

Photon Voice

v2.62

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 . . . . .	6
2.2.1.1 AudioSampleType . . . . .	6
2.2.1.2 Codec . . . . .	6
2.3 Photon.Voice.FMOD Namespace Reference . . . . .	6
2.4 Photon.Voice.Fusion Namespace Reference . . . . .	7
2.5 Photon.Voice.IOS Namespace Reference . . . . .	7
2.5.1 Enumeration Type Documentation . . . . .	7
2.5.1.1 AudioSessionCategory . . . . .	7
2.5.1.2 AudioSessionCategoryOption . . . . .	8
2.5.1.3 AudioSessionMode . . . . .	9
2.6 Photon.Voice.MacOS Namespace Reference . . . . .	10
2.7 Photon.Voice.PUN Namespace Reference . . . . .	10
2.8 Photon.Voice.PUN.UtilityScripts Namespace Reference . . . . .	10
2.9 Photon.Voice.Unity Namespace Reference . . . . .	11
2.10 Photon.Voice.Unity.FMOD Namespace Reference . . . . .	11
2.11 Photon.Voice.Unity.UtilityScripts Namespace Reference . . . . .	12
2.12 Photon.Voice.UWP Namespace Reference . . . . .	12
2.13 Photon.Voice.Windows Namespace Reference . . . . .	12
<b>3 Class Documentation</b>	<b>13</b>
3.1 AndroidAudioInAEC Class Reference . . . . .	13
3.2 AndroidAudioInParameters Struct Reference . . . . .	13
3.3 ArrayPool< T > Class Template Reference . . . . .	14
3.3.1 Detailed Description . . . . .	14
3.4 ArrayPoolSet< T > Class Template Reference . . . . .	14
3.4.1 Detailed Description . . . . .	14
3.5 ArraySegmentPool< T > Class Template Reference . . . . .	15
3.5.1 Detailed Description . . . . .	15
3.6 AudioChangesHandler Class Reference . . . . .	15
3.6.1 Detailed Description . . . . .	16
3.6.2 Member Data Documentation . . . . .	16
3.6.2.1 HandleDeviceChange . . . . .	16
3.6.2.2 HandleDeviceChangeAndroid . . . . .	16
3.6.2.3 HandleDeviceChangeIOS . . . . .	16
3.7 AudioClipWrapper Class Reference . . . . .	16
3.8 AudioDesc Class Reference . . . . .	17
3.9 AudioInChangeNotifier Class Reference . . . . .	17
3.9.1 Member Function Documentation . . . . .	17

---

---

3.9.1.1 Dispose() . . . . .	18
3.9.2 Property Documentation . . . . .	18
3.9.2.1 Error . . . . .	18
3.10 AudioInChangeNotifier Class Reference . . . . .	18
3.10.1 Member Function Documentation . . . . .	18
3.10.1.1 Dispose() . . . . .	18
3.10.2 Property Documentation . . . . .	19
3.10.2.1 Error . . . . .	19
3.11 AudioInChangeNotifierNotSupportedException Class Reference . . . . .	19
3.12 AudioInEnumerator Class Reference . . . . .	19
3.12.1 Detailed Description . . . . .	20
3.12.2 Member Function Documentation . . . . .	20
3.12.2.1 Dispose() . . . . .	20
3.12.2.2 Refresh() . . . . .	20
3.13 AudioInEnumerator Class Reference . . . . .	20
3.14 AudioInEnumerator Class Reference . . . . .	20
3.15 AudioInEnumerator Class Reference . . . . .	21
3.16 AudioInEnumerator Class Reference . . . . .	21
3.16.1 Detailed Description . . . . .	21
3.16.2 Member Function Documentation . . . . .	22
3.16.2.1 Dispose() . . . . .	22
3.16.2.2 Refresh() . . . . .	22
3.17 AudioInPusher Class Reference . . . . .	22
3.18 AudioInPusher Class Reference . . . . .	22
3.19 AudioInPusher Class Reference . . . . .	23
3.19.1 Property Documentation . . . . .	23
3.19.1.1 Channels . . . . .	23
3.20 AudioInReader Class Reference . . . . .	24
3.21 AudioInReader Class Reference . . . . .	24
3.22 AudioInReader< T > Class Template Reference . . . . .	24
3.22.1 Member Function Documentation . . . . .	25
3.22.1.1 Read() . . . . .	25
3.23 AudioOut< T > Class Template Reference . . . . .	25
3.24 AudioOutCapture Class Reference . . . . .	26
3.25 AudioOutDelayControl Class Reference . . . . .	26
3.26 AudioOutDelayControl Class Reference . . . . .	26
3.27 AudioOutDummy< T > Class Template Reference . . . . .	26
3.28 AudioOutEvent< T > Class Template Reference . . . . .	27
3.29 AudioSessionParameters Struct Reference . . . . .	27
3.30 AudioSessionParametersPresets Class Reference . . . . .	28
3.30.1 Member Data Documentation . . . . .	28
3.30.1.1 Game . . . . .	28

