interface Template Reference . . . . .	41
3.37.1 Detailed Description . . . . .	41
3.38 IDecoderQueuedOutputImageNative Interface Reference . . . . .	41
3.39 IEncoder Interface Reference . . . . .	41
3.39.1 Detailed Description . . . . .	42
3.39.2 Member Function Documentation . . . . .	42
3.39.2.1 DequeueOutput() . . . . .	42
3.39.2.2 EndOfStream() . . . . .	42
3.39.3 Property Documentation . . . . .	42
3.39.3.1 Error . . . . .	43
3.39.3.2 Output . . . . .	43
3.40 IEncoderDirect< B > Interface Template Reference . . . . .	43
3.40.1 Detailed Description . . . . .	43
3.40.2 Member Function Documentation . . . . .	43
3.40.2.1 Input() . . . . .	43
3.41 IEncoderDirectImage Interface Reference . . . . .	44
3.41.1 Detailed Description . . . . .	44
3.42 AudioUtil.ILevelMeter Interface Reference . . . . .	44
3.42.1 Detailed Description . . . . .	44
3.42.2 Member Function Documentation . . . . .	45
3.42.2.1 ResetAccumAvgPeakAmp() . . . . .	45
3.42.3 Property Documentation . . . . .	45
3.42.3.1 AccumAvgPeakAmp . . . . .	45
3.42.3.2 CurrentAvgAmp . . . . .	45
3.42.3.3 CurrentPeakAmp . . . . .	45
3.43 ILocalVoiceAudio Interface Reference . . . . .	45
3.43.1 Detailed Description . . . . .	46
3.43.2 Member Function Documentation . . . . .	46
3.43.2.1 VoiceDetectorCalibrate() . . . . .	46
3.43.3 Property Documentation . . . . .	46
3.43.3.1 LevelMeter . . . . .	47

3.43.3.2 VoiceDetector	47
3.43.3.3 VoiceDetectorCalibrating	47
3.44 ILoggable Interface Reference	47
3.45 ILoggableDependent Interface Reference	47
3.46 ILogger Interface Reference	47
3.47 ImageBufferInfo Class Reference	48
3.48 ImageBufferNative Class Reference	48
3.49 ImageBufferNativeAlloc Class Reference	48
3.50 ImageBufferNativeGCHandleSinglePlane Class Reference	49
3.51 ImageBufferNativePool< T > Class Template Reference	49
3.52 ImageOutputBuf Struct Reference	49
3.53 IOSAudioForceToSpeaker Class Reference	50
3.54 IProcessor< T > Interface Template Reference	50
3.54.1 Detailed Description	50
3.54.2 Member Function Documentation	50
3.54.2.1 Process()	50
3.55 IResettable Interface Reference	51
3.56 IServiceable Interface Reference	51
3.56.1 Detailed Description	51
3.56.2 Member Function Documentation	51
3.56.2.1 Service()	51
3.57 AudioUtil.IVoiceDetector Interface Reference	51
3.57.1 Detailed Description	52
3.57.2 Property Documentation	52
3.57.2.1 ActivityDelayMs	52
3.57.2.2 Detected	52
3.57.2.3 DetectedTime	52
3.57.2.4 On	53
3.57.2.5 Threshold	53
3.57.3 Event Documentation	53
3.57.3.1 OnDetected	53
3.58 IVoiceTransport Interface Reference	53
3.59 AudioUtil.LevelMeter< T > Class Template Reference	53
3.59.1 Detailed Description	54
3.59.2 Member Function Documentation	54
3.59.2.1 Process()	54
3.59.2.2 ResetAccumAvgPeakAmp()	55
3.60 AudioUtil.LevelMeterDummy Class Reference	55
3.60.1 Detailed Description	55
3.60.2 Member Function Documentation	55
3.60.2.1 ResetAccumAvgPeakAmp()	55
3.61 AudioUtil.LevelMeterFloat Class Reference	56

3.61.1 Detailed Description	56
3.61.2 Constructor & Destructor Documentation	56
3.61.2.1 LevelMeterFloat()	56
3.62 AudioUtil.LevelMeterShort Class Reference	56
3.62.1 Detailed Description	57
3.62.2 Constructor & Destructor Documentation	57
3.62.2.1 LevelMeterShort()	57
3.63 LoadBalancingFrontend Class Reference	57
3.64 LoadBalancingTransport Class Reference	57
3.64.1 Detailed Description	58
3.64.2 Constructor & Destructor Documentation	58
3.64.2.1 LoadBalancingTransport()	58
3.64.3 Member Function Documentation	59
3.64.3.1 Dispose()	59
3.64.3.2 Service()	59
3.64.4 Property Documentation	59
3.64.4.1 GlobalInterestGroup	59
3.64.4.2 VoiceClient	59
3.65 LoadBalancingTransport2 Class Reference	60
3.65.1 Detailed Description	60
3.66 LocalVoice Class Reference	60
3.66.1 Detailed Description	61
3.66.2 Member Function Documentation	61
3.66.2.1 RemoveSelf()	61
3.66.3 Property Documentation	62
3.66.3.1 DebugEchoMode	62
3.66.3.2 Encrypt	62
3.66.3.3 FramesSent	62
3.66.3.4 FramesSentBytes	62
3.66.3.5 Info	62
3.66.3.6 InterestGroup	63
3.66.3.7 IsCurrentlyTransmitting	63
3.66.3.8 LocalUserServiceable	63
3.66.3.9 Reliable	63
3.66.3.10 TransmitEnabled	63
3.67 LocalVoiceAudio< T > Class Template Reference	63
3.67.1 Detailed Description	64
3.67.2 Member Function Documentation	64
3.67.2.1 Create()	64
3.67.2.2 VoiceDetectorCalibrate()	65
3.67.3 Property Documentation	65
3.67.3.1 VoiceDetectorCalibrating	65



3.68 LocalVoiceAudioDummy Class Reference	65
3.68.1 Detailed Description	66
3.68.2 Member Function Documentation	66
3.68.2.1 VoiceDetectorCalibrate()	66
3.68.3 Member Data Documentation	67
3.68.3.1 Dummy	67
3.69 LocalVoiceAudioFloat Class Reference	67
3.69.1 Detailed Description	67
3.70 LocalVoiceAudioShort Class Reference	67
3.70.1 Detailed Description	67
3.71 LocalVoiceFramed< T > Class Template Reference	67
3.71.1 Detailed Description	68
3.71.2 Member Function Documentation	69
3.71.2.1 AddPostProcessor()	69
3.71.2.2 AddPreProcessor()	69
3.71.2.3 ClearProcessors()	69
3.71.2.4 Dispose()	69
3.71.2.5 PushData()	70
3.71.2.6 PushDataAsync()	70
3.71.3 Property Documentation	70
3.71.3.1 PushDataAsyncReady	70
3.72 LocalVoiceFramedBase Class Reference	70
3.72.1 Detailed Description	70
3.72.2 Property Documentation	71
3.72.2.1 FrameSize	71
3.73 Logger Class Reference	71
3.74 MicAmplifier Class Reference	71
3.75 MicAmplifierFloat Class Reference	71
3.76 MicAmplifierShort Class Reference	72
3.77 MicWrapper Class Reference	72
3.78 MicWrapperPusher Class Reference	72
3.79 MonoPInvokeCallbackAttribute Class Reference	73
3.80 NativeAndroidMicrophoneSettings Struct Reference	73
3.81 ObjectFactory< TType, TInfo > Interface Template Reference	73
3.81.1 Detailed Description	73
3.82 ObjectPool< TType, TInfo > Class Template Reference	74
3.82.1 Detailed Description	74
3.82.2 Constructor & Destructor Documentation	75
3.82.2.1 ObjectPool() [1/2]	75
3.82.2.2 ObjectPool() [2/2]	75
3.82.3 Member Function Documentation	75
3.82.3.1 AcquireOrCreate() [1/2]	76

3.82.3.2 AcquireOrCreate() [2/2]	76
3.82.3.3 Dispose()	76
3.82.3.4 Init()	76
3.82.3.5 Release() [1/2]	76
3.82.3.6 Release() [2/2]	77
3.82.4 Property Documentation	77
3.82.4.1 Info	77
3.83 OpusCodec Class Reference	77
3.84 OpusDecoder< T > Class Template Reference	78
3.85 OpusEncoder Class Reference	78
3.85.1 Property Documentation	79
3.85.1.1 EncoderDelay	79
3.86 OpusException Class Reference	79
3.87 OpusLib Class Reference	79
3.88 PhotonVoiceCreatedParams Class Reference	80
3.89 Recorder.PhotonVoiceCreatedParams Class Reference	80
3.90 PhotonVoiceLagSimulationGui Class Reference	80
3.91 PhotonVoiceNetwork Class Reference	80
3.91.1 Detailed Description	81
3.91.2 Member Function Documentation	81
3.91.2.1 ConnectAndJoinRoom()	82
3.91.2.2 Disconnect()	82
3.91.3 Member Data Documentation	82
3.91.3.1 AutoConnectAndJoin	82
3.91.3.2 AutoLeaveAndDisconnect	82
3.91.3.3 VoiceRoomNameSuffix	82
3.91.3.4 WorkInOfflineMode	83
3.91.4 Property Documentation	83
3.91.4.1 Instance	83
3.91.4.2 UsePunAuthValues	83
3.92 PhotonVoiceStatsGui Class Reference	83
3.92.1 Detailed Description	83
3.93 PhotonVoiceView Class Reference	83
3.93.1 Detailed Description	84
3.93.2 Member Function Documentation	85
3.93.2.1 Init()	85
3.93.3 Member Data Documentation	85
3.93.3.1 AutoCreateRecorderIfNotFound	85
3.93.3.2 SetupDebugSpeaker	85
3.93.3.3 UsePrimaryRecorder	85
3.93.4 Property Documentation	85
3.93.4.1 IsPhotonViewReady	86

3.93.4.2 IsRecorder	86
3.93.4.3 IsRecording	86
3.93.4.4 IsSetup	86
3.93.4.5 IsSpeaker	86
3.93.4.6 IsSpeakerLinked	86
3.93.4.7 IsSpeaking	87
3.93.4.8 RecorderInUse	87
3.93.4.9 SpeakerInUse	87
3.94 Platform Class Reference	87
3.95 PlaybackDelaySettings Struct Reference	87
3.95.1 Detailed Description	88
3.95.2 Member Data Documentation	88
3.95.2.1 MaxDelayHard	88
3.95.2.2 MaxDelaySoft	88
3.95.2.3 MinDelaySoft	88
3.96 UnityAudioOut.PlayDelayConfig Struct Reference	88
3.97 PrimitiveArrayPool< T > Class Template Reference	89
3.97.1 Detailed Description	89
3.98 RawCodec Class Reference	89
3.99 Recorder Class Reference	89
3.99.1 Detailed Description	92
3.99.2 Member Function Documentation	92
3.99.2.1 Init() [1/2]	92
3.99.2.2 Init() [2/2]	93
3.99.2.3 ResetLocalAudio()	93
3.99.2.4 RestartRecording()	93
3.99.2.5 StartRecording()	94
3.99.2.6 StopRecording()	94
3.99.2.7 VoiceDetectorCalibrate()	94
3.99.3 Property Documentation	94
3.99.3.1 AudioClip	94
3.99.3.2 AudioGroup	94
3.99.3.3 AutoStart	95
3.99.3.4 Bitrate	95
3.99.3.5 DebugEchoMode	95
3.99.3.6 Encrypt	95
3.99.3.7 FrameDuration	95
3.99.3.8 InputFactory	95
3.99.3.9 InterestGroup	96
3.99.3.10 IsCurrentlyTransmitting	96
3.99.3.11 IsInitialized	96
3.99.3.12 IsRecording	96

3.99.3.13 LevelMeter . . . . .	96
3.99.3.14 LoopAudioClip . . . . .	96
3.99.3.15 MicrophoneType . . . . .	97
3.99.3.16 PhotonMicrophoneDeviceId . . . . .	97
3.99.3.17 PhotonMicrophoneEnumerator . . . . .	97
3.99.3.18 ReactOnSystemChanges . . . . .	97
3.99.3.19 RecordOnlyWhenEnabled . . . . .	97
3.99.3.20 ReliableMode . . . . .	97
3.99.3.21 RequiresRestart . . . . .	98
3.99.3.22 SamplingRate . . . . .	98
3.99.3.23 SkipDeviceChangeChecks . . . . .	98
3.99.3.24 SourceType . . . . .	98
3.99.3.25 StopRecordingWhenPaused . . . . .	98
3.99.3.26 TransmitEnabled . . . . .	98
3.99.3.27 TrySamplingRateMatch . . . . .	99
3.99.3.28 TypeConvert . . . . .	99
3.99.3.29 UnityMicrophoneDevice . . . . .	99
3.99.3.30 UseMicrophoneTypeFallback . . . . .	99
3.99.3.31 UseOnAudioFilterRead . . . . .	99
3.99.3.32 UserData . . . . .	99
3.99.3.33 VoiceDetection . . . . .	100
3.99.3.34 VoiceDetectionDelayMs . . . . .	100
3.99.3.35 VoiceDetectionThreshold . . . . .	100
3.99.3.36 VoiceDetector . . . . .	100
3.99.3.37 VoiceDetectorCalibrating . . . . .	100
3.100 RemoteVoiceInfo Class Reference . . . . .	100
3.100.1 Detailed Description . . . . .	101
3.100.2 Property Documentation . . . . .	101
3.100.2.1 ChannelId . . . . .	101
3.100.2.2 Info . . . . .	101
3.100.2.3 playerId . . . . .	101
3.100.2.4 VoiceId . . . . .	101
3.101 RemoteVoiceLink Class Reference . . . . .	102
3.102 RemoteVoiceOptions Struct Reference . . . . .	102
3.102.1 Detailed Description . . . . .	102
3.102.2 Member Function Documentation . . . . .	102
3.102.2.1 SetOutput() . . . . .	103
3.102.3 Property Documentation . . . . .	103
3.102.3.1 Decoder . . . . .	103
3.102.3.2 OnRemoteVoiceRemoveAction . . . . .	103
3.103 AudioUtil.Resampler< T > Class Template Reference . . . . .	103
3.103.1 Detailed Description . . . . .	104

3.103.2 Constructor & Destructor Documentation	104
3.103.2.1 Resampler()	104
3.103.3 Member Function Documentation	104
3.103.3.1 Process()	104
3.104 SaveIncomingStreamToFile Class Reference	105
3.105 SaveOutgoingStreamToFile Class Reference	105
3.106 Speaker Class Reference	105
3.106.1 Detailed Description	106
3.106.2 Member Function Documentation	106
3.106.2.1 RestartPlayback()	106
3.106.2.2 SetPlaybackDelaySettings() [1/2]	106
3.106.2.3 SetPlaybackDelaySettings() [2/2]	107
3.106.2.4 StartPlayback()	107
3.106.2.5 StopPlayback()	107
3.106.3 Property Documentation	108
3.106.3.1 Actor	108
3.106.3.2 IsLinked	108
3.106.3.3 IsPlaying	108
3.106.3.4 Lag	108
3.106.3.5 OnRemoteVoiceRemoveAction	108
3.106.3.6 PlaybackDelayMaxHard	109
3.106.3.7 PlaybackDelayMaxSoft	109
3.106.3.8 PlaybackDelayMinSoft	109
3.106.3.9 PlaybackOnlyWhenEnabled	109
3.106.3.10 PlaybackStarted	109
3.107 TestTone Class Reference	109
3.108 AudioUtil.ToneAudioPusher< T > Class Template Reference	109
3.108.1 Detailed Description	110
3.108.2 Constructor & Destructor Documentation	110
3.108.2.1 ToneAudioPusher()	110
3.108.3 Member Function Documentation	110
3.108.3.1 SetCallback()	111
3.109 AudioUtil.ToneAudioReader< T > Class Template Reference	111
3.109.1 Detailed Description	111
3.109.2 Constructor & Destructor Documentation	112
3.109.2.1 ToneAudioReader()	112
3.109.3 Member Function Documentation	112
3.109.3.1 Read()	112
3.109.4 Property Documentation	112
3.109.4.1 Channels	113
3.109.4.2 Error	113
3.109.4.3 SamplingRate	113

3.110 ToneAudioReader Class Reference	113
3.111 UnityAudioOut Class Reference	113
3.112 UnityMicrophone Class Reference	114
3.112.1 Detailed Description	114
3.113 UnsupportedCodecException Class Reference	114
3.113.1 Detailed Description	115
3.113.2 Constructor & Destructor Documentation	115
3.113.2.1 UnsupportedCodecException()	115
3.114 UnsupportedSampleTypeException Class Reference	115
3.114.1 Detailed Description	115
3.114.2 Constructor & Destructor Documentation	116
3.114.2.1 UnsupportedSampleTypeException()	116
3.115 OpusCodec.Util Class Reference	116
3.116 VoiceClient Class Reference	116
3.116.1 Detailed Description	117
3.116.2 Member Function Documentation	117
3.116.2.1 CreateLocalVoice()	117
3.116.2.2 CreateLocalVoiceAudioFromSource()	118
3.116.2.3 CreateLocalVoiceFramed< T >()	118
3.116.2.4 LocalVoicesInChannel()	120
3.116.2.5 RemoteVoiceInfoDelegate()	120
3.116.2.6 RemoveLocalVoice()	120
3.116.2.7 Service()	121
3.116.3 Property Documentation	121
3.116.3.1 DebugLostPercent	121
3.116.3.2 FramesLost	121
3.116.3.3 FramesReceived	121
3.116.3.4 FramesSent	121
3.116.3.5 FramesSentBytes	122
3.116.3.6 LocalVoices	122
3.116.3.7 OnRemoteVoiceInfoAction	122
3.116.3.8 RemoteVoiceInfos	122
3.116.3.9 RoundTripTime	122
3.116.3.10 RoundTripTimeVariance	122
3.116.3.11 SuppressInfoDuplicateWarning	123
3.117 VoiceComponent Class Reference	123
3.118 VoiceConnection Class Reference	123
3.118.1 Detailed Description	125
3.118.2 Member Function Documentation	125
3.118.2.1 ConnectUsingSettings()	125
3.118.2.2 Dispatch()	126
3.118.2.3 InitRecorder()	126