---

---

3.30.1.2 VoIP . . . . .	28
3.31 AudioSyncBuffer< T > Class Template Reference . . . . .	28
3.32 AudioUtil Class Reference . . . . .	29
3.32.1 Detailed Description . . . . .	30
3.32.2 Member Function Documentation . . . . .	30
3.32.2.1 Convert() [1/2] . . . . .	30
3.32.2.2 Convert() [2/2] . . . . .	31
3.32.2.3 ForceToStereo< T >() . . . . .	31
3.32.2.4 Resample< T >() . . . . .	31
3.32.2.5 ResampleAndConvert() [1/2] . . . . .	32
3.32.2.6 ResampleAndConvert() [2/2] . . . . .	32
3.33 BufferReaderPushAdapterAsyncPool< T > Class Template Reference . . . . .	33
3.33.1 Detailed Description . . . . .	33
3.33.2 Constructor & Destructor Documentation . . . . .	33
3.33.2.1 BufferReaderPushAdapterAsyncPool() . . . . .	33
3.33.3 Member Function Documentation . . . . .	33
3.33.3.1 Service() . . . . .	34
3.34 BufferReaderPushAdapterAsyncPoolFloatToShort Class Reference . . . . .	34
3.34.1 Detailed Description . . . . .	34
3.34.2 Constructor & Destructor Documentation . . . . .	34
3.34.2.1 BufferReaderPushAdapterAsyncPoolFloatToShort() . . . . .	34
3.34.3 Member Function Documentation . . . . .	35
3.34.3.1 Service() . . . . .	35
3.35 BufferReaderPushAdapterAsyncPoolShortToFloat Class Reference . . . . .	35
3.35.1 Detailed Description . . . . .	35
3.35.2 Constructor & Destructor Documentation . . . . .	36
3.35.2.1 BufferReaderPushAdapterAsyncPoolShortToFloat() . . . . .	36
3.35.3 Member Function Documentation . . . . .	36
3.35.3.1 Service() . . . . .	36
3.36 BufferReaderPushAdapterBase< T > Class Template Reference . . . . .	36
3.36.1 Detailed Description . . . . .	37
3.36.2 Constructor & Destructor Documentation . . . . .	37
3.36.2.1 BufferReaderPushAdapterBase() . . . . .	37
3.36.3 Member Function Documentation . . . . .	37
3.36.3.1 Dispose() . . . . .	37
3.36.3.2 Service() . . . . .	37
3.37 ByteStreamDecoder Class Reference . . . . .	38
3.37.1 Member Function Documentation . . . . .	38
3.37.1.1 Input() . . . . .	38
3.37.1.2 Open() . . . . .	38
3.38 ByteStreamEncoder Class Reference . . . . .	39
3.39 CaptureDevice Class Reference . . . . .	39

---

---

3.39.1 Member Function Documentation . . . . .	40
3.39.1.1 CleanUpAsync() . . . . .	40
3.39.1.2 SelectPreferredCameraStreamSettingAsync() . . . . .	40
3.39.1.3 StartRecordingAsync() . . . . .	40
3.39.1.4 StopRecordingAsync() . . . . .	41
3.39.2 Property Documentation . . . . .	41
3.39.2.1 CaptureSource . . . . .	41
3.40 ConnectAndJoin Class Reference . . . . .	41
3.41 VoiceClient.CreateOptions Struct Reference . . . . .	42
3.41.1 Member Data Documentation . . . . .	42
3.41.1.1 Default . . . . .	42
3.42 OpusCodec.Decoder< T > Class Template Reference . . . . .	42
3.42.1 Member Function Documentation . . . . .	43
3.42.1.1 Input() . . . . .	43
3.42.1.2 Open() . . . . .	43
3.43 RawCodec.Decoder< T > Class Template Reference . . . . .	43
3.43.1 Member Function Documentation . . . . .	44
3.43.1.1 Input() . . . . .	44
3.43.1.2 Open() . . . . .	44
3.44 DecoderConfigFrame Class Reference . . . . .	44
3.44.1 Detailed Description . . . . .	45
3.44.2 Member Function Documentation . . . . .	45
3.44.2.1 TryConfigure() . . . . .	45
3.45 DeviceEnumerator Class Reference . . . . .	45
3.46 DeviceEnumeratorBase Class Reference . . . . .	46
3.47 DeviceFeatures Class Reference . . . . .	46
3.48 DeviceInfo Struct Reference . . . . .	47
3.49 RecorderPreset.DSP Struct Reference . . . . .	47
3.50 OpusCodec.Encoder< T > Class Template Reference . . . . .	47
3.51 RawCodec.Encoder< T > Class Template Reference . . . . .	48
3.52 OpusCodec.EncoderFloat Class Reference . . . . .	48
3.53 OpusCodec.EncoderShort Class Reference . . . . .	49
3.54 OpusCodec.Factory Class Reference . . . . .	49
3.55 Flip Struct Reference . . . . .	49
3.56 FMODRecorderSetup Class Reference . . . . .	50
3.57 FrameBuffer Struct Reference . . . . .	50
3.58 FrameOut< T > Class Template Reference . . . . .	51
3.59 Framer< T > Class Template Reference . . . . .	51
3.59.1 Detailed Description . . . . .	51
3.59.2 Constructor & Destructor Documentation . . . . .	51
3.59.2.1 Framer() . . . . .	51
3.59.3 Member Function Documentation . . . . .	52