3.118.2.4 SetGlobalPlaybackDelaySettings()	126
3.118.2.5 SetPlaybackDelaySettings()	126
3.118.3 Member Data Documentation	127
3.118.3.1 AutoCreateSpeakerIfNotFound	127
3.118.3.2 MinimalTimeScaleToDispatchInFixedUpdate	127
3.118.3.3 Settings	127
3.118.3.4 SpeakerFactory	127
3.118.4 Property Documentation	127
3.118.4.1 BestRegionSummaryInPreferences	128
3.118.4.2 ClientState	128
3.118.4.3 FramesLostPercent	128
3.118.4.4 FramesLostPerSecond	128
3.118.4.5 FramesReceivedPerSecond	128
3.118.4.6 GlobalPlaybackDelayMaxHard	128
3.118.4.7 GlobalPlaybackDelayMaxSoft	129
3.118.4.8 GlobalPlaybackDelayMinSoft	129
3.118.4.9 Logger	129
3.118.4.10 LogLevel	129
3.118.4.11 PrimaryRecorder	129
3.118.4.12 SpeakerPrefab	129
3.118.4.13 VoiceClient	130
3.118.5 Event Documentation	130
3.118.5.1 RemoteVoiceAdded	130
3.118.5.2 SpeakerLinked	130
3.119 VoiceDebugScript Class Reference	130
3.119.1 Detailed Description	131
3.119.2 Member Data Documentation	131
3.119.2.1 DisableVad	131
3.119.2.2 ForceRecordingAndTransmission	131
3.119.2.3 IncreaseLogLevels	131
3.119.2.4 LocalDebug	131
3.119.2.5 TestAudioClip	131
3.119.2.6 TestUsingAudioClip	132
3.120 AudioUtil.VoiceDetector< T > Class Template Reference	132
3.120.1 Detailed Description	133
3.120.2 Member Function Documentation	133
3.120.2.1 Process()	133
3.120.3 Property Documentation	133
3.120.3.1 ActivityDelayMs	133
3.120.3.2 Detected	133
3.120.3.3 DetectedTime	134
3.120.3.4 On	134

3.120.3.5 Threshold	134
3.120.4 Event Documentation	134
3.120.4.1 OnDetected	134
3.121 AudioUtil.VoiceDetectorCalibration< T > Class Template Reference	134
3.121.1 Detailed Description	135
3.121.2 Constructor & Destructor Documentation	135
3.121.2.1 VoiceDetectorCalibration()	135
3.121.3 Member Function Documentation	135
3.121.3.1 Calibrate()	135
3.121.3.2 Process()	136
3.122 AudioUtil.VoiceDetectorDummy Class Reference	136
3.122.1 Detailed Description	136
3.123 AudioUtil.VoiceDetectorFloat Class Reference	137
3.123.1 Detailed Description	137
3.123.2 Constructor & Destructor Documentation	137
3.123.2.1 VoiceDetectorFloat()	137
3.124 AudioUtil.VoiceDetectorShort Class Reference	137
3.124.1 Detailed Description	138
3.124.2 Constructor & Destructor Documentation	138
3.124.2.1 VoiceDetectorShort()	138
3.125 VoiceEvent Class Reference	138
3.125.1 Member Data Documentation	138
3.125.1.1 Code	138
3.126 VoiceInfo Struct Reference	139
3.126.1 Detailed Description	139
3.126.2 Member Function Documentation	140
3.126.2.1 CreateAudio()	140
3.126.2.2 CreateAudioOpus()	140
3.126.3 Property Documentation	141
3.126.3.1 Bitrate	141
3.126.3.2 Channels	141
3.126.3.3 FPS	141
3.126.3.4 FrameDurationSamples	141
3.126.3.5 FrameDurationUs	141
3.126.3.6 FrameSize	142
3.126.3.7 Height	142
3.126.3.8 KeyFrameInt	142
3.126.3.9 SamplingRate	142
3.126.3.10 UserData	142
3.126.3.11 Width	142
3.127 AudioUtil.VoiceLevelDetectCalibrate< T > Class Template Reference	143
3.127.1 Detailed Description	143



---

3.127.2 Constructor & Destructor Documentation . . . . .	143
3.127.2.1 VoiceLevelDetectCalibrate() . . . . .	143
3.127.3 Member Function Documentation . . . . .	144
3.127.3.1 Calibrate() . . . . .	144
3.127.3.2 Process() . . . . .	144
3.127.4 Property Documentation . . . . .	144
3.127.4.1 LevelMeter . . . . .	144
3.127.4.2 VoiceDetector . . . . .	145
3.128 VoiceLogger Class Reference . . . . .	145
3.129 WebRtcAudioDsp Class Reference . . . . .	145
3.129.1 Member Function Documentation . . . . .	146
3.129.1.1 SetOrSwitchAudioListener() . . . . .	146
3.129.1.2 SetOrSwitchAudioOutCapture() . . . . .	146
3.130 WebRTCAudioLib Class Reference . . . . .	147
3.131 WebRTCAudioProcessor Class Reference . . . . .	147
<b>Index</b>	<b>149</b>



# Chapter 1

## Main Page

Photon Voice 2 has three key classes:

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

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

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

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

Read more in the official documentation [here](#).

You can download Photon Voice 2 [here](#).



## Chapter 2

# Namespace Documentation

## 2.1 Photon Namespace Reference

## 2.2 Photon.Voice Namespace Reference

### Classes

- class [AudioDesc](#)
- class [AudioInChangeNotifier](#)
- class [AudioInEnumerator](#)  
*Enumerates microphones available on device.*
- class [AudioSyncBuffer](#)
- class [AudioUtil](#)  
*Collection of Audio Utility functions and classes.*
- class [BufferReaderPushAdapter](#)  
*Simple [BufferReaderPushAdapterBase](#) implementation using a single buffer, using synchronous [LocalVoice.PushData](#)*
- class [BufferReaderPushAdapterAsyncPool](#)  
*[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#).*
- class [BufferReaderPushAdapterAsyncPoolCopy](#)  
*[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#) and data copy.*
- class [BufferReaderPushAdapterAsyncPoolFloatToShort](#)  
*[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#), converting float samples to short.*
- class [BufferReaderPushAdapterAsyncPoolShortToFloat](#)  
*[BufferReaderPushAdapter](#) implementation using asynchronous [LocalVoice.PushDataAsync](#), converting short samples to float.*
- class [BufferReaderPushAdapterBase](#)  
*Adapter base class to move data by reading from [IDataReader.Read](#) and pushing to [LocalVoice](#).*
- class [FactoryPrimitiveArrayPool](#)  
*[PrimitiveArrayPool<T>](#) as wrapped in object factory interface.*
- class [FactoryReusableArray](#)  
*Array factory returnig the same array instance as long as it requested with the same array length. If length changes, new array instance created.*
- struct [FrameBuffer](#)
- class [FrameOut](#)
- class [Framer](#)

- Utility class to re-frame audio packets.*
- interface [IAudioDesc](#)
  - Audio Source interface.*
- interface [IAudioOut](#)
- interface [IAudioPusher](#)
  - Audio Pusher interface.*
- interface [IAudioReader](#)
  - Audio Reader interface.*
- interface [IDataReader](#)
  - Interface for pulling data, in case this is more appropriate than pushing it.*
- interface [IDecoder](#)
  - Generic decoder interface.*
- interface [IDecoderDirect](#)
  - Interface for an decoder which outputs data via explicit call.*
- interface [IDecoderQueuedOutputImageNative](#)
- interface [IEncoder](#)
  - Generic encoder interface.*
- interface [IEncoderDirect](#)
  - Interface for an encoder which consumes input data via explicit call.*
- interface [IEncoderDirectImage](#)
  - Interface for an encoder which consumes images via explicit call.*
- interface [ILocalVoiceAudio](#)
  - Interface for an outgoing audio stream.*
- interface [ILogger](#)
- class [ImageBufferInfo](#)
- class [ImageBufferNative](#)
- class [ImageBufferNativeAlloc](#)
- class [ImageBufferNativeGCHandleSinglePlane](#)
- class [ImageBufferNativePool](#)
- struct [ImageOutputBuf](#)
- interface [IProcessor](#)
  - Audio Processor interface.*
- interface [IResettable](#)
- interface [IServiceable](#)
  - Interface for classes that want their [Service\(\)](#) function to be called regularly in the context of a [LocalVoice](#).*
- interface [IVoiceTransport](#)
- class [LoadBalancingFrontend](#)
- class [LoadBalancingTransport](#)
  - Extends [LoadBalancingClient](#) with media streaming functionality.*
- class [LoadBalancingTransport2](#)
  - Variant of [LoadBalancingTransport](#). Aims to be non-alloc at the cost of breaking compatibility with older clients.*
- class [LocalVoice](#)
  - Represents outgoing data stream.*
- class [LocalVoiceAudio](#)
  - Outgoing audio stream.*
- class [LocalVoiceAudioDummy](#)
  - Dummy [LocalVoiceAudio](#)*
- class [LocalVoiceAudioFloat](#)
  - Specialization of [LocalVoiceAudio](#) for float audio*
- class [LocalVoiceAudioShort](#)
  - Specialization of [LocalVoiceAudio](#) for short audio*
- class [LocalVoiceFramed](#)

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

## Enumerations

- enum [AudioSampleType](#)
  - The type of samples used for audio processing.*
- enum **FrameFlags** : byte
- enum [Codec](#)
  - Enum for Media Codecs supported by PhotonVoice.*
- enum **ImageFormat**
- enum **Rotation**
- enum **Flip**

### 2.2.1 Enumeration Type Documentation

### 2.2.1.1 AudioSampleType

```
enum AudioSampleType [strong]
```

The type of samples used for audio processing.

### 2.2.1.2 Codec

```
enum Codec [strong]
```

Enum for Media Codecs supported by PhotonVoice.

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

Enumerator

AudioOpus	OPUS audio
-----------	------------

## 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 AudioSessionCategory

```
enum AudioSessionCategory [strong]
```

Enumerator

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



## Enumerator

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 AudioSessionCategoryOption

```
enum AudioSessionCategoryOption [strong]
```

## Enumerator

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

## Enumerator

DuckOthers	This allows an application to set whether or not other active audio apps will be ducked when when your app's audio session goes active. An example of this is the Nike app, which provides periodic updates to its user (it reduces the volume of any music currently being played while it provides its status). This defaults to off. Note that the other audio will be ducked for as long as the current session is active. You will need to deactivate your audio session when you want full volume playback of the other audio. If your category is AVAudioSessionCategoryPlayback, AVAudioSessionCategoryPlayAndRecord, or AVAudioSessionCategoryMultiRoute, by default the audio session will be non-mixable and non-ducking. Setting this option will also make your category mixable with others (AVAudioSessionCategoryOptionMixWithOthers will be set). DuckOthers is only valid with AVAudioSessionCategoryAmbient, AVAudioSessionCategoryPlayAndRecord, AVAudioSessionCategoryPlayback, and AVAudioSessionCategoryMultiRoute
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 AudioSessionMode

```
enum AudioSessionMode [strong]
```

## Enumerator

Default	Modes modify the audio category in order to introduce behavior that is tailored to the specific use of audio within an application. Available in iOS 5.0 and greater. The default mode API_AVAILABLE(ios(5.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);
VoiceChat	Only valid with AVAudioSessionCategoryPlayAndRecord. Appropriate for Voice over IP (VoIP) applications. Reduces the number of allowable audio routes to be only those that are appropriate for VoIP applications and may engage appropriate system-supplied signal processing. Has the side effect of setting AVAudioSessionCategoryOptionAllowBluetooth API_AVAILABLE(ios(5.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);

## Enumerator

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.PUN.UtilityScripts Namespace Reference

### Classes

- class [VoiceDebugScript](#)

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

## 2.6 Photon.Voice.Unity Namespace Reference

### Classes

- class [AndroidAudioInAEC](#)
- class [AudioClipWrapper](#)

- class [AudioOutCapture](#)
- interface [ILoggable](#)
- interface [ILoggableDependent](#)
- class [IOSAudioForceToSpeaker](#)
- class [Logger](#)
- class [MicWrapper](#)
- class [MicWrapperPusher](#)
- struct [NativeAndroidMicrophoneSettings](#)
- class [PhotonVoiceCreatedParams](#)
- struct [PlaybackDelaySettings](#)
  - Playback delay configuration container.*
- class [Recorder](#)
  - Component representing outgoing audio stream in scene.*
- class [RemoteVoiceLink](#)
- class [Speaker](#)
  - Component representing remote audio stream in local scene.*
- class [UnityAudioOut](#)
- class [UnityMicrophone](#)
  - A wrapper around `UnityEngine.Microphone` to be able to safely use `Microphone` and compile for WebGL.*
- class [VoiceComponent](#)
- class [VoiceConnection](#)
  - Component that represents a client voice connection to [Photon](#) Servers.*
- class [VoiceLogger](#)
- class [WebRtcAudioDsp](#)

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

### Classes

- class [ConnectAndJoin](#)
- class [MicAmplifier](#)
- class [MicAmplifierFloat](#)
- class [MicAmplifierShort](#)
- class [PhotonVoiceLagSimulationGui](#)
- class [PhotonVoiceStatsGui](#)
  - Basic GUI to show traffic and health statistics of the connection to [Photon](#), toggled by shift+tab.*
- class [SaveIncomingStreamToFile](#)
- class [SaveOutgoingStreamToFile](#)
- class [TestTone](#)
- class [ToneAudioReader](#)

## 2.8 POpusCodec Namespace Reference

### Classes

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

## 2.9 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

### 2.9.1 Enumeration Type Documentation

#### 2.9.1.1 Bandwidth

```
enum Bandwidth : int [strong]
```

##### Enumerator

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

#### 2.9.1.2 Channels

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

##### Enumerator

Mono	1 Channel
Stereo	2 Channels

#### 2.9.1.3 Delay

```
enum Delay [strong]
```

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

#### Enumerator

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

### 2.9.1.4 OpusApplicationType

```
enum OpusApplicationType : int [strong]
```

#### Enumerator

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

### 2.9.1.5 SignalHint

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

#### Enumerator

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

## Chapter 3

# Class Documentation

### 3.1 AndroidAudioInAEC Class Reference

Inherits [IAudioPusher< short >](#), and [IResettable](#).

#### Public Member Functions

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

#### Properties

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

### 3.2 AudioClipWrapper Class Reference

Inherits [IAudioReader< float >](#).

#### Public Member Functions

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

## Properties

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

## 3.3 AudioDesc Class Reference

Inherits [IAudioDesc](#).

### Public Member Functions

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

## Properties

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

## 3.4 AudioInChangeNotifier Class Reference

Inherits [IDisposable](#).

### Public Member Functions

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

### Public Attributes

- readonly bool **IsSupported** = false

## Properties

- string **Error** [get]

## 3.5 AudioInEnumerator Class Reference

Enumerates microphones available on device.

Inherits [IDisposable](#).



## Public Member Functions

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

## Public Attributes

- readonly bool **IsSupported** = false

## Properties

- string **Error** [get]
- int **Count** [get]
- [IEnumerable< string >](#) **Names** [get]

### 3.5.1 Detailed Description

Enumerates microphones available on device.

## 3.6 AudioOutCapture Class Reference

Inherits [MonoBehaviour](#).

## Events

- [Action< float\[\], int >](#) **OnAudioFrame**

## 3.7 AudioSessionParameters Struct Reference

### Public Member Functions

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

### Public Attributes

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

## 3.8 AudioSessionParametersPresets Class Reference

### Static Public Attributes

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

### 3.8.1 Member Data Documentation

#### 3.8.1.1 Game

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

##### Initial value:

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

#### 3.8.1.2 VoIP

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

##### Initial value:

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

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

## 3.9 AudioSyncBuffer< T > Class Template Reference

Inherits [IAudioOut< T >](#).

### Public Member Functions

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

## Static Public Attributes

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

## Properties

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

## 3.10 AudioUtil Class Reference

Collection of Audio Utility functions and classes.

### Classes

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

## Static Public Member Functions

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

### 3.10.1 Detailed Description

Collection of Audio Utility functions and classes.

### 3.10.2 Member Function Documentation

#### 3.10.2.1 [Convert\(\)](#) [1/2]

```
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.

#### 3.10.2.2 [Convert\(\)](#) [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.

### 3.10.2.3 ForceToStereo< T >()

```

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

```

Convert audio buffer with arbitrary number of channels to stereo.

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

#### Parameters

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

### 3.10.2.4 Resample< T >()

```

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

```

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

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

#### Parameters

<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.

### 3.10.2.5 ResampleAndConvert() [1/2]

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

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

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

#### Parameters

<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.

### 3.10.2.6 ResampleAndConvert() [2/2]

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

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

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

#### Parameters

<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.

## 3.11 BufferReaderPushAdapter< T > Class Template Reference

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

Inherits [BufferReaderPushAdapterBase< T >](#).

## Public Member Functions

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

## Protected Attributes

- T[] [buffer](#)

### 3.11.1 Detailed Description

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

### 3.11.2 Constructor & Destructor Documentation

#### 3.11.2.1 BufferReaderPushAdapter()

```
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.