---

---

3.59.3.1 Frame() . . . . .	52
3.60 FramerResampler< T > Class Template Reference . . . . .	52
3.60.1 Member Function Documentation . . . . .	52
3.60.1.1 Frame() . . . . .	52
3.61 FusionVoiceClient Class Reference . . . . .	53
3.61.1 Member Data Documentation . . . . .	53
3.61.1.1 UseFusionAppSettings . . . . .	54
3.61.1.2 UseFusionAuthValues . . . . .	54
3.62 AudioUtil.GeneratorPusher< T > Class Template Reference . . . . .	54
3.62.1 Detailed Description . . . . .	55
3.62.2 Member Function Documentation . . . . .	55
3.62.2.1 SetCallback() . . . . .	55
3.63 AudioUtil.GeneratorReader< T > Class Template Reference . . . . .	55
3.63.1 Member Function Documentation . . . . .	56
3.63.1.1 Read() . . . . .	56
3.64 IAudioDesc Interface Reference . . . . .	56
3.64.1 Detailed Description . . . . .	56
3.64.2 Property Documentation . . . . .	56
3.64.2.1 Channels . . . . .	57
3.64.2.2 Error . . . . .	57
3.64.2.3 SamplingRate . . . . .	57
3.65 IAudioInChangeNotifier Interface Reference . . . . .	57
3.66 IAudioOut< T > Interface Template Reference . . . . .	57
3.67 IAudioPusher< T > Interface Template Reference . . . . .	58
3.67.1 Detailed Description . . . . .	58
3.67.2 Member Function Documentation . . . . .	58
3.67.2.1 SetCallback() . . . . .	58
3.68 IAudioReader< T > Interface Template Reference . . . . .	59
3.68.1 Detailed Description . . . . .	59
3.69 IDataReader< T > Interface Template Reference . . . . .	59
3.69.1 Detailed Description . . . . .	59
3.69.2 Member Function Documentation . . . . .	59
3.69.2.1 Read() . . . . .	59
3.70 IDecoder Interface Reference . . . . .	60
3.70.1 Detailed Description . . . . .	60
3.70.2 Member Function Documentation . . . . .	60
3.70.2.1 Input() . . . . .	60
3.70.2.2 Open() . . . . .	60
3.70.3 Property Documentation . . . . .	61
3.70.3.1 Error . . . . .	61
3.71 IDecoderDirect< B > Interface Template Reference . . . . .	61
3.71.1 Detailed Description . . . . .	61

---

3.71.2 Property Documentation . . . . .	61
3.71.2.1 Output . . . . .	61
3.72 IDeviceEnumerator Interface Reference . . . . .	62
3.73 IEncoder Interface Reference . . . . .	62
3.73.1 Detailed Description . . . . .	62
3.73.2 Member Function Documentation . . . . .	63
3.73.2.1 DequeueOutput() . . . . .	63
3.73.2.2 EndOfStream() . . . . .	63
3.73.2.3 GetPlatformAPI< I >() . . . . .	63
3.73.3 Property Documentation . . . . .	63
3.73.3.1 Error . . . . .	63
3.73.3.2 Output . . . . .	63
3.74 IEncoderDirect< B > Interface Template Reference . . . . .	64
3.74.1 Detailed Description . . . . .	64
3.74.2 Member Function Documentation . . . . .	64
3.74.2.1 Input() . . . . .	64
3.75 IEncoderDirectImage Interface Reference . . . . .	64
3.75.1 Detailed Description . . . . .	65
3.75.2 Property Documentation . . . . .	65
3.75.2.1 ImageFormat . . . . .	65
3.76 AudioUtil.ILevelMeter Interface Reference . . . . .	65
3.76.1 Detailed Description . . . . .	65
3.76.2 Member Function Documentation . . . . .	65
3.76.2.1 ResetAccumAvgPeakAmp() . . . . .	66
3.76.3 Property Documentation . . . . .	66
3.76.3.1 AccumAvgPeakAmp . . . . .	66
3.76.3.2 CurrentAvgAmp . . . . .	66
3.76.3.3 CurrentPeakAmp . . . . .	66
3.77 ILocalVoiceAudio Interface Reference . . . . .	66
3.77.1 Detailed Description . . . . .	67
3.77.2 Member Function Documentation . . . . .	67
3.77.2.1 VoiceDetectorCalibrate() . . . . .	67
3.77.3 Property Documentation . . . . .	67
3.77.3.1 LevelMeter . . . . .	67
3.77.3.2 VoiceDetector . . . . .	68
3.77.3.3 VoiceDetectorCalibrating . . . . .	68
3.78 ILogger Interface Reference . . . . .	68
3.79 ImageBufferInfo Struct Reference . . . . .	68
3.80 ImageBufferNative Class Reference . . . . .	69
3.81 ImageBufferNativeAlloc Class Reference . . . . .	69
3.82 ImageBufferNativeGCHandleBytes Class Reference . . . . .	69
3.83 ImageBufferNativeGCHandleSinglePlane Class Reference . . . . .	70