### 3.11.3 Member Function Documentation

#### 3.11.3.1 Service()

```
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 [BufferedReaderPushAdapterBase< T >](#).

## 3.12 BufferedReaderPushAdapterAsyncPool< T > Class Template Reference

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

Inherits [BufferedReaderPushAdapterBase< T >](#).

### Public Member Functions

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

### Additional Inherited Members

#### 3.12.1 Detailed Description

[BufferedReaderPushAdapter](#) 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.

#### 3.12.2 Constructor & Destructor Documentation

##### 3.12.2.1 BufferedReaderPushAdapterAsyncPool()

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

Create a new [BufferedReaderPushAdapter](#) instance

## Parameters

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



### 3.12.3 Member Function Documentation

#### 3.12.3.1 `Service()`

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

Do the actual data read/push.

Parameters

<code>localVoice</code>	<code>LocalVoice</code> instance to push data to. Must be a <code>LocalVoiceFramed&lt;T&gt;</code> of same T.
-------------------------	---

Implements `BufferReaderPushAdapterBase< T >`.

## 3.13 `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`

#### 3.13.1 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 call.

#### 3.13.2 Constructor & Destructor Documentation

##### 3.13.2.1 `BufferReaderPushAdapterAsyncPoolCopy()`

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

Create a new `BufferReaderPushAdapter` instance

## Parameters

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

### 3.13.3 Member Function Documentation

#### 3.13.3.1 Service()

```
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 >](#).

## 3.14 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

#### 3.14.1 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.

### 3.14.2 Constructor & Destructor Documentation

#### 3.14.2.1 BufferReaderPushAdapterAsyncPoolFloatToShort()

```
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>	DataReader to read from.

### 3.14.3 Member Function Documentation

#### 3.14.3.1 Service()

```
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 >](#).

## 3.15 BufferReaderPushAdapterAsyncPoolShortToFloat Class Reference

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

Inherits [BufferReaderPushAdapterBase< short >](#).

### Public Member Functions

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

## Additional Inherited Members

### 3.15.1 Detailed Description

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

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

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

### 3.15.2 Constructor & Destructor Documentation

#### 3.15.2.1 [BufferReaderPushAdapterAsyncPoolShortToFloat\(\)](#)

```
BufferReaderPushAdapterAsyncPoolShortToFloat (
    LocalVoice localVoice,
    IDataReader< short > 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.

### 3.15.3 Member Function Documentation

#### 3.15.3.1 [Service\(\)](#)

```
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< short >](#).

## 3.16 `BufferedReaderPushAdapterBase< T >` Class Template Reference

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

Inherits `IServiceable`.

Inherited by `BufferedReaderPushAdapter< T >`, `BufferedReaderPushAdapterAsyncPool< T >`, and `BufferedReaderPushAdapterAsyncPool`.

### Public Member Functions

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

### Protected Attributes

- `IDataReader< T >` `reader`

#### 3.16.1 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.

#### 3.16.2 Constructor & Destructor Documentation

##### 3.16.2.1 `BufferedReaderPushAdapterBase()`

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

Create a new `BufferedReaderPushAdapterBase` instance

Parameters

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

#### 3.16.3 Member Function Documentation

### 3.16.3.1 Dispose()

```
void Dispose ( )
```

Release resources associated with this instance.

### 3.16.3.2 Service()

```
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 [BufferedReaderPushAdapterAsyncPoolShortToFloat](#), [BufferedReaderPushAdapterAsyncPoolFloatToShort](#), [BufferedReaderPushAdapterAsyncPoolCopy< T >](#), [BufferedReaderPushAdapterAsyncPool< T >](#), and [BufferedReaderPushAdapter< T >](#).

## 3.17 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]

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

Inherits [IDecoder](#).

## Public Member Functions

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

## Properties

- string **Error** [get]

### 3.18.1 Member Function Documentation

#### 3.18.1.1 Input()

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

Consumes the given encoded data.

Implements [IDecoder](#).

#### 3.18.1.2 Open()

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

Open (initialize) the decoder.

## Parameters

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

Implements [IDecoder](#).

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

Inherits [IDecoder](#).

### Public Member Functions

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

### Protected Attributes

- [OpusDecoder](#)< T > **decoder**

### Properties

- string **Error** [get]

## 3.19.1 Member Function Documentation

### 3.19.1.1 Input()

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

Consumes the given encoded data.

Implements [IDecoder](#).

### 3.19.1.2 Open()

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

Open (initialize) the decoder.



## Parameters

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

Implements [IDecoder](#).

## 3.20 OpusCodec.DecoderFactory Class Reference

### Static Public Member Functions

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

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

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

### Public Member Functions

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

### Properties

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

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

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

### Public Member Functions

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

## Protected Member Functions

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

## Protected Attributes

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

## Properties

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

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

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

### Public Member Functions

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

## Properties

- `int Info` [get]

### 3.27.1 Detailed Description

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

#### Template Parameters

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

## 3.28 FrameBuffer Struct Reference

### Public Member Functions

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

### Public Attributes

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

## Properties

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

## 3.29 FrameOut< T > Class Template Reference

### Public Member Functions

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

## Properties

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

## 3.30 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.*

### 3.30.1 Detailed Description

Utility class to re-frame audio packets.

### 3.30.2 Constructor & Destructor Documentation

#### 3.30.2.1 Framer()

```
Framer (  
    int frameSize )
```

Create new [Framer](#) instance.

### 3.30.3 Member Function Documentation

#### 3.30.3.1 Count()

```
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.

**3.30.3.2 Frame()**

```
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).

**3.31 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.*

**3.31.1 Detailed Description**

Audio Source interface.

**3.31.2 Property Documentation**

### 3.31.2.1 Channels

```
int Channels [get]
```

Number of channels in the audio signal.

### 3.31.2.2 Error

```
string Error [get]
```

If not null, audio object is in invalid state.

### 3.31.2.3 SamplingRate

```
int SamplingRate [get]
```

Sampling rate of the audio signal (in Hz).

## 3.32 IAudioOut< T > Interface Template Reference

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

### Public Member Functions

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

### Properties

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

## 3.33 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

### 3.33.1 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,

### 3.33.2 Member Function Documentation

#### 3.33.2.1 SetCallback()

```
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 >](#).

## 3.34 IAudioReader< T > Interface Template Reference

Audio Reader interface.

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

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

## Additional Inherited Members

### 3.34.1 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).



## 3.35 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.*

#### 3.35.1 Detailed Description

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

#### 3.35.2 Member Function Documentation

##### 3.35.2.1 Read()

```
bool Read (
    T[] buffer )
```

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

##### Parameters

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

##### Returns

True if buffer was filled successfully, false otherwise.

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

## 3.36 IDecoder Interface Reference

Generic decoder interface.

Inherits IDisposable.

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

## Public Member Functions

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

## Properties

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

### 3.36.1 Detailed Description

Generic decoder interface.

### 3.36.2 Member Function Documentation

#### 3.36.2.1 Input()

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

Consumes the given encoded data.

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

#### 3.36.2.2 Open()

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

Open (initialize) the decoder.

##### Parameters

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

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

### 3.36.3 Property Documentation

#### 3.36.3.1 Error

`string Error [get]`

If not null, the object is in invalid state.

## 3.37 IDecoderDirect< B > Interface Template Reference

Interface for an decoder which outputs data via explicit call.

Inherits [IDecoder](#).

### Properties

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

### Additional Inherited Members

#### 3.37.1 Detailed Description

Interface for an decoder which outputs data via explicit call.

## 3.38 IDecoderQueuedOutputImageNative Interface Reference

Inherits [IDecoderDirect< ImageOutputBuf >](#).

### Properties

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

## 3.39 IEncoder Interface Reference

Generic encoder interface.

Inherits [IDisposable](#).

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

## Public Member Functions

- `ArraySegment< byte > DequeueOutput (out FrameFlags flags)`  
*Returns next encoded data frame (if such output supported).*
- `void EndOfStream ()`  
*Forces an encoder to flush and produce frame with EndOfStream flag (in output queue).*
- `I GetPlatformAPI< I > ()`

## Properties

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

### 3.39.1 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.

### 3.39.2 Member Function Documentation

#### 3.39.2.1 DequeueOutput()

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

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

#### 3.39.2.2 EndOfStream()

```
void EndOfStream ( )
```

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

### 3.39.3 Property Documentation

### 3.39.3.1 Error

```
string Error [get]
```

If not null, the object is in invalid state.

### 3.39.3.2 Output

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

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

## 3.40 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

#### 3.40.1 Detailed Description

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

#### 3.40.2 Member Function Documentation

##### 3.40.2.1 Input()

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

Consumes the given raw data.

## Parameters

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

## 3.41 IEncoderDirectImage Interface Reference

Interface for an encoder which consumes images via explicit call.

Inherits [IEncoderDirect< ImageBufferNative >](#).

### Properties

- ImageFormat **ImageFormat** [get]

### Additional Inherited Members

#### 3.41.1 Detailed Description

Interface for an encoder which consumes images via explicit call.

## 3.42 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.*

#### 3.42.1 Detailed Description

Audio Level Metering interface.

## 3.42.2 Member Function Documentation

### 3.42.2.1 ResetAccumAvgPeakAmp()

```
void ResetAccumAvgPeakAmp ( )
```

Reset [AccumAvgPeakAmp](#).

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

## 3.42.3 Property Documentation

### 3.42.3.1 AccumAvgPeakAmp

```
float AccumAvgPeakAmp [get]
```

Average of CurrentPeakAmps since last reset.

### 3.42.3.2 CurrentAvgAmp

```
float CurrentAvgAmp [get]
```

Average amplitude value over last half second.

### 3.42.3.3 CurrentPeakAmp

```
float CurrentPeakAmp [get]
```

Maximum amplitude value over last half second sec.

## 3.43 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.*

### 3.43.1 Detailed Description

Interface for an outgoing audio stream.

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

### 3.43.2 Member Function Documentation

#### 3.43.2.1 VoiceDetectorCalibrate()

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

Trigger voice detector calibration process.

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

### 3.43.3 Property Documentation



### 3.43.3.1 LevelMeter

[AudioUtil.ILoggable](#) LevelMeter [get]

The LevelMeter utility in use.

### 3.43.3.2 VoiceDetector

[AudioUtil.ILoggable](#) VoiceDetector [get]

The VoiceDetector in use.

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

### 3.43.3.3 VoiceDetectorCalibrating

bool VoiceDetectorCalibrating [get]

If true, voice detector calibration is in progress.

## 3.44 ILoggable Interface Reference

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

### Properties

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

## 3.45 ILoggableDependent Interface Reference

Inherits [ILoggable](#).

Inherited by [VoiceComponent](#).

### Properties

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

## 3.46 ILogger Interface Reference

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

## Public Member Functions

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

## 3.47 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]

## 3.48 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]

## 3.49 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

## 3.50 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

## 3.51 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

## 3.52 ImageOutputBuf Struct Reference

### Public Attributes

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

### 3.53 IOSAudioForceToSpeaker Class Reference

Inherits MonoBehaviour.

### 3.54 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.*

#### 3.54.1 Detailed Description

Audio Processor interface.

#### 3.54.2 Member Function Documentation

##### 3.54.2.1 Process()

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

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

## 3.55 IResettable Interface Reference

Inherited by [AndroidAudioInAEC](#).

### Public Member Functions

- void **Reset** ()

## 3.56 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.*

### 3.56.1 Detailed Description

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

### 3.56.2 Member Function Documentation

#### 3.56.2.1 Service()

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

Service function that should be called regularly.

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

## 3.57 AudioUtil.IVoiceDetector Interface Reference

[Voice](#) Activity Detector interface.

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

## Properties

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

## Events

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

### 3.57.1 Detailed Description

[Voice](#) Activity Detector interface.

### 3.57.2 Property Documentation

#### 3.57.2.1 ActivityDelayMs

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

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

#### 3.57.2.2 Detected

```
bool Detected [get]
```

If true, voice detected.

#### 3.57.2.3 DetectedTime

```
DateTime DetectedTime [get]
```

Last time when switched to detected state.

### 3.57.2.4 On

bool On [get], [set]

If true, voice detection enabled.

### 3.57.2.5 Threshold

float Threshold [get], [set]

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

## 3.57.3 Event Documentation

### 3.57.3.1 OnDetected

Action OnDetected

Called when switched to detected state.

## 3.58 IVoiceTransport Interface Reference

Inherited by [LoadBalancingTransport](#).

### Public Member Functions

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

## 3.59 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]

### 3.59.1 Detailed Description

Audio Level Meter.

### 3.59.2 Member Function Documentation

#### 3.59.2.1 Process()

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

Implements [IProcessor< T >](#).

#### 3.59.2.2 ResetAccumAvgPeakAmp()

```
void ResetAccumAvgPeakAmp ( )
```

Reset AccumAvgPeakAmp.

Implements [AudioUtil.ILevelMeter](#).

## 3.60 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]

### 3.60.1 Detailed Description

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

### 3.60.2 Member Function Documentation

#### 3.60.2.1 ResetAccumAvgPeakAmp()

```
void ResetAccumAvgPeakAmp ( )
```

Reset AccumAvgPeakAmp.

Implements [AudioUtil.ILevelMeter](#).

## 3.61 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

#### 3.61.1 Detailed Description

[LevelMeter](#) specialization for float audio.

#### 3.61.2 Constructor & Destructor Documentation

##### 3.61.2.1 LevelMeterFloat()

```
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.

## 3.62 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

### 3.62.1 Detailed Description

[LevelMeter](#) specialization for short audio.

### 3.62.2 Constructor & Destructor Documentation

#### 3.62.2.1 LevelMeterShort()

```
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.

## 3.63 LoadBalancingFrontend Class Reference

Inherits [LoadBalancingTransport](#).

## Additional Inherited Members

## 3.64 LoadBalancingTransport Class Reference

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

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

Inherited by [LoadBalancingFrontend](#), and [LoadBalancingTransport2](#).

## Public Member Functions

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

- [LoadBalancingTransport](#) ([ILogger](#) logger=null, [ConnectionProtocol](#) connectionProtocol=[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 [SendVoiceRemove](#) ([LocalVoice](#) voice, int channelId, int targetPlayerId)
- virtual void [SendFrame](#) (ArraySegment< byte > data, [FrameFlags](#) flags, byte evNumber, byte voiceId, int channelId, int targetPlayerId, bool reliable, [LocalVoice](#) localVoice)
- string [ChannelIdStr](#) (int channelId)
- string [PlayerIdStr](#) (int playerId)
- void [Dispose](#) ()  
*Releases all resources used by the [LoadBalancingTransport](#) instance.*

## Protected Member Functions

- virtual void [onEventActionVoiceClient](#) (EventData ev)

## Protected Attributes

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

## Properties

- [VoiceClient](#) [VoiceClient](#) [get]  
*The [VoiceClient](#) implementation associated with this [LoadBalancingTransport](#).*
- byte [GlobalAudioGroup](#) [get, set]
- byte [GlobalInterestGroup](#) [get, set]  
*Set global interest group for this client. This call sets [InterestGroup](#) for existing local voices and for created later to given value. Client set as listening to this group only until [LoadBalancingPeer.OpChangeGroups\(\)](#) called. This method can be called any time.*

### 3.64.1 Detailed Description

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

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

### 3.64.2 Constructor & Destructor Documentation

#### 3.64.2.1 [LoadBalancingTransport](#)()

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

Initializes a new [LoadBalancingTransport](#).

## Parameters

<i>logger</i>	<a href="#">ILogger</a> instance. If null, this instance <a href="#">LoadBalancingClient.DebugReturn</a> implementation is used. <a href="#">ConnectionProtocol</a>
<i>connectionProtocol</i>	Connection protocol (UDP or TCP). <a href="#">ConnectionProtocol</a>

### 3.64.3 Member Function Documentation

#### 3.64.3.1 Dispose()

```
void Dispose ( )
```

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

#### 3.64.3.2 Service()

```
new void Service ( )
```

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

### 3.64.4 Property Documentation

#### 3.64.4.1 GlobalInterestGroup

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

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

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

#### 3.64.4.2 VoiceClient

```
VoiceClient VoiceClient [get]
```

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

## 3.65 LoadBalancingTransport2 Class Reference

Variant of [LoadBalancingTransport](#). Aims to be non-alloc at the cost of breaking compatibility with older clients.

Inherits [LoadBalancingTransport](#).

### Public Member Functions

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

### Protected Member Functions

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

### Additional Inherited Members

#### 3.65.1 Detailed Description

Variant of [LoadBalancingTransport](#). Aims to be non-alloc at the cost of breaking compatibility with older clients.

## 3.66 LocalVoice Class Reference

Represents outgoing data stream.

Inherits [IDisposable](#).

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

### Public Member Functions

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

### Static Public Attributes

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

## Protected Attributes

- [VoiceInfo](#) **info**
- [IEncoder](#) **encoder**
- [VoiceClient](#) **voiceClient**
- `ArraySegment< byte >` **configFrame**
- volatile bool **disposed**
- object **disposeLock** = new object()

## Properties

- byte **Group** [get, set]
- byte [InterestGroup](#) [get, set]  
*If InterestGroup != 0, voice's data is sent only to clients listening to this group (if supported by transport).*
- [VoiceInfo](#) **Info** [get]  
*Returns Info structure assigned on local voice cration.*
- bool [TransmitEnabled](#) [get, set]  
*If true, stream data broadcasted.*
- bool [IsCurrentlyTransmitting](#) [get]  
*Returns true if stream broadcasts.*
- int [FramesSent](#) [get]  
*Sent frames counter.*
- int [FramesSentBytes](#) [get]  
*Sent frames bytes counter.*
- bool [Reliable](#) [get, set]  
*Send data reliable.*
- bool [Encrypt](#) [get, set]  
*Send data encrypted.*
- [IServiceable](#) [LocalUserServiceable](#) [get, set]  
*Optional user object attached to [LocalVoice](#). its Service() will be called at each [VoiceClient.Service\(\)](#) call.*
- bool [DebugEchoMode](#) [get, set]  
*If true, outgoing stream routed back to client via server same way as for remote client's streams. Can be swithed any time. OnRemoteVoiceInfoAction and OnRemoteVoiceRemoveAction are triggered if required. This functionality availability depends on transport.*

### 3.66.1 Detailed Description

Represents outgoing data stream.

### 3.66.2 Member Function Documentation

#### 3.66.2.1 RemoveSelf()

```
void RemoveSelf ( )
```

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

.

### 3.66.3 Property Documentation

#### 3.66.3.1 DebugEchoMode

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

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

#### 3.66.3.2 Encrypt

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

Send data encrypted.

#### 3.66.3.3 FramesSent

```
int FramesSent [get]
```

Sent frames counter.

#### 3.66.3.4 FramesSentBytes

```
int FramesSentBytes [get]
```

Sent frames bytes counter.

#### 3.66.3.5 Info

```
VoiceInfo Info [get]
```

Returns Info structure assigned on local voice cration.



### 3.66.3.6 InterestGroup

byte InterestGroup [get], [set]

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

### 3.66.3.7 IsCurrentlyTransmitting

bool IsCurrentlyTransmitting [get]

Returns true if stream broadcasts.

### 3.66.3.8 LocalUserServiceable

IServiceable LocalUserServiceable [get], [set]

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

### 3.66.3.9 Reliable

bool Reliable [get], [set]

Send data reliable.

### 3.66.3.10 TransmitEnabled

bool TransmitEnabled [get], [set]

If true, stream data broadcasted.

## 3.67 LocalVoiceAudio< T > Class Template Reference

Outgoing audio stream.

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

### Public Member Functions

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

## Static Public Member Functions

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

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

## Protected Member Functions

- void [initBuiltinProcessors](#) ()

## Protected Attributes

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

## Properties

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

*True if the VoiceDetector is currently calibrating.*

## Additional Inherited Members

### 3.67.1 Detailed Description

Outgoing audio stream.

### 3.67.2 Member Function Documentation

#### 3.67.2.1 Create()

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

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

## Parameters

<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.

**3.67.2.2 VoiceDetectorCalibrate()**

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

Trigger voice detector calibration process.

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

## Parameters

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

Implements [ILocalVoiceAudio](#).

**3.67.3 Property Documentation****3.67.3.1 VoiceDetectorCalibrating**

```
bool VoiceDetectorCalibrating [get]
```

True if the VoiceDetector is currently calibrating.

**3.68 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

### 3.68.1 Detailed Description

Dummy [LocalVoiceAudio](#)

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

### 3.68.2 Member Function Documentation

#### 3.68.2.1 VoiceDetectorCalibrate()

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

Trigger voice detector calibration process.

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

### 3.68.3 Member Data Documentation

#### 3.68.3.1 Dummy

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

A Dummy [LocalVoiceAudio](#) instance.

## 3.69 LocalVoiceAudioFloat Class Reference

Specialization of [LocalVoiceAudio](#) for float audio

Inherits [LocalVoiceAudio< float >](#).

### Additional Inherited Members

#### 3.69.1 Detailed Description

Specialization of [LocalVoiceAudio](#) for float audio

## 3.70 LocalVoiceAudioShort Class Reference

Specialization of [LocalVoiceAudio](#) for short audio

Inherits [LocalVoiceAudio< short >](#).

### Additional Inherited Members

#### 3.70.1 Detailed Description

Specialization of [LocalVoiceAudio](#) for short audio

## 3.71 LocalVoiceFramed< T > Class Template Reference

Typed re-framing [LocalVoice](#)

Inherits [LocalVoiceFramedBase](#).

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

## Public Member Functions

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

## Protected Member Functions

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

## Properties

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

## Additional Inherited Members

### 3.71.1 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>channel↔ Id</i>	Transport channel specific to transport.
<i>encoder</i>	Encoder producing the stream.

**Returns**

Outgoing stream handler.

## 3.71.2 Member Function Documentation

### 3.71.2.1 AddPostProcessor()

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

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

**Parameters**

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

### 3.71.2.2 AddPreProcessor()

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

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

**Parameters**

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

### 3.71.2.3 ClearProcessors()

```
void ClearProcessors ( )
```

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

### 3.71.2.4 Dispose()

```
override void Dispose ( ) [virtual]
```

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

Reimplemented from [LocalVoice](#).

### 3.71.2.5 PushData()

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

Synchronously push data into this stream.

### 3.71.2.6 PushDataAsync()

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

Asynchronously push data into this stream.

## 3.71.3 Property Documentation

### 3.71.3.1 PushDataAsyncReady

```
bool PushDataAsyncReady [get]
```

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

## 3.72 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

### 3.72.1 Detailed Description

Typed re-framing [LocalVoice](#)

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



## 3.72.2 Property Documentation

### 3.72.2.1 FrameSize

`int FrameSize [get]`

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

## 3.73 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)

## 3.74 MicAmplifier Class Reference

Inherits [VoiceComponent](#).

### Properties

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

### Additional Inherited Members

## 3.75 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]

## 3.76 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]

## 3.77 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]

## 3.78 MicWrapperPusher Class Reference

Inherits [IAudioPusher< float >](#).

## Public Member Functions

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

## Properties

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

## 3.79 MonoPInvokeCallbackAttribute Class Reference

Inherits Attribute.

## Public Member Functions

- **MonoPInvokeCallbackAttribute** (Type t)

## 3.80 NativeAndroidMicrophoneSettings Struct Reference

## Public Attributes

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

## 3.81 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]

### 3.81.1 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).

## 3.82 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.*

#### 3.82.1 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).

## 3.82.2 Constructor & Destructor Documentation

### 3.82.2.1 ObjectPool() [1/2]

```
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.

### 3.82.2.2 ObjectPool() [2/2]

```
ObjectPool (
    int capacity,
    string name,
    TInfo info )
```

Create a new [ObjectPool](#) instance with the given info structure. Calls [Init\(\)](#).

## Parameters

<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.

## 3.82.3 Member Function Documentation

### 3.82.3.1 AcquireOrCreate() [1/2]

```
TType AcquireOrCreate ( )
```

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

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

### 3.82.3.2 AcquireOrCreate() [2/2]

```
TType AcquireOrCreate (
    TInfo info )
```

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

#### Parameters

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

### 3.82.3.3 Dispose()

```
void Dispose ( )
```

Free resources assoicated with this [ObjectPool](#)

### 3.82.3.4 Init()

```
void Init (
    TInfo info )
```

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

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

#### Parameters

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

### 3.82.3.5 Release() [1/2]

```
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.
------------	-----------------------------------

### 3.82.3.6 Release() [2/2]

```
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.

## 3.82.4 Property Documentation

### 3.82.4.1 Info

```
TInfo Info [get]
```

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

## 3.83 OpusCodec Class Reference

### Classes

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

## Public Types

- enum **FrameDuration**

## Properties

- static string **Version** [get]

## 3.84 OpusDecoder< T > Class Template Reference

Inherits IDisposable.

### Public Member Functions

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

### Properties

- [Bandwidth](#)? **PreviousPacketBandwidth** [get]

## 3.85 OpusEncoder Class Reference

Inherits IDisposable.

### Public Member Functions

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

### Static Public Attributes

- const int **BitrateMax** = -1



## Properties

- SamplingRate **InputSamplingRate** [get]
- Channels **InputChannels** [get]
- Delay **EncoderDelay** [get, set]  
*Using a duration of less than 10 ms will prevent the encoder from using the LPC or hybrid modes.*
- int **FrameSizePerChannel** [get]
- int **Bitrate** [get, set]
- Bandwidth **MaxBandwidth** [get, set]
- Complexity **Complexity** [get, set]
- int **ExpectedPacketLossPercentage** [get, set]
- SignalHint **SignalHint** [get, set]
- ForceChannels **ForceChannels** [get, set]
- bool? **UseInbandFEC** [get, set]
- int **PacketLossPercentage** [get, set]
- bool? **UseUnconstrainedVBR** [get, set]
- bool? **DtxEnabled** [get, set]

## 3.85.1 Property Documentation

### 3.85.1.1 EncoderDelay

Delay **EncoderDelay** [get], [set]

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

## 3.86 OpusException Class Reference

Inherits Exception.

## Public Member Functions

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

## Properties

- OpusStatusCode **Statuscode** [get]

## 3.87 OpusLib Class Reference

## Properties

- static string **Version** [get]

## 3.88 PhotonVoiceCreatedParams Class Reference

Inherited by [Recorder.PhotonVoiceCreatedParams](#).

### Properties

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

## 3.89 Recorder.PhotonVoiceCreatedParams Class Reference

Inherits [PhotonVoiceCreatedParams](#).

### Additional Inherited Members

## 3.90 PhotonVoiceLagSimulationGui Class Reference

Inherits [MonoBehaviour](#).

### Public Member Functions

- void **OnEnable** ()

## 3.91 PhotonVoiceNetwork Class Reference

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

Inherits [VoiceConnection](#).

### Public Member Functions

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

## Public Attributes

- bool [AutoConnectAndJoin](#) = true  
*Auto connect voice client and join a voice room when [PUN](#) client is joined to a [PUN](#) room*
- bool [AutoLeaveAndDisconnect](#) = true  
*Auto disconnect voice client when [PUN](#) client is not joined to a [PUN](#) room*
- bool [WorkInOfflineMode](#) = true  
*Whether or not [Photon Voice](#) client should follow [PUN](#) client if the latter is in offline mode.*

## Static Public Attributes

- const string [VoiceRoomNameSuffix](#) = "\_\_voice\_"  
*Suffix for voice room names appended to [PUN](#) room names.*

## Protected Member Functions

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

## Properties

- static [PhotonVoiceNetwork](#) [Instance](#) [get, set]  
*Singleton instance for [PhotonVoiceNetwork](#)*
- bool [UsePunAuthValues](#) [get, set]  
*Whether or not to use the same [PhotonNetwork.AuthValues](#) in [PhotonVoiceNetwork.Instance.Client.AuthValues](#).*

## Additional Inherited Members

### 3.91.1 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.

### 3.91.2 Member Function Documentation

### 3.91.2.1 ConnectAndJoinRoom()

```
bool ConnectAndJoinRoom ( )
```

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

#### Returns

If true, connection command send from client

### 3.91.2.2 Disconnect()

```
void Disconnect ( )
```

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

## 3.91.3 Member Data Documentation

### 3.91.3.1 AutoConnectAndJoin

```
bool AutoConnectAndJoin = true
```

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

### 3.91.3.2 AutoLeaveAndDisconnect

```
bool AutoLeaveAndDisconnect = true
```

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

### 3.91.3.3 VoiceRoomNameSuffix

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

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

### 3.91.3.4 WorkInOfflineMode

```
bool WorkInOfflineMode = true
```

Whether or not [Photon Voice](#) client should follow [PUN](#) client if the latter is in offline mode.

## 3.91.4 Property Documentation

### 3.91.4.1 Instance

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

Singleton instance for [PhotonVoiceNetwork](#)

### 3.91.4.2 UsePunAuthValues

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

Whether or not to use the same [PhotonNetwork.AuthValues](#) in [PhotonVoiceNetwork.Instance.Client.AuthValues](#).

## 3.92 PhotonVoiceStatsGui Class Reference

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

Inherits [MonoBehaviour](#).

### 3.92.1 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).

## 3.93 PhotonVoiceView Class Reference

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

Inherits [VoiceComponent](#).

## Public Member Functions

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

## Public Attributes

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

## Protected Member Functions

- override void **Awake** ()

## Properties

- [Recorder RecorderInUse](#) [get, set]  
*The [Recorder](#) component currently used by this [PhotonVoiceView](#)*
- [Speaker SpeakerInUse](#) [get, set]  
*The [Speaker](#) component currently used by this [PhotonVoiceView](#)*
- bool [IsSetup](#) [get]  
*If true, this [PhotonVoiceView](#) is setup and ready to be used*
- bool [IsSpeaker](#) [get]  
*If true, this [PhotonVoiceView](#) has a [Speaker](#) setup for playback of received audio frames from remote audio source*
- bool [IsSpeaking](#) [get]  
*If true, this [PhotonVoiceView](#) has a [Speaker](#) that is currently playing received audio frames from remote audio source*
- bool [IsRecorder](#) [get]  
*If true, this [PhotonVoiceView](#) has a [Recorder](#) setup for transmission of audio stream from local audio source*
- bool [IsRecording](#) [get]  
*If true, this [PhotonVoiceView](#) has a [Recorder](#) that is currently transmitting audio stream from local audio source*
- bool [IsSpeakerLinked](#) [get]  
*If true, the [SpeakerInUse](#) is linked to the remote voice stream*
- bool [IsPhotonViewReady](#) [get]  
*If true, the [PhotonView](#) attached to the same [GameObject](#) has a valid [ViewID](#) > 0*

## Additional Inherited Members

### 3.93.1 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.

## 3.93.2 Member Function Documentation

### 3.93.2.1 Init()

```
void Init ( )
```

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

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

## 3.93.3 Member Data Documentation

### 3.93.3.1 AutoCreateRecorderIfNotFound

```
bool AutoCreateRecorderIfNotFound
```

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

### 3.93.3.2 SetupDebugSpeaker

```
bool SetupDebugSpeaker
```

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

### 3.93.3.3 UsePrimaryRecorder

```
bool UsePrimaryRecorder
```

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

## 3.93.4 Property Documentation

#### 3.93.4.1 IsPhotonViewReady

```
bool IsPhotonViewReady [get]
```

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

#### 3.93.4.2 IsRecorder

```
bool IsRecorder [get]
```

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

#### 3.93.4.3 IsRecording

```
bool IsRecording [get]
```

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

#### 3.93.4.4 IsSetup

```
bool IsSetup [get]
```

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

#### 3.93.4.5 IsSpeaker

```
bool IsSpeaker [get]
```

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

#### 3.93.4.6 IsSpeakerLinked

```
bool IsSpeakerLinked [get]
```

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



### 3.93.4.7 IsSpeaking

```
bool IsSpeaking [get]
```

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

### 3.93.4.8 RecorderInUse

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

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

### 3.93.4.9 SpeakerInUse

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

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

## 3.94 Platform Class Reference

### Static Public Member Functions

- static [IEncoder](#) [CreateDefaultAudioEncoder](#)< T > ([ILogger](#) logger, [VoiceInfo](#) info)

## 3.95 PlaybackDelaySettings Struct Reference

Playback delay configuration container.

### Public Member Functions

- override string [ToString](#) ()

### Public Attributes

- int [MinDelaySoft](#)  
*ms: Audio player tries to keep the delay above this value.*
- int [MaxDelaySoft](#)  
*ms: Audio player tries to keep the delay below this value.*
- int [MaxDelayHard](#)  
*ms: Audio player guarantees that the delay never exceeds this value.*

## Static Public Attributes

- const int **DEFAULT\_LOW** = 200
- const int **DEFAULT\_HIGH** = 400
- const int **DEFAULT\_MAX** = 1000

### 3.95.1 Detailed Description

Playback delay configuration container.

### 3.95.2 Member Data Documentation

#### 3.95.2.1 MaxDelayHard

```
int MaxDelayHard
```

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

#### 3.95.2.2 MaxDelaySoft

```
int MaxDelaySoft
```

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

#### 3.95.2.3 MinDelaySoft