---

3.84 ImageBufferNativePool< T > Class Template Reference . . . . .	70
3.85 IProcessor< T > Interface Template Reference . . . . .	71
3.85.1 Detailed Description . . . . .	71
3.85.2 Member Function Documentation . . . . .	71
3.85.2.1 Process() . . . . .	71
3.86 IResettable Interface Reference . . . . .	71
3.87 IServiceable Interface Reference . . . . .	72
3.87.1 Detailed Description . . . . .	72
3.87.2 Member Function Documentation . . . . .	72
3.87.2.1 Service() . . . . .	72
3.88 AudioUtil.IVoiceDetector Interface Reference . . . . .	72
3.88.1 Detailed Description . . . . .	73
3.88.2 Property Documentation . . . . .	73
3.88.2.1 ActivityDelayMs . . . . .	73
3.88.2.2 Detected . . . . .	73
3.88.2.3 DetectedTime . . . . .	73
3.88.2.4 On . . . . .	74
3.88.2.5 Threshold . . . . .	74
3.88.3 Event Documentation . . . . .	74
3.88.3.1 OnDetected . . . . .	74
3.89 IVoiceTransport Interface Reference . . . . .	74
3.90 AudioUtil.LevelMeter< T > Class Template Reference . . . . .	74
3.90.1 Detailed Description . . . . .	75
3.90.2 Member Function Documentation . . . . .	75
3.90.2.1 Process() . . . . .	75
3.90.2.2 ResetAccumAvgPeakAmp() . . . . .	76
3.91 AudioUtil.LevelMeterDummy Class Reference . . . . .	76
3.91.1 Detailed Description . . . . .	76
3.91.2 Member Function Documentation . . . . .	76
3.91.2.1 ResetAccumAvgPeakAmp() . . . . .	76
3.92 AudioUtil.LevelMeterFloat Class Reference . . . . .	77
3.92.1 Detailed Description . . . . .	77
3.92.2 Constructor & Destructor Documentation . . . . .	77
3.92.2.1 LevelMeterFloat() . . . . .	77
3.93 AudioUtil.LevelMeterShort Class Reference . . . . .	77
3.93.1 Detailed Description . . . . .	78
3.93.2 Constructor & Destructor Documentation . . . . .	78
3.93.2.1 LevelMeterShort() . . . . .	78
3.94 LoadBalancingTransport Class Reference . . . . .	78
3.94.1 Detailed Description . . . . .	79
3.94.2 Constructor & Destructor Documentation . . . . .	79
3.94.2.1 LoadBalancingTransport() . . . . .	79

---

3.94.3 Member Function Documentation . . . . .	80
3.94.3.1 Dispose() . . . . .	80
3.94.3.2 Service() . . . . .	80
3.94.4 Property Documentation . . . . .	80
3.94.4.1 VoiceClient . . . . .	80
3.95 LoadBalancingTransport2 Class Reference . . . . .	80
3.95.1 Detailed Description . . . . .	81
3.96 LocalVoice Class Reference . . . . .	81
3.96.1 Detailed Description . . . . .	82
3.96.2 Member Function Documentation . . . . .	82
3.96.2.1 RemoveSelf() . . . . .	83
3.96.3 Property Documentation . . . . .	83
3.96.3.1 DebugEchoMode . . . . .	83
3.96.3.2 Encrypt . . . . .	83
3.96.3.3 FEC . . . . .	83
3.96.3.4 Fragment . . . . .	83
3.96.3.5 FramesSent . . . . .	83
3.96.3.6 FramesSentBytes . . . . .	84
3.96.3.7 FramesSentFragmented . . . . .	84
3.96.3.8 FramesSentFragments . . . . .	84
3.96.3.9 Info . . . . .	84
3.96.3.10 InterestGroup . . . . .	84
3.96.3.11 IsCurrentlyTransmitting . . . . .	84
3.96.3.12 LocalUserServiceable . . . . .	85
3.96.3.13 Reliable . . . . .	85
3.96.3.14 SendSpacingProfileMax . . . . .	85
3.96.3.15 TargetPlayers . . . . .	85
3.96.3.16 TransmitEnabled . . . . .	85
3.97 LocalVoiceAudio< T > Class Template Reference . . . . .	85
3.97.1 Detailed Description . . . . .	86
3.97.2 Member Function Documentation . . . . .	86
3.97.2.1 VoiceDetectorCalibrate() . . . . .	86
3.97.3 Property Documentation . . . . .	87
3.97.3.1 VoiceDetectorCalibrating . . . . .	87
3.98 LocalVoiceAudioDummy Class Reference . . . . .	87
3.98.1 Detailed Description . . . . .	87
3.98.2 Member Function Documentation . . . . .	87
3.98.2.1 VoiceDetectorCalibrate() . . . . .	87
3.98.3 Member Data Documentation . . . . .	88
3.98.3.1 Dummy . . . . .	88
3.99 LocalVoiceAudioFloat Class Reference . . . . .	88
3.99.1 Detailed Description . . . . .	88

---

3.100 LocalVoiceAudioShort Class Reference . . . . .	88
3.100.1 Detailed Description . . . . .	88
3.101 LocalVoiceFramed< T > Class Template Reference . . . . .	89
3.101.1 Detailed Description . . . . .	89
3.101.2 Member Function Documentation . . . . .	90
3.101.2.1 AddPostProcessor() . . . . .	90
3.101.2.2 AddPreProcessor() . . . . .	91
3.101.2.3 ClearProcessors() . . . . .	91
3.101.2.4 Dispose() . . . . .	91
3.101.2.5 PushData() . . . . .	91
3.101.2.6 PushDataAsync() . . . . .	92
3.101.2.7 RemoveProcessor() . . . . .	92
3.101.3 Property Documentation . . . . .	92
3.101.3.1 BufferFactory . . . . .	92
3.101.3.2 PushDataAsyncReady . . . . .	92
3.102 Logger Class Reference . . . . .	92
3.103 MicAmplifier Class Reference . . . . .	93
3.104 MicAmplifierFloat Class Reference . . . . .	93
3.105 MicAmplifierShort Class Reference . . . . .	93
3.106 MicrophonePermission Class Reference . . . . .	94
3.106.1 Detailed Description . . . . .	94
3.107 MicWrapper Class Reference . . . . .	94
3.108 MicWrapperPusher Class Reference . . . . .	95
3.109 MicWrapperPusherOnAudioFilterRead Class Reference . . . . .	95
3.110 MonoPInvokeCallbackAttribute Class Reference . . . . .	95
3.111 ObjectFactory< TType, TInfo > Interface Template Reference . . . . .	95
3.111.1 Detailed Description . . . . .	95
3.112 ObjectPool< TType, TInfo > Class Template Reference . . . . .	96
3.112.1 Detailed Description . . . . .	96
3.112.2 Constructor & Destructor Documentation . . . . .	97
3.112.2.1 ObjectPool() . . . . .	97
3.112.3 Member Function Documentation . . . . .	97
3.112.3.1 Dispose() . . . . .	97
3.112.3.2 Free() [1/2] . . . . .	97
3.112.3.3 Free() [2/2] . . . . .	98
3.112.3.4 New() [1/2] . . . . .	98
3.112.3.5 New() [2/2] . . . . .	98
3.112.4 Property Documentation . . . . .	99
3.112.4.1 Info . . . . .	99
3.113 OpusCodec Class Reference . . . . .	99
3.114 PhotonAppSettings Class Reference . . . . .	99
3.114.1 Detailed Description . . . . .	100

---

---

3.114.2 Member Function Documentation . . . . .	100
3.114.2.1 ToString() . . . . .	100
3.114.2.2 UseCloud() . . . . .	100
3.115 PhotonVoiceCreatedParams Class Reference . . . . .	100
3.116 PhotonVoiceLagSimulationGui Class Reference . . . . .	101
3.117 PhotonVoiceStatsGui Class Reference . . . . .	101
3.117.1 Detailed Description . . . . .	101
3.118 PhotonVoiceView Class Reference . . . . .	101
3.118.1 Detailed Description . . . . .	102
3.118.2 Property Documentation . . . . .	102
3.118.2.1 IsRecording . . . . .	102
3.118.2.2 IsSpeaking . . . . .	102
3.118.2.3 RecorderInUse . . . . .	102
3.118.2.4 SpeakerInUse . . . . .	102
3.119 ImageBufferNative.PlaneSet Struct Reference . . . . .	103
3.120 Platform Class Reference . . . . .	103
3.121 AudioOutDelayControl.PlayDelayConfig Struct Reference . . . . .	103
3.121.1 Member Data Documentation . . . . .	104
3.121.1.1 Default . . . . .	104
3.122 PunVoiceClient Class Reference . . . . .	104
3.122.1 Detailed Description . . . . .	105
3.122.2 Member Data Documentation . . . . .	105
3.122.2.1 VoiceRoomNameSuffix . . . . .	105
3.122.3 Property Documentation . . . . .	105
3.122.3.1 Instance . . . . .	105
3.122.3.2 UsePunAppSettings . . . . .	105
3.122.3.3 UsePunAuthValues . . . . .	106
3.123 RawCodec Class Reference . . . . .	106
3.124 Recorder Class Reference . . . . .	106
3.124.1 Detailed Description . . . . .	108
3.124.2 Member Function Documentation . . . . .	108
3.124.2.1 ResetLocalAudio() . . . . .	108
3.124.2.2 RestartRecording() . . . . .	109
3.124.2.3 SetAndroidNativeMicrophoneSettings() . . . . .	109
3.124.2.4 SetiosAudioSessionParameters() [1/2] . . . . .	109
3.124.2.5 SetiosAudioSessionParameters() [2/2] . . . . .	110
3.124.2.6 VoiceDetectorCalibrate() . . . . .	110
3.124.3 Property Documentation . . . . .	110
3.124.3.1 AudioClip . . . . .	110
3.124.3.2 Bitrate . . . . .	110
3.124.3.3 DebugEchoMode . . . . .	111
3.124.3.4 Encrypt . . . . .	111