```
int MinDelaySoft
```

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

## 3.96 UnityAudioOut.PlayDelayConfig Struct Reference

### Static Public Attributes

- static [PlayDelayConfig](#) **Default** = new [PlayDelayConfig](#) { Low = 200, High = 400, Max = 1000 }

## Properties

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

## 3.97 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

### 3.97.1 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.
----------	---------------------

## 3.98 RawCodec Class Reference

### Classes

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

## 3.99 Recorder Class Reference

Component representing outgoing audio stream in scene.

Inherits [VoiceComponent](#).

## Classes

- class [PhotonVoiceCreatedParams](#)

## Public Types

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

## Public Member Functions

- void [Init](#) ([VoiceClient](#) voiceClient, object customObj=null)  
*Initializes the [Recorder](#) component to be able to transmit audio.*
- void [Init](#) ([VoiceConnection](#) voiceConnection)  
*Initializes the [Recorder](#) component to be able to transmit audio.*
- void **Relnit** ()
- void [RestartRecording](#) (bool force=false)  
*Restarts recording if something has changed that requires this.*
- void [VoiceDetectorCalibrate](#) (int durationMs, Action< float > detectionEndedCallback=null)  
*Trigger voice detector calibration process. While calibrating, keep silence. [Voice](#) detector sets threshold basing on measured background noise level.*
- void [StartRecording](#) ()  
*Starts recording.*
- void [StopRecording](#) ()  
*Stops recording.*
- bool [ResetLocalAudio](#) ()  
*Resets audio session and parameters locally to fix broken recording due to system configuration modifications or audio interruptions or audio routing changes.*

## Static Public Member Functions

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

## Static Public Attributes

- const int **MIN\_OPUS\_BITRATE** = 6000
- const int **MAX\_OPUS\_BITRATE** = 510000

## Protected Member Functions

- virtual void **SendPhotonVoiceCreatedMessage** ()

## Properties

- static [AudioInEnumerator PhotonMicrophoneEnumerator](#) [get]  
*Enumerator for the available microphone devices gathered by the [Photon](#) plugin.*
- bool [IsInitialized](#) [get]  
*If true, this [Recorder](#) has been initialized and is ready to transmit to remote clients. Otherwise call [Init\(VoiceConnection\)](#).*
- bool [RequiresInit](#) [get]
- bool [RequiresRestart](#) [get, protected set]  
*Returns true if something has changed in the [Recorder](#) while recording that won't take effect unless recording is restarted using [RestartRecording](#).*
- bool [TransmitEnabled](#) [get, set]  
*If true, audio transmission is enabled.*
- bool [Encrypt](#) [get, set]  
*If true, voice stream is sent encrypted.*
- bool [DebugEchoMode](#) [get, set]  
*If true, outgoing stream routed back to client via server same way as for remote client's streams.*
- bool [ReliableMode](#) [get, set]  
*If true, stream data sent in reliable mode.*
- bool [VoiceDetection](#) [get, set]  
*If true, voice detection enabled.*
- float [VoiceDetectionThreshold](#) [get, set]  
*Voice detection threshold (0..1, where 1 is full amplitude).*
- int [VoiceDetectionDelayMs](#) [get, set]  
*Keep detected state during this time after signal level dropped below threshold. Default is 500ms*
- object [UserData](#) [get, set]  
*Custom user object to be sent in the voice stream info event.*
- Func< [IAudioDesc](#) > [InputFactory](#) [get, set]  
*Set the method returning new [Voice.IAudioDesc](#) instance to be assigned to a new voice created with Source set to Factory*
- [AudioUtil.IVoiceDetector?](#) [VoiceDetector](#) [get]  
*Returns voice activity detector for recorder's audio stream.*
- string [UnityMicrophoneDevice](#) [get, set]  
*Set or get [Unity](#) microphone device used for streaming.*
- int [PhotonMicrophoneDeviceId](#) [get, set]  
*Set or get photon microphone device used for streaming.*
- byte [AudioGroup](#) [get, set]  
*Target interest group that will receive transmitted audio.*
- byte [InterestGroup](#) [get, set]  
*Target interest group that will receive transmitted audio.*
- bool [IsCurrentlyTransmitting](#) [get]  
*Returns true if audio stream broadcasts.*
- [AudioUtil.ILevelMeter?](#) [LevelMeter](#) [get]  
*Level meter utility.*
- bool [VoiceDetectorCalibrating](#) [get]  
*If true, voice detector calibration is in progress.*
- [ILocalVoiceAudio](#) [voiceAudio](#) [get]
- InputSourceType [SourceType](#) [get, set]  
*Audio data source.*
- MicType [MicrophoneType](#) [get, set]  
*Which microphone API to use when the Source is set to Microphone.*
- SampleTypeConv [TypeConvert](#) [get, set]

- Force creation of 'short' pipeline and convert audio data to short for 'float' audio sources.
- AudioClip [AudioClip](#) [get, set]  
Source audio clip.
- bool [LoopAudioClip](#) [get, set]  
Loop playback for audio clip sources.
- SamplingRate [SamplingRate](#) [get, set]  
Outgoing audio stream sampling rate.
- OpusCodec.FrameDuration [FrameDuration](#) [get, set]  
Outgoing audio stream encoder delay.
- int [Bitrate](#) [get, set]  
Outgoing audio stream bitrate.
- bool [IsRecording](#) [get, set]  
Gets or sets whether this [Recorder](#) is actively recording audio to be transmitted.
- bool [ReactOnSystemChanges](#) [get, set]  
If true, the [Recorder](#) will automatically restart recording to recover from audio device changes.
- bool [AutoStart](#) [get, set]  
If true, automatically start recording when initialized.
- bool [RecordOnlyWhenEnabled](#) [get, set]  
If true, component will work only when enabled and active in hierarchy.
- bool [SkipDeviceChangeChecks](#) [get, set]  
If true, restarts recording without checking if audio config/device changes affected recording.
- bool [StopRecordingWhenPaused](#) [get, set]  
If true, stop recording when paused resume/restart when un-paused.
- bool [UseOnAudioFilterRead](#) [get, set]  
If true, recording will make use of [Unity](#)'s OnAudioFiltterRead callback from a muted local AudioSource.
- bool [TrySamplingRateMatch](#) [get, set]  
If true, [Recorder](#) will try to match sampling rates of microphone device and Opus encoder to avoid re sampling of audio input.
- bool [UseMicrophoneTypeFallback](#) [get, set]  
If true, if recording fails to start with [Unity](#) microphone type, [Photon](#) microphone type is used -if available- as a fallback and vice versa.

## Additional Inherited Members

### 3.99.1 Detailed Description

Component representing outgoing audio stream in scene.

### 3.99.2 Member Function Documentation

#### 3.99.2.1 Init() [1/2]

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

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

## Parameters

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

**3.99.2.2 Init()** [2/2]

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

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

## Parameters

<i>voiceConnection</i>	The <a href="#">VoiceConnection</a> to be used with this <a href="#">Recorder</a> .
------------------------	---

**3.99.2.3 ResetLocalAudio()**

```
bool ResetLocalAudio ( )
```

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

## Returns

If reset is done.

**3.99.2.4 RestartRecording()**

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

Restarts recording if something has changed that requires this.

## Parameters

<i>force</i>	Set to true if you want to restart even if this is not required (RequiresRestart = false)
--------------	---

### 3.99.2.5 StartRecording()

```
void StartRecording ( )
```

Starts recording.

### 3.99.2.6 StopRecording()

```
void StopRecording ( )
```

Stops recording.

### 3.99.2.7 VoiceDetectorCalibrate()

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

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

#### Parameters

<i>durationMs</i>	Duration of calibration in milliseconds.
<i>detectionEndedCallback</i>	Callback when VAD calibration ends.

## 3.99.3 Property Documentation

### 3.99.3.1 AudioClip

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

Source audio clip.

### 3.99.3.2 AudioGroup

```
byte AudioGroup [get], [set]
```

Target interest group that will receive transmitted audio.

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



### 3.99.3.3 AutoStart

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

If true, automatically start recording when initialized.

### 3.99.3.4 Bitrate

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

Outgoing audio stream bitrate.

### 3.99.3.5 DebugEchoMode

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

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

### 3.99.3.6 Encrypt

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

If true, voice stream is sent encrypted.

### 3.99.3.7 FrameDuration

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

Outgoing audio stream encoder delay.

### 3.99.3.8 InputFactory

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

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

### 3.99.3.9 InterestGroup

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

Target interest group that will receive transmitted audio.

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

### 3.99.3.10 IsCurrentlyTransmitting

```
bool IsCurrentlyTransmitting [get]
```

Returns true if audio stream broadcasts.

### 3.99.3.11 IsInitialized

```
bool IsInitialized [get]
```

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

### 3.99.3.12 IsRecording

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

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

### 3.99.3.13 LevelMeter

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

Level meter utility.

### 3.99.3.14 LoopAudioClip

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

Loop playback for audio clip sources.

### 3.99.3.15 MicrophoneType

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

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

### 3.99.3.16 PhotonMicrophoneDeviceId

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

Set or get photon microphone device used for streaming.

### 3.99.3.17 PhotonMicrophoneEnumerator

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

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

### 3.99.3.18 ReactOnSystemChanges

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

If true, the [Recorder](#) will automatically restart recording to recover from audio device changes.

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

### 3.99.3.19 RecordOnlyWhenEnabled

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

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

### 3.99.3.20 ReliableMode

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

If true, stream data sent in reliable mode.

### 3.99.3.21 RequiresRestart

```
bool RequiresRestart [get], [protected set]
```

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

Think of this as a "isDirty" flag.

### 3.99.3.22 SamplingRate

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

Outgoing audio stream sampling rate.

### 3.99.3.23 SkipDeviceChangeChecks

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

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

To be used when [Recorder.ReactOnSystemChanges](#) is true.

### 3.99.3.24 SourceType

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

Audio data source.

### 3.99.3.25 StopRecordingWhenPaused

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

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

### 3.99.3.26 TransmitEnabled

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

If true, audio transmission is enabled.

### 3.99.3.27 TrySamplingRateMatch

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

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

### 3.99.3.28 TypeConvert

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

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

### 3.99.3.29 UnityMicrophoneDevice

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

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

### 3.99.3.30 UseMicrophoneTypeFallback

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

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

### 3.99.3.31 UseOnAudioFilterRead

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

If true, recording will make use of [Unity](#)'s OnAudioFiltlerRead callback from a muted local AudioSource.

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

### 3.99.3.32 UserData

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

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

### 3.99.3.33 VoiceDetection

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

If true, voice detection enabled.

### 3.99.3.34 VoiceDetectionDelayMs

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

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

### 3.99.3.35 VoiceDetectionThreshold

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

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

### 3.99.3.36 VoiceDetector

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

Returns voice activity detector for recorder's audio stream.

### 3.99.3.37 VoiceDetectorCalibrating

```
bool VoiceDetectorCalibrating [get]
```

If true, voice detector calibration is in progress.

## 3.100 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).*

### 3.100.1 Detailed Description

Information about a remote voice (incoming stream).

### 3.100.2 Property Documentation

#### 3.100.2.1 ChannelId

```
int ChannelId [get]
```

ID of channel used for transmission.

#### 3.100.2.2 Info

```
VoiceInfo Info [get]
```

Remote voice info.

#### 3.100.2.3 PlayerId

```
int PlayerId [get]
```

Player ID of voice owner.

#### 3.100.2.4 VoiceId

```
byte VoiceId [get]
```

[Voice](#) ID (unique in the room).

## 3.101 RemoteVoiceLink Class Reference

### Public Member Functions

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

### Properties

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

### Events

- Action< [FrameOut](#)< float > > **FloatFrameDecoded**
- Action **RemoteVoiceRemoved**

## 3.102 RemoteVoiceOptions Struct Reference

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

### Public Member Functions

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

### Properties

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

### 3.102.1 Detailed Description

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

### 3.102.2 Member Function Documentation



### 3.102.2.1 SetOutput()

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

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

## 3.102.3 Property Documentation

### 3.102.3.1 Decoder

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

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

### 3.102.3.2 OnRemoteVoiceRemoveAction

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

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

## 3.103 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](#)

### 3.103.1 Detailed Description

Sample-rate conversion Audio Processor.

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

### 3.103.2 Constructor & Destructor Documentation

#### 3.103.2.1 Resampler()

```
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.

### 3.103.3 Member Function Documentation

#### 3.103.3.1 Process()

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

Implements `IProcessor< T >`.

## 3.104 SaveIncomingStreamToFile Class Reference

Inherits [VoiceComponent](#).

### Protected Member Functions

- override void **Awake** ()

### Additional Inherited Members

## 3.105 SaveOutgoingStreamToFile Class Reference

Inherits [VoiceComponent](#).

### Additional Inherited Members

## 3.106 Speaker Class Reference

Component representing remote audio stream in local scene.

Inherits [VoiceComponent](#).

### Public Member Functions

- bool [StartPlayback](#) ()  
*Starts the audio playback of the linked incoming remote audio stream via AudioSource component.*
- bool [StopPlayback](#) ()  
*Stops the audio playback of the linked incoming remote audio stream via AudioSource component.*
- bool [RestartPlayback](#) ()  
*Restarts the audio playback of the linked incoming remote audio stream via AudioSource component.*
- bool [SetPlaybackDelaySettings](#) ([PlaybackDelaySettings](#) pdc)  
*Sets the settings for the playback behaviour in case of delays.*
- bool [SetPlaybackDelaySettings](#) (int low, int high, int max)  
*Sets the settings for the playback behaviour in case of delays.*

## Properties

- int **PlayDelayMs** [get, set]
- bool **IsPlaying** [get]  
*Is the speaker playing right now.*
- int? **Lag** [get]  
*Smoothed difference between (jittering) stream and (clock-driven) audioOutput.*
- Action< **Speaker** > **OnRemoteVoiceRemoveAction** [get, set]  
*Register a method to be called when remote voice removed.*
- Realtime.Player **Actor** [get, set]  
*Per room, the connected users/players are represented with a Realtime.Player, also known as Actor.*
- bool **IsLinked** [get]  
*Whether or not this **Speaker** has been linked to a remote voice stream.*
- bool **PlaybackOnlyWhenEnabled** [get, set]  
*If true, component will work only when enabled and active in hierarchy.*
- bool **PlaybackStarted** [get]  
*Returns if the playback is on.*
- int **PlaybackDelayMinSoft** [get]  
*Gets the value in ms above which the audio player tries to keep the delay.*
- int **PlaybackDelayMaxSoft** [get]  
*Gets the value in ms below which the audio player tries to keep the delay.*
- int **PlaybackDelayMaxHard** [get]  
*Gets the value in ms that audio play delay will not exceed.*

## Additional Inherited Members

### 3.106.1 Detailed Description

Component representing remote audio stream in local scene.

### 3.106.2 Member Function Documentation

#### 3.106.2.1 RestartPlayback()

```
bool RestartPlayback ( )
```

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

##### Returns

True if playback is successfully restarted.

#### 3.106.2.2 SetPlaybackDelaySettings() [1/2]

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

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

**Parameters**

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

**Returns**

If a change has been made.

**3.106.2.3 SetPlaybackDelaySettings() [2/2]**

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

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

**Parameters**

<i>pdc</i>	Playback delay configuration struct.
------------	--------------------------------------

**Returns**

If a change has been made.

**3.106.2.4 StartPlayback()**

```
bool StartPlayback ( )
```

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

**Returns**

True if playback is successfully started.

**3.106.2.5 StopPlayback()**

```
bool StopPlayback ( )
```

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

**Returns**

True if playback is successfully stopped.

### 3.106.3 Property Documentation

#### 3.106.3.1 Actor

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

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

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

#### 3.106.3.2 IsLinked

```
bool IsLinked [get]
```

Whether or not this [Speaker](#) has been linked to a remote voice stream.

#### 3.106.3.3 IsPlaying

```
bool IsPlaying [get]
```

Is the speaker playing right now.

#### 3.106.3.4 Lag

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

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

#### 3.106.3.5 OnRemoteVoiceRemoveAction

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

Register a method to be called when remote voice removed.

### 3.106.3.6 PlaybackDelayMaxHard

```
int PlaybackDelayMaxHard [get]
```

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

### 3.106.3.7 PlaybackDelayMaxSoft

```
int PlaybackDelayMaxSoft [get]
```

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

### 3.106.3.8 PlaybackDelayMinSoft

```
int PlaybackDelayMinSoft [get]
```

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

### 3.106.3.9 PlaybackOnlyWhenEnabled

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

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

### 3.106.3.10 PlaybackStarted

```
bool PlaybackStarted [get]
```

Returns if the playback is on.

## 3.107 TestTone Class Reference

Inherits MonoBehaviour.

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

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

Inherits [IAudioPusher< T >](#).

## Public Member Functions

- [ToneAudioPusher](#) (int frequency=440, int bufSizeMs=100, int samplingRate=48000, int channels=2)  
*Create a new [ToneAudioReader](#) instance*
- void [SetCallback](#) (Action< T[]> callback, [ObjectFactory](#)< T[], int > bufferFactory)  
*Set the callback function used for pushing data*
- void **Dispose** ()

## Properties

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

### 3.108.1 Detailed Description

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

### 3.108.2 Constructor & Destructor Documentation

#### 3.108.2.1 ToneAudioPusher()

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

### 3.108.3 Member Function Documentation



### 3.108.3.1 SetCallback()

```
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 >](#).

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

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

Inherits [IAudioReader< T >](#).

### Public Member Functions

- [ToneAudioReader](#) (Func< double > clockSec=null, double frequency=440, int samplingRate=48000, int channels=2)  
*Create a new [ToneAudioReader](#) instance*
- void **Dispose** ()
- bool [Read](#) (T[] buf)  
*Fill full given frame buffer with source uncompressed data or return false if not enough such data.*

### Properties

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

### 3.109.1 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.

### 3.109.2 Constructor & Destructor Documentation

#### 3.109.2.1 ToneAudioReader()

```
ToneAudioReader (
    Func< double > clockSec = null,
    double frequency = 440,
    int samplingRate = 48000,
    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 '()' => <code>AudioSettings.dspTime</code> 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.

### 3.109.3 Member Function Documentation

#### 3.109.3.1 Read()

```
bool Read (
    T[] buffer )
```

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

##### Parameters

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

##### Returns

True if buffer was filled successfully, false otherwise.

Implements [IDataReader< T >](#).

### 3.109.4 Property Documentation

#### 3.109.4.1 Channels

`int Channels [get]`

Number of channels in the audio signal.

#### 3.109.4.2 Error

`string Error [get]`

If not null, audio object is in invalid state.

#### 3.109.4.3 SamplingRate

`int SamplingRate [get]`

Sampling rate of the audio signal (in Hz).

### 3.110 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]

### 3.111 UnityAudioOut Class Reference

Inherits [IAudioOut< float >](#).

#### Classes

- struct [PlayDelayConfig](#)

## Public Member Functions

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

## Static Public Attributes

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

## Properties

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

## 3.112 UnityMicrophone Class Reference

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

### Static Public Member Functions

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

## Properties

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

### 3.112.1 Detailed Description

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

## 3.113 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](#).*

### 3.113.1 Detailed Description

Exception thrown if an unsupported codec is encountered.

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

### 3.113.2 Constructor & Destructor Documentation

#### 3.113.2.1 UnsupportedCodecException()

```
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>	Loogger.

## 3.114 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](#).*

### 3.114.1 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.

### 3.114.2 Constructor & Destructor Documentation

#### 3.114.2.1 UnsupportedSampleTypeException()

```
UnsupportedSampleTypeException (
    Type t )
```

Create a new [UnsupportedSampleTypeException](#).

##### Parameters

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

## 3.115 OpusCodec.Util Class Reference

## 3.116 VoiceClient Class Reference

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

Inherits [IDisposable](#).

### Public Member Functions

- delegate void [RemoteVoiceInfoDelegate](#) (int channelId, int playerId, byte voiceId, [VoiceInfo](#) voiceInfo, ref [RemoteVoiceOptions](#) options)  
*Remote voice info event delegate.*
- IEnumerable< [LocalVoice](#) > [LocalVoicesInChannel](#) (int channelId)  
*Iterates through copy of all local voices list of given channel.*
- void **LogRemoteVoiceStates** ()
- void **SetRemoteVoiceDelayFrames** ([Codec](#) codec, int delayFrames)
- void [Service](#) ()  
*This method dispatches all available incoming commands and then sends this client's outgoing commands. Call this method regularly (2..20 times a second).*
- [LocalVoice](#) [CreateLocalVoice](#) ([VoiceInfo](#) voiceInfo, int channelId=0, [IEncoder](#) encoder=null)  
*Creates basic outgoing stream w/o data processing support. Provided encoder should generate output data stream.*
- [LocalVoiceFramed](#)< T > [CreateLocalVoiceFramed](#)< T > ([VoiceInfo](#) voiceInfo, int frameSize, int channelId=0, [IEncoder](#) encoder=null)  
*Creates outgoing stream consuming sequence of values passed in array buffers of arbitrary length which repacked in frames of constant length for further processing and encoding.*
- [LocalVoice](#) [CreateLocalVoiceAudioFromSource](#) ([VoiceInfo](#) voiceInfo, [IAudioDesc](#) source, [AudioSampleType](#) sampleType, [IEncoder](#) encoder=null, int channelId=0)  
*Creates outgoing audio stream of type automatically assigned and adds procedures (callback or serviceable) for consuming given audio source data. Adds audio specific features (e.g. resampling, level meter) to processing pipeline and to returning stream handler.*
- void [RemoveLocalVoice](#) ([LocalVoice](#) voice)  
*Removes local voice (outgoing data stream).*

*Parameters*

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

- void **Dispose** ()

**Properties**

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

**3.116.1 Detailed Description**

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

**3.116.2 Member Function Documentation****3.116.2.1 CreateLocalVoice()**

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

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

## Parameters

<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.

### 3.116.2.2 CreateLocalVoiceAudioFromSource()

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

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

## Parameters

<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>sampleType</i>	<a href="#">Voice</a> 's audio sample type. If does not match source audio sample type, conversion will occur.
<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.

### 3.116.2.3 CreateLocalVoiceFramed< T >()

```
LocalVoiceFramed<T> CreateLocalVoiceFramed< T > (
    VoiceInfo voiceInfo,
    int frameSize,
```



```
int channelId = 0,  
IEncoder encoder = null )
```

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

## Template Parameters

<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.

## 3.116.2.4 LocalVoicesInChannel()

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

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

## 3.116.2.5 RemoteVoiceInfoDelegate()

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

Remote voice info event delegate.

## 3.116.2.6 RemoveLocalVoice()

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

Removes local voice (outgoing data stream).

**Parameters**

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

**3.116.2.7 Service()**

```
void Service ( )
```

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

**3.116.3 Property Documentation****3.116.3.1 DebugLostPercent**

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

Lost frames simulation ratio.

**3.116.3.2 FramesLost**

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

Lost frames counter.

**3.116.3.3 FramesReceived**

```
int FramesReceived [get]
```

Received frames counter.

**3.116.3.4 FramesSent**

```
int FramesSent [get]
```

Sent frames counter.

### 3.116.3.5 FramesSentBytes

```
int FramesSentBytes [get]
```

Sent frames bytes counter.

### 3.116.3.6 LocalVoices

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

Iterates through copy of all local voices list.

### 3.116.3.7 OnRemoteVoiceInfoAction

```
RemoteVoiceInfoDelegate OnRemoteVoiceInfoAction [get], [set]
```

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

### 3.116.3.8 RemoteVoiceInfos

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

Iterates through all remote voices infos.

### 3.116.3.9 RoundTripTime

```
int RoundTripTime [get]
```

Average time required voice packet to return to sender.

### 3.116.3.10 RoundTripTimeVariance

```
int RoundTripTimeVariance [get]
```

Average round trip time variation.

### 3.116.3.11 SuppressInfoDuplicateWarning

bool SuppressInfoDuplicateWarning [get], [set]

Do not log warning when duplicate info received.

## 3.117 VoiceComponent Class Reference

Inherits MonoBehaviour, and [ILoggableDependent](#).

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

### Protected Member Functions

- virtual void **Awake** ()

### Protected Attributes

- DebugLevel **LogLevel** = DebugLevel.INFO

### Properties

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

## 3.118 VoiceConnection Class Reference

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

Inherits ConnectionHandler, and [ILoggable](#).

Inherited by [PhotonVoiceNetwork](#).

### Public Member Functions

- bool [ConnectUsingSettings](#) (AppSettings overwriteSettings=null)  
*Connect to [Photon](#) server using [Settings](#)*
- void [InitRecorder](#) ([Recorder](#) rec)  
*Initializes the [Recorder](#) component to be able to transmit audio.*
- void [SetPlaybackDelaySettings](#) ([PlaybackDelaySettings](#) gpds)  
*Sets the global configuration for the playback behaviour in case of delays.*
- void [SetGlobalPlaybackDelaySettings](#) (int low, int high, int max)  
*Sets the global configuration for the playback behaviour in case of delays.*

## Public Attributes

- AppSettings [Settings](#)  
*Settings to be used by this voice connection*
- Func< int, byte, object, [Speaker](#) > [SpeakerFactory](#)  
*Special factory to link [Speaker](#) components with incoming remote audio streams*
- float [MinimalTimeScaleToDispatchInFixedUpdate](#) = -1f  
*Configures the minimal Time.timeScale at which [Voice](#) client will dispatch incoming messages within LateUpdate.*
- bool [AutoCreateSpeakerIfNotFound](#) = true  
*Auto instantiate a GameObject and attach a [Speaker](#) component to link to a remote audio stream if no candidate could be found*

## Protected Member Functions

- override void **Awake** ()
- virtual void **Update** ()
- virtual void **FixedUpdate** ()
- void [Dispatch](#) ()  
*Dispatches incoming network messages for [Voice](#) client. Called in FixedUpdate or LateUpdate.*
- override void **OnDisable** ()
- virtual void **OnDestroy** ()
- virtual [Speaker](#) **SimpleSpeakerFactory** (int playerId, byte 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*
- [Recorder](#) [PrimaryRecorder](#) [get, set]

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

- DebugLevel **GlobalRecordersLogLevel** [get, set]
- DebugLevel **GlobalSpeakersLogLevel** [get, set]
- int **GlobalPlaybackDelay** [get, set]
- string **BestRegionSummaryInPreferences** [get, set]

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

- int **GlobalPlaybackDelayMinSoft** [get]

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

- int **GlobalPlaybackDelayMaxSoft** [get]

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

- int **GlobalPlaybackDelayMaxHard** [get]

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

## Events

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

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

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

Fires when a remote voice stream is added

### 3.118.1 Detailed Description

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

### 3.118.2 Member Function Documentation

#### 3.118.2.1 ConnectUsingSettings()

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

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

##### Parameters

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

##### Returns

If true voice connection command was sent from client

### 3.118.2.2 Dispatch()

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

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

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

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

### 3.118.2.3 InitRecorder()

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

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

#### Parameters

<i>rec</i>	The <a href="#">Recorder</a> to be initialized.
------------	---

### 3.118.2.4 SetGlobalPlaybackDelaySettings()

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

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

#### Parameters

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

### 3.118.2.5 SetPlaybackDelaySettings()

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

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



## Parameters

<code>gpds</code>	Playback delay configuration struct.
-------------------	--------------------------------------

### 3.118.3 Member Data Documentation

#### 3.118.3.1 AutoCreateSpeakerIfNotFound

```
bool AutoCreateSpeakerIfNotFound = true
```

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

#### 3.118.3.2 MinimalTimeScaleToDispatchInFixedUpdate

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

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

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

#### 3.118.3.3 Settings

```
AppSettings Settings
```

Settings to be used by this voice connection

#### 3.118.3.4 SpeakerFactory

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

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

### 3.118.4 Property Documentation

#### 3.118.4.1 BestRegionSummaryInPreferences

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

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

#### 3.118.4.2 ClientState

```
ClientState ClientState [get]
```

Returns [Photon Voice](#) client state.

#### 3.118.4.3 FramesLostPercent

```
float FramesLostPercent [get]
```

Percentage of lost frames.

#### 3.118.4.4 FramesLostPerSecond

```
float FramesLostPerSecond [get]
```

Number of frames lost per second.

#### 3.118.4.5 FramesReceivedPerSecond

```
float FramesReceivedPerSecond [get]
```

Number of frames received per second.

#### 3.118.4.6 GlobalPlaybackDelayMaxHard

```
int GlobalPlaybackDelayMaxHard [get]
```

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

#### 3.118.4.7 GlobalPlaybackDelayMaxSoft

```
int GlobalPlaybackDelayMaxSoft [get]
```

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

#### 3.118.4.8 GlobalPlaybackDelayMinSoft

```
int GlobalPlaybackDelayMinSoft [get]
```

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

#### 3.118.4.9 Logger

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

[Logger](#) used by this component

#### 3.118.4.10 LogLevel

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

Log level for this component

#### 3.118.4.11 PrimaryRecorder

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

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

#### 3.118.4.12 SpeakerPrefab

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

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

### 3.118.4.13 VoiceClient

`VoiceClient VoiceClient [get]`

Returns underlying [Photon Voice](#) client.

## 3.118.5 Event Documentation

### 3.118.5.1 RemoteVoiceAdded

`Action<RemoteVoiceLink> RemoteVoiceAdded`

Fires when a remote voice stream is added

### 3.118.5.2 SpeakerLinked

`Action<Speaker> SpeakerLinked`

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

## 3.119 VoiceDebugScript Class Reference

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

Inherits [MonoBehaviourPun](#).

### Public Member Functions

- void **CantHearYou** ()

### Public Attributes

- bool [ForceRecordingAndTransmission](#)  
*Make sure recorder.TransmitEnabled and recorder.IsRecording are true.*
- AudioClip [TestAudioClip](#)  
*Audio file to be broadcast when TestUsingAudioClip is enabled.*
- bool [TestUsingAudioClip](#)  
*Broadcast Audio file to make sure transmission over network works if microphone (audio input device/hardware) is not reliable. Requires setting AudioClip in TestAudioClip.*
- bool [DisableVad](#)  
*Disable recorder.VoiceDetection for easier testing.*
- bool [IncreaseLogLevels](#)  
*Set main voice component's log level to ALL (max).*
- bool [LocalDebug](#)  
*Debug DebugEcho mode (Can't Hear My Self?!).*

### 3.119.1 Detailed Description

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

### 3.119.2 Member Data Documentation

#### 3.119.2.1 DisableVad

`bool DisableVad`

Disable recorder.VoiceDetection for easier testing.

#### 3.119.2.2 ForceRecordingAndTransmission

`bool ForceRecordingAndTransmission`

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

#### 3.119.2.3 IncreaseLogLevels

`bool IncreaseLogLevels`

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

#### 3.119.2.4 LocalDebug

`bool LocalDebug`

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

#### 3.119.2.5 TestAudioClip

`AudioClip TestAudioClip`

Audio file to be broadcast when `TestUsingAudioClip` is enabled.

### 3.119.2.6 TestUsingAudioClip

```
bool TestUsingAudioClip
```

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

## 3.120 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.*

### 3.120.1 Detailed Description

Simple voice activity detector triggered by signal level.

### 3.120.2 Member Function Documentation

#### 3.120.2.1 Process()

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

Implements [IProcessor< T >](#).

### 3.120.3 Property Documentation

#### 3.120.3.1 ActivityDelayMs

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

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

#### 3.120.3.2 Detected

```
bool Detected [get], [protected set]
```

If true, voice detected.

### 3.120.3.3 DetectedTime

`DateTime DetectedTime [get]`

Last time when switched to detected state.

### 3.120.3.4 On

`bool On [get], [set]`

If true, voice detection enabled.

### 3.120.3.5 Threshold

`float Threshold [get], [set]`

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

## 3.120.4 Event Documentation

### 3.120.4.1 OnDetected

`Action OnDetected`

Called when switched to detected state.

## 3.121 AudioUtil.VoiceDetectorCalibration< T > Class Template Reference

Calibration Utility for [Voice](#) Detector

Inherits [IProcessor< T >](#).

### Public Member Functions

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



## Protected Attributes

- int `calibrateCount`

## Properties

- bool `IsCalibrating` [get]

### 3.121.1 Detailed Description

Calibration Utility for [Voice](#) Detector

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

### 3.121.2 Constructor & Destructor Documentation

#### 3.121.2.1 VoiceDetectorCalibration()

```
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.

### 3.121.3 Member Function Documentation

#### 3.121.3.1 Calibrate()

```
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.

**3.121.3.2 Process()**

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

Implements [IProcessor< T >](#).

**3.122 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****3.122.1 Detailed Description**

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

## 3.123 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

#### 3.123.1 Detailed Description

[VoiceDetector](#) specialization for float audio.

#### 3.123.2 Constructor & Destructor Documentation

##### 3.123.2.1 VoiceDetectorFloat()

```
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.

## 3.124 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

### 3.124.1 Detailed Description

[VoiceDetector](#) specialization for float audio.

### 3.124.2 Constructor & Destructor Documentation

#### 3.124.2.1 VoiceDetectorShort()

```
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.

## 3.125 VoiceEvent Class Reference

### Static Public Attributes

- const byte [Code](#) = 202  
*Single event used for voice communications.*
- const byte **FrameCode** = 203

### 3.125.1 Member Data Documentation

#### 3.125.1.1 Code

```
const byte Code = 202 [static]
```

Single event used for voice communications.

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

## 3.126 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, OpusCodec.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]  
*Number of channels.*
- int [FrameDurationUs](#) [get, set]  
*Uncompressed frame (audio packet) size in microseconds.*
- int [Bitrate](#) [get, set]  
*Target bitrate (in bits/second).*
- int [Width](#) [get, set]  
*Video width.*
- int [Height](#) [get, set]  
*Video height*
- int [FPS](#) [get, set]  
*Video frames per second*
- int [KeyFrameInt](#) [get, set]  
*Video keyframe interval in frames*
- object [UserData](#) [get, set]  
*Optional user data. Should be serializable by [Photon](#).*
- int [FrameDurationSamples](#) [get]  
*Uncompressed frame (data packet) size in samples.*
- int [FrameSize](#) [get]  
*Uncompressed frame (data packet) array size.*

### 3.126.1 Detailed Description

Describes stream properties.

### 3.126.2 Member Function Documentation

#### 3.126.2.1 CreateAudio()

```
static VoiceInfo CreateAudio (
    Codec codec,
    int samplingRate,
    int channels,
    int frameDurationUs,
    object userdata = null ) [static]
```

Create stream info for an Opus audio stream.

##### Parameters

<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.

#### 3.126.2.2 CreateAudioOpus()

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

Create stream info for an Opus audio stream.

##### Parameters

<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.

### 3.126.3 Property Documentation

#### 3.126.3.1 Bitrate

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

Target bitrate (in bits/second).

#### 3.126.3.2 Channels

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

Number of channels.

#### 3.126.3.3 FPS

```
int FPS [get], [set]
```

Video frames per second

#### 3.126.3.4 FrameDurationSamples

```
int FrameDurationSamples [get]
```

Uncompressed frame (data packet) size in samples.

#### 3.126.3.5 FrameDurationUs

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

Uncompressed frame (audio packet) size in microseconds.

### 3.126.3.6 FrameSize

```
int FrameSize [get]
```

Uncompressed frame (data packet) array size.

### 3.126.3.7 Height

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

Video height

### 3.126.3.8 KeyFrameInt

```
int KeyFrameInt [get], [set]
```

Video keyframe interval in frames

### 3.126.3.9 SamplingRate

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

Audio sampling rate (frequency, in Hz).

### 3.126.3.10 UserData

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

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

### 3.126.3.11 Width

```
int Width [get], [set]
```

Video width.



## 3.127 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]

### 3.127.1 Detailed Description

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

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

### 3.127.2 Constructor & Destructor Documentation

#### 3.127.2.1 VoiceLevelDetectCalibrate()

```
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.

### 3.127.3 Member Function Documentation

#### 3.127.3.1 Calibrate()

```
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.

#### 3.127.3.2 Process()

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

Implements [IProcessor< T >](#).

### 3.127.4 Property Documentation

#### 3.127.4.1 LevelMeter

```
ILevelMeter LevelMeter [get]
```

The [LevelMeter](#) in use.

### 3.127.4.2 VoiceDetector

[IVoiceDetector](#) [VoiceDetector](#) [get]

The [VoiceDetector](#) in use

## 3.128 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]

## 3.129 WebRtcAudioDsp Class Reference

Inherits [VoiceComponent](#).

### Public Member Functions

- bool [SetOrSwitchAudioListener](#) ([AudioListener](#) audioListener)  
*Set the [AudioListener](#) to be used with this [WebRtcAudioDsp](#)*
- bool [SetOrSwitchAudioOutCapture](#) ([AudioOutCapture](#) audioOutCapture)  
*Set the [AudioOutCapture](#) to be used with this [WebRtcAudioDsp](#)*

### Public Attributes

- bool **AECMobileComfortNoise**

## Protected Member Functions

- override void **Awake** ()

## Properties

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

## Additional Inherited Members

### 3.129.1 Member Function Documentation

#### 3.129.1.1 SetOrSwitchAudioListener()

```
bool SetOrSwitchAudioListener (
    AudioListener audioListener )
```

Set the AudioListener to be used with this [WebRtcAudioDsp](#)

##### Parameters

<i>audioListener</i>	The audioListener to be used
----------------------	------------------------------

##### Returns

Success or failure

#### 3.129.1.2 SetOrSwitchAudioOutCapture()