---

---

3.124.3.5 FrameDuration . . . . .	111
3.124.3.6 InputFactory . . . . .	111
3.124.3.7 InterestGroup . . . . .	111
3.124.3.8 IsCurrentlyTransmitting . . . . .	111
3.124.3.9 LevelMeter . . . . .	112
3.124.3.10 LoopAudioClip . . . . .	112
3.124.3.11 MicrophoneType . . . . .	112
3.124.3.12 RecordingEnabled . . . . .	112
3.124.3.13 RecordWhenJoined . . . . .	112
3.124.3.14 ReliableMode . . . . .	112
3.124.3.15 SamplingRate . . . . .	113
3.124.3.16 SourceType . . . . .	113
3.124.3.17 StopRecordingWhenPaused . . . . .	113
3.124.3.18 TargetPlayers . . . . .	113
3.124.3.19 TransmitEnabled . . . . .	113
3.124.3.20 UseMicrophoneTypeFallback . . . . .	113
3.124.3.21 UseOnAudioFilterRead . . . . .	114
3.124.3.22 UserData . . . . .	114
3.124.3.23 VoiceDetection . . . . .	114
3.124.3.24 VoiceDetectionDelayMs . . . . .	114
3.124.3.25 VoiceDetectionThreshold . . . . .	114
3.124.3.26 VoiceDetector . . . . .	114
3.124.3.27 VoiceDetectorCalibrating . . . . .	115
3.125 RecorderPreset Class Reference . . . . .	115
3.126 RemoteVoiceInfo Class Reference . . . . .	115
3.126.1 Detailed Description . . . . .	116
3.126.2 Property Documentation . . . . .	116
3.126.2.1 ChannelId . . . . .	116
3.126.2.2 Info . . . . .	116
3.126.2.3 PlayerId . . . . .	116
3.126.2.4 Voiceld . . . . .	116
3.127 RemoteVoiceLink Class Reference . . . . .	116
3.128 RemoteVoiceOptions Struct Reference . . . . .	117
3.128.1 Detailed Description . . . . .	117
3.128.2 Member Function Documentation . . . . .	117
3.128.2.1 SetOutput() [1/2] . . . . .	117
3.128.2.2 SetOutput() [2/2] . . . . .	118
3.128.3 Property Documentation . . . . .	118
3.128.3.1 Decoder . . . . .	118
3.128.3.2 OnRemoteVoiceRemoveAction . . . . .	118
3.129 AudioUtil.Resampler< T > Class Template Reference . . . . .	118
3.129.1 Detailed Description . . . . .	119

---

---

3.129.2 Constructor & Destructor Documentation . . . . .	119
3.129.2.1 Resampler() . . . . .	119
3.129.3 Member Function Documentation . . . . .	119
3.129.3.1 Process() . . . . .	119
3.130 SaveIncomingStreamToFile Class Reference . . . . .	120
3.131 SaveOutgoingStreamToFile Class Reference . . . . .	120
3.132 SendFrameParams Struct Reference . . . . .	120
3.133 RawCodec.ShortToFloat Class Reference . . . . .	120
3.134 Speaker Class Reference . . . . .	120
3.134.1 Member Function Documentation . . . . .	121
3.134.1.1 RestartPlayback() . . . . .	121
3.134.2 Property Documentation . . . . .	122
3.134.2.1 IsLinked . . . . .	122
3.134.2.2 IsPlaying . . . . .	122
3.134.2.3 Lag . . . . .	122
3.134.2.4 OnRemoteVoiceRemoveAction . . . . .	122
3.134.2.5 PlayDelay . . . . .	122
3.134.2.6 PlayDelayConfig . . . . .	123
3.135 SpeakerAudioFilterRead Class Reference . . . . .	123
3.136 SpeakerFMOD Class Reference . . . . .	123
3.137 ImageBufferInfo.StrideSet Struct Reference . . . . .	123
3.138 AudioUtil.TempoUp< T > Class Template Reference . . . . .	124
3.139 TestTone Class Reference . . . . .	124
3.140 TimerWorker Class Reference . . . . .	124
3.141 AudioUtil.ToneAudioPusher< T > Class Template Reference . . . . .	124
3.141.1 Detailed Description . . . . .	125
3.141.2 Constructor & Destructor Documentation . . . . .	125
3.141.2.1 ToneAudioPusher() . . . . .	125
3.142 AudioUtil.ToneAudioReader< T > Class Template Reference . . . . .	125
3.142.1 Detailed Description . . . . .	126
3.142.2 Constructor & Destructor Documentation . . . . .	126
3.142.2.1 ToneAudioReader() . . . . .	126
3.143 UnityAudioOut Class Reference . . . . .	126
3.144 UnityLogger Class Reference . . . . .	127
3.145 UnityMicrophone Class Reference . . . . .	127
3.145.1 Detailed Description . . . . .	127
3.146 UnityVoiceClient Class Reference . . . . .	127
3.146.1 Detailed Description . . . . .	128
3.146.2 Member Function Documentation . . . . .	128
3.146.2.1 ConnectUsingSettings() . . . . .	128
3.146.3 Member Data Documentation . . . . .	128
3.146.3.1 UseVoiceAppSettings . . . . .	129