```
bool SetOrSwitchAudioOutCapture (
    AudioOutCapture audioOutCapture )
```

Set the [AudioOutCapture](#) to be used with this [WebRtcAudioDsp](#)

## Parameters

<i>audioOutCapture</i>	The audioOutCapture to be used
------------------------	--------------------------------

## Returns

Success or failure

## 3.130 WebRTCAudioLib Class Reference

Inherited by [WebRTCAudioProcessor](#).

### Public Types

- enum **Error**
- enum **Param**

### Public Member Functions

- static IntPtr **webrtc\_audio\_processor\_create** (int samplingRate, int channels, int frameSize, int reverseSamplingRate, int reverseChannels)
- 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)

## 3.131 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** ()

### Static Public Attributes

- static readonly int[] **SupportedSamplingRates** = { 8000, 16000, 32000, 48000 }

## Properties

- int **AECStreamDelayMs** [set]
- bool?? **AEC** [set]
- bool? **AECHighPass** [set]
- bool?? **AECMobile** [set]
- bool? **HighPass** [set]
- bool? **NoiseSuppression** [set]
- bool? **AGC** [set]
- int **AGCCompressionGain** [set]
- int **AGCTargetLevel** [set]
- bool? **AGC2** [set]
- bool? **VAD** [set]
- bool **Bypass** [set]

## Additional Inherited Members

# Index

- AccumAvgPeakAmp
  - AudioUtil.ILevelMeter, [45](#)
- AcquireOrCreate
  - ObjectPool< TType, TInfo >, [75](#), [76](#)
- ActivityDelayMs
  - AudioUtil.IVoiceDetector, [52](#)
  - AudioUtil.VoiceDetector< T >, [133](#)
- Actor
  - Speaker, [108](#)
- AddPostProcessor
  - LocalVoiceFramed< T >, [69](#)
- AddPreProcessor
  - LocalVoiceFramed< T >, [69](#)
- AllowBluetooth
  - Photon.Voice.IOS, [8](#)
- Ambient
  - Photon.Voice.IOS, [6](#)
- AndroidAudioInAEC, [13](#)
- Audio
  - POpusCodec.Enums, [12](#)
- AudioClip
  - Recorder, [94](#)
- AudioClipWrapper, [13](#)
- AudioDesc, [14](#)
- AudioGroup
  - Recorder, [94](#)
- AudioInChangeNotifier, [14](#)
- AudioInEnumerator, [14](#)
- AudioOpus
  - Photon.Voice, [6](#)
- AudioOutCapture, [15](#)
- AudioProcessing
  - Photon.Voice.IOS, [7](#)
- AudioSampleType
  - Photon.Voice, [5](#)
- AudioSessionCategory
  - Photon.Voice.IOS, [6](#)
- AudioSessionCategoryOption
  - Photon.Voice.IOS, [7](#)
- AudioSessionMode
  - Photon.Voice.IOS, [8](#)
- AudioSessionParameters, [15](#)
- AudioSessionParametersPresets, [16](#)
  - Game, [16](#)
  - VoIP, [16](#)
- AudioSyncBuffer< T >, [16](#)
- AudioUtil, [17](#)
  - Convert, [18](#)
  - ForceToStereo< T >, [19](#)
  - Resample< T >, [19](#)
  - ResampleAndConvert, [20](#)
- AudioUtil.ILevelMeter, [44](#)
  - AccumAvgPeakAmp, [45](#)
  - CurrentAvgAmp, [45](#)
  - CurrentPeakAmp, [45](#)
  - ResetAccumAvgPeakAmp, [45](#)
- AudioUtil.IVoiceDetector, [51](#)
  - ActivityDelayMs, [52](#)
  - Detected, [52](#)
  - DetectedTime, [52](#)
  - On, [52](#)
  - OnDetected, [53](#)
  - Threshold, [53](#)
- AudioUtil.LevelMeter< T >, [53](#)
  - Process, [54](#)
  - ResetAccumAvgPeakAmp, [55](#)
- AudioUtil.LevelMeterDummy, [55](#)
  - ResetAccumAvgPeakAmp, [55](#)
- AudioUtil.LevelMeterFloat, [56](#)
  - LevelMeterFloat, [56](#)
- AudioUtil.LevelMeterShort, [56](#)
  - LevelMeterShort, [57](#)
- AudioUtil.Resampler< T >, [103](#)
  - Process, [104](#)
  - Resampler, [104](#)
- AudioUtil.ToneAudioPusher< T >, [109](#)
  - SetCallback, [110](#)
  - ToneAudioPusher, [110](#)
- AudioUtil.ToneAudioReader< T >, [111](#)
  - Channels, [112](#)
  - Error, [113](#)
  - Read, [112](#)
  - SamplingRate, [113](#)
  - ToneAudioReader, [112](#)
- AudioUtil.VoiceDetector< T >, [132](#)
  - ActivityDelayMs, [133](#)
  - Detected, [133](#)
  - DetectedTime, [133](#)
  - On, [134](#)
  - OnDetected, [134](#)
  - Process, [133](#)
  - Threshold, [134](#)
- AudioUtil.VoiceDetectorCalibration< T >, [134](#)
  - Calibrate, [135](#)
  - Process, [136](#)
  - VoiceDetectorCalibration, [135](#)
- AudioUtil.VoiceDetectorDummy, [136](#)
- AudioUtil.VoiceDetectorFloat, [137](#)

- VoiceDetectorFloat, 137
- AudioUtil.VoiceDetectorShort, 137
- VoiceDetectorShort, 138
- AudioUtil.VoiceLevelDetectCalibrate< T >, 143
  - Calibrate, 144
  - LevelMeter, 144
  - Process, 144
  - VoiceDetector, 144
  - VoiceLevelDetectCalibrate, 143
- Auto
  - POpusCodec.Enums, 12
- AutoConnectAndJoin
  - PhotonVoiceNetwork, 82
- AutoCreateRecorderIfNotFound
  - PhotonVoiceView, 85
- AutoCreateSpeakerIfNotFound
  - VoiceConnection, 127
- AutoLeaveAndDisconnect
  - PhotonVoiceNetwork, 82
- AutoStart
  - Recorder, 94
- Bandwidth
  - POpusCodec.Enums, 11
- BestRegionSummaryInPreferences
  - VoiceConnection, 127
- Bitrate
  - Recorder, 95
  - VoiceInfo, 141
- BufferReaderPushAdapter
  - BufferReaderPushAdapter< T >, 21
- BufferReaderPushAdapter< T >, 20
  - BufferReaderPushAdapter, 21
  - Service, 21
- BufferReaderPushAdapterAsyncPool
  - BufferReaderPushAdapterAsyncPool< T >, 22
- BufferReaderPushAdapterAsyncPool< T >, 22
  - BufferReaderPushAdapterAsyncPool, 22
  - Service, 23
- BufferReaderPushAdapterAsyncPoolCopy
  - BufferReaderPushAdapterAsyncPoolCopy< T >, 23
- BufferReaderPushAdapterAsyncPoolCopy< T >, 23
  - BufferReaderPushAdapterAsyncPoolCopy, 23
  - Service, 24
- BufferReaderPushAdapterAsyncPoolFloatToShort, 24
  - BufferReaderPushAdapterAsyncPoolFloatToShort, 25
  - Service, 25
- BufferReaderPushAdapterAsyncPoolShortToFloat, 25
  - BufferReaderPushAdapterAsyncPoolShortToFloat, 26
  - Service, 26
- BufferReaderPushAdapterBase
  - BufferReaderPushAdapterBase< T >, 27
- BufferReaderPushAdapterBase< T >, 27
  - BufferReaderPushAdapterBase, 27
  - Dispose, 27
  - Service, 28
- Calibrate
  - AudioUtil.VoiceDetectorCalibration< T >, 135
  - AudioUtil.VoiceLevelDetectCalibrate< T >, 144
- ChannelId
  - RemoteVoiceInfo, 101
- Channels
  - AudioUtil.ToneAudioReader< T >, 112
  - IAudioDesc, 36
  - POpusCodec.Enums, 11
  - VoiceInfo, 141
- ClearProcessors
  - LocalVoiceFramed< T >, 69
- ClientState
  - VoiceConnection, 128
- Code
  - VoiceEvent, 138
- Codec
  - Photon.Voice, 6
- ConnectAndJoin, 28
- ConnectAndJoinRoom
  - PhotonVoiceNetwork, 81
- ConnectUsingSettings
  - VoiceConnection, 125
- Convert
  - AudioUtil, 18
- Count
  - Framer< T >, 35
- Create
  - LocalVoiceAudio< T >, 64
- CreateAudio
  - VoiceInfo, 140
- CreateAudioOpus
  - VoiceInfo, 140
- CreateLocalVoice
  - VoiceClient, 117
- CreateLocalVoiceAudioFromSource
  - VoiceClient, 118
- CreateLocalVoiceFramed< T >
  - VoiceClient, 118
- CurrentAvgAmp
  - AudioUtil.ILevelMeter, 45
- CurrentPeakAmp
  - AudioUtil.ILevelMeter, 45
- DebugEchoMode
  - LocalVoice, 62
  - Recorder, 95
- DebugLostPercent
  - VoiceClient, 121
- Decoder
  - RemoteVoiceOptions, 103
- Default
  - Photon.Voice.IOS, 8
- DefaultToSpeaker
  - Photon.Voice.IOS, 8
- Delay
  - POpusCodec.Enums, 11
- Delay10ms
  - POpusCodec.Enums, 12



- Delay20ms
  - POpusCodec.Enums, 12
- Delay2dot5ms
  - POpusCodec.Enums, 12
- Delay40ms
  - POpusCodec.Enums, 12
- Delay5ms
  - POpusCodec.Enums, 12
- Delay60ms
  - POpusCodec.Enums, 12
- DequeueOutput
  - IEncoder, 42
- Detected
  - AudioUtil.IVoiceDetector, 52
  - AudioUtil.VoiceDetector< T >, 133
- DetectedTime
  - AudioUtil.IVoiceDetector, 52
  - AudioUtil.VoiceDetector< T >, 133
- DisableVad
  - VoiceDebugScript, 131
- Disconnect
  - PhotonVoiceNetwork, 82
- Dispatch
  - VoiceConnection, 125
- Dispose
  - BufferedReaderPushAdapterBase< T >, 27
  - LoadBalancingTransport, 59
  - LocalVoiceFramed< T >, 69
  - ObjectPool< TType, TInfo >, 76
- DuckOthers
  - Photon.Voice.IOS, 8
- Dummy
  - LocalVoiceAudioDummy, 67
- EncoderDelay
  - OpusEncoder, 79
- Encrypt
  - LocalVoice, 62
  - Recorder, 95
- EndOfStream
  - IEncoder, 42
- Error
  - AudioUtil.ToneAudioReader< T >, 113
  - IAudioDesc, 37
  - IDecoder, 41
  - IEncoder, 42
- FactoryPrimitiveArrayPool< T >, 33
- FactoryReusableArray< T >, 33
- ForceRecordingAndTransmission
  - VoiceDebugScript, 131
- ForceToStereo< T >
  - AudioUtil, 19
- FPS
  - VoiceInfo, 141
- Frame
  - Framer< T >, 36
- FrameBuffer, 34
- FrameDuration
  - Recorder, 95
- FrameDurationSamples
  - VoiceInfo, 141
- FrameDurationUs
  - VoiceInfo, 141
- FrameOut< T >, 34
- Framer
  - Framer< T >, 35
- Framer< T >, 35
  - Count, 35
  - Frame, 36
  - Framer, 35
- FrameSize
  - LocalVoiceFramedBase, 71
  - VoiceInfo, 141
- FramesLost
  - VoiceClient, 121
- FramesLostPercent
  - VoiceConnection, 128
- FramesLostPerSecond
  - VoiceConnection, 128
- FramesReceived
  - VoiceClient, 121
- FramesReceivedPerSecond
  - VoiceConnection, 128
- FramesSent
  - LocalVoice, 62
  - VoiceClient, 121
- FramesSentBytes
  - LocalVoice, 62
  - VoiceClient, 121
- Fullband
  - POpusCodec.Enums, 11
- Game
  - AudioSessionParametersPresets, 16
- GlobalInterestGroup
  - LoadBalancingTransport, 59
- GlobalPlaybackDelayMaxHard
  - VoiceConnection, 128
- GlobalPlaybackDelayMaxSoft
  - VoiceConnection, 128
- GlobalPlaybackDelayMinSoft
  - VoiceConnection, 129
- Height
  - VoiceInfo, 142
- IAudioDesc, 36
  - Channels, 36
  - Error, 37
  - SamplingRate, 37
- IAudioOut< T >, 37
- IAudioPusher< T >, 37
  - SetCallback, 38
- IAudioReader< T >, 38
- IDataReader< T >, 39
  - Read, 39
- IDecoder, 39

- Error, [41](#)
- Input, [40](#)
- Open, [40](#)
- IDecoderDirect< B >, [41](#)
- IDecoderQueuedOutputImageNative, [41](#)
- IEncoder, [41](#)
  - DequeueOutput, [42](#)
  - EndOfStream, [42](#)
  - Error, [42](#)
  - Output, [43](#)
- IEncoderDirect< B >, [43](#)
  - Input, [43](#)
- IEncoderDirectImage, [44](#)
- ILocalVoiceAudio, [45](#)
  - LevelMeter, [46](#)
  - VoiceDetector, [47](#)
  - VoiceDetectorCalibrate, [46](#)
  - VoiceDetectorCalibrating, [47](#)
- ILoggable, [47](#)
- ILoggableDependent, [47](#)
- ILogger, [47](#)
- ImageBufferInfo, [48](#)
- ImageBufferNative, [48](#)
- ImageBufferNativeAlloc, [48](#)
- ImageBufferNativeGCHandleSinglePlane, [49](#)
- ImageBufferNativePool< T >, [49](#)
- ImageOutputBuf, [49](#)
- IncreaseLogLevels
  - VoiceDebugScript, [131](#)
- Info
  - LocalVoice, [62](#)
  - ObjectPool< TType, TInfo >, [77](#)
  - RemoteVoiceInfo, [101](#)
- Init
  - ObjectPool< TType, TInfo >, [76](#)
  - PhotonVoiceView, [85](#)
  - Recorder, [92](#), [93](#)
- InitRecorder
  - VoiceConnection, [126](#)
- Input
  - IDecoder, [40](#)
  - IEncoderDirect< B >, [43](#)
  - OpusCodec.Decoder< T >, [30](#)
  - RawCodec.Decoder< T >, [29](#)
- InputFactory
  - Recorder, [95](#)
- Instance
  - PhotonVoiceNetwork, [83](#)
- InterestGroup
  - LocalVoice, [62](#)
  - Recorder, [95](#)
- IOAudioForceToSpeaker, [50](#)
- IProcessor< T >, [50](#)
  - Process, [50](#)
- IResettable, [51](#)
- IsCurrentlyTransmitting
  - LocalVoice, [63](#)
  - Recorder, [96](#)
- IServiceable, [51](#)
  - Service, [51](#)
- IsInitialized
  - Recorder, [96](#)
- IsLinked
  - Speaker, [108](#)
- IsPhotonViewReady
  - PhotonVoiceView, [85](#)
- IsPlaying
  - Speaker, [108](#)
- IsRecorder
  - PhotonVoiceView, [86](#)
- IsRecording
  - PhotonVoiceView, [86](#)
  - Recorder, [96](#)
- IsSetup
  - PhotonVoiceView, [86](#)
- IsSpeaker
  - PhotonVoiceView, [86](#)
- IsSpeakerLinked
  - PhotonVoiceView, [86](#)
- IsSpeaking
  - PhotonVoiceView, [86](#)
- IVoiceTransport, [53](#)
- KeyFrameInt
  - VoiceInfo, [142](#)
- Lag
  - Speaker, [108](#)
- LevelMeter
  - AudioUtil.VoiceLevelDetectCalibrate< T >, [144](#)
  - ILocalVoiceAudio, [46](#)
  - Recorder, [96](#)
- LevelMeterFloat
  - AudioUtil.LevelMeterFloat, [56](#)
- LevelMeterShort
  - AudioUtil.LevelMeterShort, [57](#)
- LoadBalancingFrontend, [57](#)
- LoadBalancingTransport, [57](#)
  - Dispose, [59](#)
  - GlobalInterestGroup, [59](#)
  - LoadBalancingTransport, [58](#)
  - Service, [59](#)
  - VoiceClient, [59](#)
- LoadBalancingTransport2, [60](#)
- LocalDebug
  - VoiceDebugScript, [131](#)
- LocalUserServiceable
  - LocalVoice, [63](#)
- LocalVoice, [60](#)
  - DebugEchoMode, [62](#)
  - Encrypt, [62](#)
  - FramesSent, [62](#)
  - FramesSentBytes, [62](#)
  - Info, [62](#)
  - InterestGroup, [62](#)
  - IsCurrentlyTransmitting, [63](#)
  - LocalUserServiceable, [63](#)

- Reliable, [63](#)
- RemoveSelf, [61](#)
- TransmitEnabled, [63](#)
- LocalVoiceAudio< T >, [63](#)
  - Create, [64](#)
  - VoiceDetectorCalibrate, [65](#)
  - VoiceDetectorCalibrating, [65](#)
- LocalVoiceAudioDummy, [65](#)
  - Dummy, [67](#)
  - VoiceDetectorCalibrate, [66](#)
- LocalVoiceAudioFloat, [67](#)
- LocalVoiceAudioShort, [67](#)
- LocalVoiceFramed< T >, [67](#)
  - AddPostProcessor, [69](#)
  - AddPreProcessor, [69](#)
  - ClearProcessors, [69](#)
  - Dispose, [69](#)
  - PushData, [69](#)
  - PushDataAsync, [70](#)
  - PushDataAsyncReady, [70](#)
- LocalVoiceFramedBase, [70](#)
  - FrameSize, [71](#)
- LocalVoices
  - VoiceClient, [122](#)
- LocalVoicesInChannel
  - VoiceClient, [120](#)
- Logger, [71](#)
  - VoiceConnection, [129](#)
- LogLevel
  - VoiceConnection, [129](#)
- LoopAudioClip
  - Recorder, [96](#)
- MaxDelayHard
  - PlaybackDelaySettings, [88](#)
- MaxDelaySoft
  - PlaybackDelaySettings, [88](#)
- Measurement
  - Photon.Voice.IOS, [9](#)
- Mediumband
  - POpusCodec.Enums, [11](#)
- MicAmplifier, [71](#)
- MicAmplifierFloat, [71](#)
- MicAmplifierShort, [72](#)
- MicrophoneType
  - Recorder, [96](#)
- MicWrapper, [72](#)
- MicWrapperPusher, [72](#)
- MinDelaySoft
  - PlaybackDelaySettings, [88](#)
- MinimalTimeScaleToDispatchInFixedUpdate
  - VoiceConnection, [127](#)
- MixWithOthers
  - Photon.Voice.IOS, [7](#)
- Mono
  - POpusCodec.Enums, [11](#)
- MonoPInvokeCallbackAttribute, [73](#)
- MoviePlayback
  - Photon.Voice.IOS, [9](#)
- MultiRoute
  - Photon.Voice.IOS, [7](#)
- Music
  - POpusCodec.Enums, [12](#)
- Narrowband
  - POpusCodec.Enums, [11](#)
- NativeAndroidMicrophoneSettings, [73](#)
- ObjectFactory< TType, TInfo >, [73](#)
- ObjectPool
  - ObjectPool< TType, TInfo >, [75](#)
- ObjectPool< TType, TInfo >, [74](#)
  - AcquireOrCreate, [75](#), [76](#)
  - Dispose, [76](#)
  - Info, [77](#)
  - Init, [76](#)
  - ObjectPool, [75](#)
  - Release, [76](#), [77](#)
- On
  - AudioUtil.IVoiceDetector, [52](#)
  - AudioUtil.VoiceDetector< T >, [134](#)
- OnDetected
  - AudioUtil.IVoiceDetector, [53](#)
  - AudioUtil.VoiceDetector< T >, [134](#)
- OnRemoteVoiceInfoAction
  - VoiceClient, [122](#)
- OnRemoteVoiceRemoveAction
  - RemoteVoiceOptions, [103](#)
  - Speaker, [108](#)
- Open
  - IDecoder, [40](#)
  - OpusCodec.Decoder< T >, [30](#)
  - RawCodec.Decoder< T >, [29](#)
- OpusApplicationType
  - POpusCodec.Enums, [12](#)
- OpusCodec, [77](#)
- OpusCodec.Decoder< T >, [30](#)
  - Input, [30](#)
  - Open, [30](#)
- OpusCodec.DecoderFactory, [31](#)
- OpusCodec.Encoder< T >, [31](#)
- OpusCodec.EncoderFloat, [32](#)
- OpusCodec.EncoderShort, [32](#)
- OpusCodec.Factory, [32](#)
- OpusCodec.Util, [116](#)
- OpusDecoder< T >, [78](#)
- OpusEncoder, [78](#)
  - EncoderDelay, [79](#)
- OpusException, [79](#)
- OpusLib, [79](#)
- Output
  - IEncoder, [43](#)
- Photon, [3](#)
- Photon.Voice, [3](#)
  - AudioOpus, [6](#)
  - AudioSampleType, [5](#)
  - Codec, [6](#)

- Photon.Voice.IOS, 6
  - AllowBluetooth, 8
  - Ambient, 6
  - AudioProcessing, 7
  - AudioSessionCategory, 6
  - AudioSessionCategoryOption, 7
  - AudioSessionMode, 8
  - Default, 8
  - DefaultToSpeaker, 8
  - DuckOthers, 8
  - Measurement, 9
  - MixWithOthers, 7
  - MoviePlayback, 9
  - MultiRoute, 7
  - PlayAndRecord, 7
  - Playback, 7
  - Record, 7
  - SoloAmbient, 7
  - VideoChat, 9
  - VideoRecording, 9
  - VoiceChat, 8
- Photon.Voice.PUN, 9
- Photon.Voice.PUN.UtilityScripts, 9
- Photon.Voice.Unity, 9
- Photon.Voice.Unity.UtilityScripts, 10
- PhotonMicrophoneDeviceId
  - Recorder, 97
- PhotonMicrophoneEnumerator
  - Recorder, 97
- PhotonVoiceCreatedParams, 80
- PhotonVoiceLagSimulationGui, 80
- PhotonVoiceNetwork, 80
  - AutoConnectAndJoin, 82
  - AutoLeaveAndDisconnect, 82
  - ConnectAndJoinRoom, 81
  - Disconnect, 82
  - Instance, 83
  - UsePunAuthValues, 83
  - VoiceRoomNameSuffix, 82
  - WorkInOfflineMode, 82
- PhotonVoiceStatsGui, 83
- PhotonVoiceView, 83
  - AutoCreateRecorderIfNotFound, 85
  - Init, 85
  - IsPhotonViewReady, 85
  - IsRecorder, 86
  - IsRecording, 86
  - IsSetup, 86
  - IsSpeaker, 86
  - IsSpeakerLinked, 86
  - IsSpeaking, 86
  - RecorderInUse, 87
  - SetupDebugSpeaker, 85
  - SpeakerInUse, 87
  - UsePrimaryRecorder, 85
- Platform, 87
- PlayAndRecord
  - Photon.Voice.IOS, 7
- Playback
  - Photon.Voice.IOS, 7
- PlaybackDelayMaxHard
  - Speaker, 108
- PlaybackDelayMaxSoft
  - Speaker, 109
- PlaybackDelayMinSoft
  - Speaker, 109
- PlaybackDelaySettings, 87
  - MaxDelayHard, 88
  - MaxDelaySoft, 88
  - MinDelaySoft, 88
- PlaybackOnlyWhenEnabled
  - Speaker, 109
- PlaybackStarted
  - Speaker, 109
- PlayerId
  - RemoteVoiceInfo, 101
- POpusCodec, 10
- POpusCodec.Enums, 11
  - Audio, 12
  - Auto, 12
  - Bandwidth, 11
  - Channels, 11
  - Delay, 11
  - Delay10ms, 12
  - Delay20ms, 12
  - Delay2dot5ms, 12
  - Delay40ms, 12
  - Delay5ms, 12
  - Delay60ms, 12
  - Fullband, 11
  - Mediumband, 11
  - Mono, 11
  - Music, 12
  - Narrowband, 11
  - OpusApplicationType, 12
  - RestrictedLowDelay, 12
  - SignalHint, 12
  - Stereo, 11
  - SuperWideband, 11
  - Voice, 12
  - Voip, 12
  - Wideband, 11
- PrimaryRecorder
  - VoiceConnection, 129
- PrimitiveArrayPool< T >, 89
- Process
  - AudioUtil.LevelMeter< T >, 54
  - AudioUtil.Resampler< T >, 104
  - AudioUtil.VoiceDetector< T >, 133
  - AudioUtil.VoiceDetectorCalibration< T >, 136
  - AudioUtil.VoiceLevelDetectCalibrate< T >, 144
  - IProcessor< T >, 50
- PushData
  - LocalVoiceFramed< T >, 69
- PushDataAsync
  - LocalVoiceFramed< T >, 70

- PushDataAsyncReady
  - LocalVoiceFramed< T >, 70
- RawCodec, 89
- RawCodec.Decoder< T >, 29
  - Input, 29
  - Open, 29
- RawCodec.Encoder< T >, 31
- ReactOnSystemChanges
  - Recorder, 97
- Read
  - AudioUtil.ToneAudioReader< T >, 112
  - IDataReader< T >, 39
- Record
  - Photon.Voice.IOS, 7
- Recorder, 89
  - AudioClip, 94
  - AudioGroup, 94
  - AutoStart, 94
  - Bitrate, 95
  - DebugEchoMode, 95
  - Encrypt, 95
  - FrameDuration, 95
  - Init, 92, 93
  - InputFactory, 95
  - InterestGroup, 95
  - IsCurrentlyTransmitting, 96
  - IsInitialized, 96
  - IsRecording, 96
  - LevelMeter, 96
  - LoopAudioClip, 96
  - MicrophoneType, 96
  - PhotonMicrophoneDeviceId, 97
  - PhotonMicrophoneEnumerator, 97
  - ReactOnSystemChanges, 97
  - RecordOnlyWhenEnabled, 97
  - ReliableMode, 97
  - RequiresRestart, 97
  - ResetLocalAudio, 93
  - RestartRecording, 93
  - SamplingRate, 98
  - SkipDeviceChangeChecks, 98
  - SourceType, 98
  - StartRecording, 93
  - StopRecording, 94
  - StopRecordingWhenPaused, 98
  - TransmitEnabled, 98
  - TrySamplingRateMatch, 98
  - TypeConvert, 99
  - UnityMicrophoneDevice, 99
  - UseMicrophoneTypeFallback, 99
  - UseOnAudioFilterRead, 99
  - UserData, 99
  - VoiceDetection, 99
  - VoiceDetectionDelayMs, 100
  - VoiceDetectionThreshold, 100
  - VoiceDetector, 100
  - VoiceDetectorCalibrate, 94
  - VoiceDetectorCalibrating, 100
- Recorder.PhotonVoiceCreatedParams, 80
- RecorderInUse
  - PhotonVoiceView, 87
- RecordOnlyWhenEnabled
  - Recorder, 97
- Release
  - ObjectPool< TType, TInfo >, 76, 77
- Reliable
  - LocalVoice, 63
- ReliableMode
  - Recorder, 97
- RemoteVoiceAdded
  - VoiceConnection, 130
- RemoteVoiceInfo, 100
  - ChannelId, 101
  - Info, 101
  - PlayerId, 101
  - VoicId, 101
- RemoteVoiceInfoDelegate
  - VoiceClient, 120
- RemoteVoiceInfos
  - VoiceClient, 122
- RemoteVoiceLink, 102
- RemoteVoiceOptions, 102
  - Decoder, 103
  - OnRemoteVoiceRemoveAction, 103
  - SetOutput, 102
- RemoveLocalVoice
  - VoiceClient, 120
- RemoveSelf
  - LocalVoice, 61
- RequiresRestart
  - Recorder, 97
- Resample< T >
  - AudioUtil, 19
- ResampleAndConvert
  - AudioUtil, 20
- Resampler
  - AudioUtil.Resampler< T >, 104
- ResetAccumAvgPeakAmp
  - AudioUtil.ILevelMeter, 45
  - AudioUtil.LevelMeter< T >, 55
  - AudioUtil.LevelMeterDummy, 55
- ResetLocalAudio
  - Recorder, 93
- RestartPlayback
  - Speaker, 106
- RestartRecording
  - Recorder, 93
- RestrictedLowDelay
  - POpusCodec.Enums, 12
- RoundTripTime
  - VoiceClient, 122
- RoundTripTimeVariance
  - VoiceClient, 122
- SamplingRate
  - AudioUtil.ToneAudioReader< T >, 113
  - IAudioDesc, 37

- Recorder, 98
- VoiceInfo, 142
- SaveIncomingStreamToFile, 105
- SaveOutgoingStreamToFile, 105
- Service
  - BufferReaderPushAdapter< T >, 21
  - BufferReaderPushAdapterAsyncPool< T >, 23
  - BufferReaderPushAdapterAsyncPoolCopy< T >, 24
  - BufferReaderPushAdapterAsyncPoolFloatToShort, 25
  - BufferReaderPushAdapterAsyncPoolShortToFloat, 26
  - BufferReaderPushAdapterBase< T >, 28
  - IServiceable, 51
  - LoadBalancingTransport, 59
  - VoiceClient, 121
- SetCallback
  - AudioUtil.ToneAudioPusher< T >, 110
  - IAudioPusher< T >, 38
- SetGlobalPlaybackDelaySettings
  - VoiceConnection, 126
- SetOrSwitchAudioListener
  - WebRtcAudioDsp, 146
- SetOrSwitchAudioOutCapture
  - WebRtcAudioDsp, 146
- SetOutput
  - RemoteVoiceOptions, 102
- SetPlaybackDelaySettings
  - Speaker, 106, 107
  - VoiceConnection, 126
- Settings
  - VoiceConnection, 127
- SetupDebugSpeaker
  - PhotonVoiceView, 85
- SignalHint
  - POpusCodec.Enums, 12
- SkipDeviceChangeChecks
  - Recorder, 98
- SoloAmbient
  - Photon.Voice.IOS, 7
- SourceType
  - Recorder, 98
- Speaker, 105
  - Actor, 108
  - IsLinked, 108
  - IsPlaying, 108
  - Lag, 108
  - OnRemoteVoiceRemoveAction, 108
  - PlaybackDelayMaxHard, 108
  - PlaybackDelayMaxSoft, 109
  - PlaybackDelayMinSoft, 109
  - PlaybackOnlyWhenEnabled, 109
  - PlaybackStarted, 109
  - RestartPlayback, 106
  - SetPlaybackDelaySettings, 106, 107
  - StartPlayback, 107
  - StopPlayback, 107
  - SpeakerFactory
    - VoiceConnection, 127
  - SpeakerInUse
    - PhotonVoiceView, 87
  - SpeakerLinked
    - VoiceConnection, 130
  - SpeakerPrefab
    - VoiceConnection, 129
  - StartPlayback
    - Speaker, 107
  - StartRecording
    - Recorder, 93
  - Stereo
    - POpusCodec.Enums, 11
  - StopPlayback
    - Speaker, 107
  - StopRecording
    - Recorder, 94
  - StopRecordingWhenPaused
    - Recorder, 98
  - SuperWideband
    - POpusCodec.Enums, 11
  - SuppressInfoDuplicateWarning
    - VoiceClient, 122
- TestAudioClip
  - VoiceDebugScript, 131
- TestTone, 109
- TestUsingAudioClip
  - VoiceDebugScript, 131
- Threshold
  - AudioUtil.IVoiceDetector, 53
  - AudioUtil.VoiceDetector< T >, 134
- ToneAudioPusher
  - AudioUtil.ToneAudioPusher< T >, 110
- ToneAudioReader, 113
  - AudioUtil.ToneAudioReader< T >, 112
- TransmitEnabled
  - LocalVoice, 63
  - Recorder, 98
- TrySamplingRateMatch
  - Recorder, 98
- TypeConvert
  - Recorder, 99
- UnityAudioOut, 113
- UnityAudioOut.PlayDelayConfig, 88
- UnityMicrophone, 114
- UnityMicrophoneDevice
  - Recorder, 99
- UnsupportedCodecException, 114
  - UnsupportedCodecException, 115
- UnsupportedSampleTypeException, 115
  - UnsupportedSampleTypeException, 116
- UseMicrophoneTypeFallback
  - Recorder, 99
- UseOnAudioFilterRead
  - Recorder, 99
- UsePrimaryRecorder

- PhotonVoiceView, 85
- UsePunAuthValues
  - PhotonVoiceNetwork, 83
- UserData
  - Recorder, 99
  - VoiceInfo, 142
- VideoChat
  - Photon.Voice.IOS, 9
- VideoRecording
  - Photon.Voice.IOS, 9
- Voice
  - POpusCodec.Enums, 12
- VoiceChat
  - Photon.Voice.IOS, 8
- VoiceClient, 116
  - CreateLocalVoice, 117
  - CreateLocalVoiceAudioFromSource, 118
  - CreateLocalVoiceFramed< T >, 118
  - DebugLostPercent, 121
  - FramesLost, 121
  - FramesReceived, 121
  - FramesSent, 121
  - FramesSentBytes, 121
  - LoadBalancingTransport, 59
  - LocalVoices, 122
  - LocalVoicesInChannel, 120
  - OnRemoteVoiceInfoAction, 122
  - RemoteVoiceInfoDelegate, 120
  - RemoteVoiceInfos, 122
  - RemoveLocalVoice, 120
  - RoundTripTime, 122
  - RoundTripTimeVariance, 122
  - Service, 121
  - SuppressInfoDuplicateWarning, 122
  - VoiceConnection, 129
- VoiceComponent, 123
- VoiceConnection, 123
  - AutoCreateSpeakerIfNotFound, 127
  - BestRegionSummaryInPreferences, 127
  - ClientState, 128
  - ConnectUsingSettings, 125
  - Dispatch, 125
  - FramesLostPercent, 128
  - FramesLostPerSecond, 128
  - FramesReceivedPerSecond, 128
  - GlobalPlaybackDelayMaxHard, 128
  - GlobalPlaybackDelayMaxSoft, 128
  - GlobalPlaybackDelayMinSoft, 129
  - InitRecorder, 126
  - Logger, 129
  - LogLevel, 129
  - MinimalTimeScaleToDispatchInFixedUpdate, 127
  - PrimaryRecorder, 129
  - RemoteVoiceAdded, 130
  - SetGlobalPlaybackDelaySettings, 126
  - SetPlaybackDelaySettings, 126
  - Settings, 127
  - SpeakerFactory, 127
  - SpeakerLinked, 130
  - SpeakerPrefab, 129
  - VoiceClient, 129
- VoiceDebugScript, 130
  - DisableVad, 131
  - ForceRecordingAndTransmission, 131
  - IncreaseLogLevels, 131
  - LocalDebug, 131
  - TestAudioClip, 131
  - TestUsingAudioClip, 131
- VoiceDetection
  - Recorder, 99
- VoiceDetectionDelayMs
  - Recorder, 100
- VoiceDetectionThreshold
  - Recorder, 100
- VoiceDetector
  - AudioUtil.VoiceLevelDetectCalibrate< T >, 144
  - ILocalVoiceAudio, 47
  - Recorder, 100
- VoiceDetectorCalibrate
  - ILocalVoiceAudio, 46
  - LocalVoiceAudio< T >, 65
  - LocalVoiceAudioDummy, 66
  - Recorder, 94
- VoiceDetectorCalibrating
  - ILocalVoiceAudio, 47
  - LocalVoiceAudio< T >, 65
  - Recorder, 100
- VoiceDetectorCalibration
  - AudioUtil.VoiceDetectorCalibration< T >, 135
- VoiceDetectorFloat
  - AudioUtil.VoiceDetectorFloat, 137
- VoiceDetectorShort
  - AudioUtil.VoiceDetectorShort, 138
- VoiceEvent, 138
  - Code, 138
- VoiceId
  - RemoteVoiceInfo, 101
- VoiceInfo, 139
  - Bitrate, 141
  - Channels, 141
  - CreateAudio, 140
  - CreateAudioOpus, 140
  - FPS, 141
  - FrameDurationSamples, 141
  - FrameDurationUs, 141
  - FrameSize, 141
  - Height, 142
  - KeyFrameInt, 142
  - SamplingRate, 142
  - UserData, 142
  - Width, 142
- VoiceLevelDetectCalibrate
  - AudioUtil.VoiceLevelDetectCalibrate< T >, 143
- VoiceLogger, 145
- VoiceRoomNameSuffix
  - PhotonVoiceNetwork, 82

## VoIP

- AudioSessionParametersPresets, [16](#)

## Voip

- POpusCodec.Enums, [12](#)

WebRtcAudioDsp, [145](#)

- SetOrSwitchAudioListener, [146](#)

- SetOrSwitchAudioOutCapture, [146](#)

WebRTCAudioLib, [147](#)WebRTCAudioProcessor, [147](#)

## Wideband

- POpusCodec.Enums, [11](#)

## Width

- VoiceInfo, [142](#)

## WorkInOfflineMode

- PhotonVoiceNetwork, [82](#)