---

---

3.147 UnsupportedCodecException Class Reference . . . . .	129
3.147.1 Detailed Description . . . . .	129
3.147.2 Constructor & Destructor Documentation . . . . .	129
3.147.2.1 UnsupportedCodecException() . . . . .	129
3.148 UnsupportedPlatformException Class Reference . . . . .	129
3.148.1 Detailed Description . . . . .	130
3.148.2 Constructor & Destructor Documentation . . . . .	130
3.148.2.1 UnsupportedPlatformException() . . . . .	130
3.149 UnsupportedSampleTypeException Class Reference . . . . .	130
3.149.1 Detailed Description . . . . .	131
3.149.2 Constructor & Destructor Documentation . . . . .	131
3.149.2.1 UnsupportedSampleTypeException() . . . . .	131
3.150 OpusCodec.Util Class Reference . . . . .	131
3.151 Utility Class Reference . . . . .	131
3.152 VideoInEnumerator Class Reference . . . . .	131
3.153 VideoInEnumerator Class Reference . . . . .	132
3.154 VoiceClient Class Reference . . . . .	132
3.154.1 Detailed Description . . . . .	134
3.154.2 Constructor & Destructor Documentation . . . . .	134
3.154.2.1 VoiceClient() . . . . .	134
3.154.3 Member Function Documentation . . . . .	134
3.154.3.1 CreateLocalVoice() . . . . .	134
3.154.3.2 CreateLocalVoiceAudioFromSource() . . . . .	135
3.154.3.3 CreateLocalVoiceVideo() . . . . .	135
3.154.3.4 LocalVoicesInChannel() . . . . .	136
3.154.3.5 RemoteVoiceInfoDelegate() . . . . .	136
3.154.3.6 RemoveLocalVoice() . . . . .	136
3.154.3.7 Service() . . . . .	136
3.154.4 Property Documentation . . . . .	137
3.154.4.1 DebugLostPercent . . . . .	137
3.154.4.2 EventsLost . . . . .	137
3.154.4.3 FramesFragPart . . . . .	137
3.154.4.4 FramesLate . . . . .	137
3.154.4.5 FramesLost . . . . .	137
3.154.4.6 FramesReceived . . . . .	138
3.154.4.7 FramesReceivedFEC . . . . .	138
3.154.4.8 FramesReceivedFragmented . . . . .	138
3.154.4.9 FramesReceivedFragments . . . . .	138
3.154.4.10 FramesRecovered . . . . .	138
3.154.4.11 FramesSent . . . . .	138
3.154.4.12 FramesSentBytes . . . . .	139
3.154.4.13 FramesTryFEC . . . . .	139

---

3.154.4.14 LocalVoices . . . . .	139
3.154.4.15 OnRemoteVoiceInfoAction . . . . .	139
3.154.4.16 RemoteVoiceInfos . . . . .	139
3.154.4.17 RoundTripTime . . . . .	139
3.154.4.18 RoundTripTimeVariance . . . . .	140
3.154.4.19 SuppressInfoDuplicateWarning . . . . .	140
3.155 VoiceComponent Class Reference . . . . .	140
3.156 VoiceComponentImpl Class Reference . . . . .	140
3.157 VoiceConnection Class Reference . . . . .	141
3.157.1 Detailed Description . . . . .	142
3.157.2 Member Function Documentation . . . . .	142
3.157.2.1 AddSpeaker() . . . . .	142
3.157.2.2 ConnectUsingSettings() . . . . .	143
3.157.2.3 InstantiateSpeakerPrefab() . . . . .	143
3.157.3 Member Data Documentation . . . . .	144
3.157.3.1 ChannelAudio . . . . .	144
3.157.3.2 ChannelVideo . . . . .	144
3.157.3.3 Settings . . . . .	144
3.157.3.4 UsePrimaryRecorder . . . . .	144
3.157.4 Property Documentation . . . . .	144
3.157.4.1 BestRegionSummaryInPreferences . . . . .	144
3.157.4.2 ClientState . . . . .	145
3.157.4.3 FramesLostPercent . . . . .	145
3.157.4.4 FramesLostPerSecond . . . . .	145
3.157.4.5 FramesReceivedPerSecond . . . . .	145
3.157.4.6 PrimaryRecorder . . . . .	145
3.157.4.7 SpeakerPrefab . . . . .	145
3.157.4.8 VoiceClient . . . . .	146
3.157.5 Event Documentation . . . . .	146
3.157.5.1 RemoteVoiceAdded . . . . .	146
3.157.5.2 SpeakerLinked . . . . .	146
3.158 VoiceCreateOptions Struct Reference . . . . .	146
3.158.1 Detailed Description . . . . .	147
3.158.2 Member Data Documentation . . . . .	147
3.158.2.1 DebugEchoMode . . . . .	147
3.158.2.2 Encoder . . . . .	147
3.158.2.3 Encrypt . . . . .	147
3.158.2.4 EventBufSize . . . . .	147
3.158.2.5 FEC . . . . .	147
3.158.2.6 Fragment . . . . .	148
3.158.2.7 InterestGroup . . . . .	148
3.158.2.8 Reliable . . . . .	148

---

---

3.158.2.9 TargetPlayers . . . . .	148
3.159 VoiceDebugScript Class Reference . . . . .	148
3.159.1 Detailed Description . . . . .	149
3.159.2 Member Data Documentation . . . . .	149
3.159.2.1 DisableVad . . . . .	149
3.159.2.2 ForceRecordingAndTransmission . . . . .	149
3.159.2.3 IncreaseLogLevels . . . . .	149
3.159.2.4 LocalDebug . . . . .	150
3.159.2.5 TestAudioClip . . . . .	150
3.159.2.6 TestUsingAudioClip . . . . .	150
3.160 AudioUtil.VoiceDetector< T > Class Template Reference . . . . .	150
3.160.1 Detailed Description . . . . .	151
3.160.2 Member Function Documentation . . . . .	151
3.160.2.1 Process() . . . . .	151
3.160.3 Property Documentation . . . . .	151
3.160.3.1 ActivityDelayMs . . . . .	152
3.160.3.2 Detected . . . . .	152
3.160.3.3 DetectedTime . . . . .	152
3.160.3.4 On . . . . .	152
3.160.3.5 Threshold . . . . .	152
3.160.4 Event Documentation . . . . .	152
3.160.4.1 OnDetected . . . . .	152
3.161 AudioUtil.VoiceDetectorCalibration< T > Class Template Reference . . . . .	153
3.161.1 Detailed Description . . . . .	153
3.161.2 Constructor & Destructor Documentation . . . . .	153
3.161.2.1 VoiceDetectorCalibration() . . . . .	153
3.161.3 Member Function Documentation . . . . .	154
3.161.3.1 Calibrate() . . . . .	154
3.161.3.2 Process() . . . . .	154
3.162 AudioUtil.VoiceDetectorDummy Class Reference . . . . .	155
3.162.1 Detailed Description . . . . .	155
3.163 AudioUtil.VoiceDetectorFloat Class Reference . . . . .	155
3.163.1 Detailed Description . . . . .	155
3.163.2 Constructor & Destructor Documentation . . . . .	155
3.163.2.1 VoiceDetectorFloat() . . . . .	155
3.164 AudioUtil.VoiceDetectorShort Class Reference . . . . .	156
3.164.1 Detailed Description . . . . .	156
3.164.2 Constructor & Destructor Documentation . . . . .	156
3.164.2.1 VoiceDetectorShort() . . . . .	156
3.165 VoiceEvent Class Reference . . . . .	157
3.165.1 Member Data Documentation . . . . .	157
3.165.1.1 Code . . . . .	157

---

---

3.166 VoiceFollowClient Class Reference . . . . .	157
3.166.1 Detailed Description . . . . .	158
3.166.2 Member Function Documentation . . . . .	158
3.166.2.1 ConnectAndJoinRoom() . . . . .	158
3.166.2.2 Disconnect() . . . . .	158
3.166.3 Member Data Documentation . . . . .	158
3.166.3.1 AutoConnectAndJoin . . . . .	158
3.167 VoicelInfo Struct Reference . . . . .	159
3.167.1 Detailed Description . . . . .	159
3.167.2 Member Function Documentation . . . . .	160
3.167.2.1 CreateAudio() . . . . .	160
3.167.2.2 CreateAudioOpus() . . . . .	160
3.167.2.3 CreateVideo() . . . . .	161
3.167.3 Property Documentation . . . . .	161
3.167.3.1 Bitrate . . . . .	161
3.167.3.2 Channels . . . . .	162
3.167.3.3 FPS . . . . .	162
3.167.3.4 FrameDurationSamples . . . . .	162
3.167.3.5 FrameDurationUs . . . . .	162
3.167.3.6 FrameSize . . . . .	162
3.167.3.7 Height . . . . .	162
3.167.3.8 KeyFrameInt . . . . .	163
3.167.3.9 SamplingRate . . . . .	163
3.167.3.10 UserData . . . . .	163
3.167.3.11 Width . . . . .	163
3.168 AudioUtil.VoiceLevelDetectCalibrate< T > Class Template Reference . . . . .	163
3.168.1 Detailed Description . . . . .	164
3.168.2 Constructor & Destructor Documentation . . . . .	164
3.168.2.1 VoiceLevelDetectCalibrate() . . . . .	164
3.168.3 Member Function Documentation . . . . .	164
3.168.3.1 Calibrate() . . . . .	164
3.168.3.2 Process() . . . . .	165
3.168.4 Property Documentation . . . . .	165
3.168.4.1 LevelMeter . . . . .	165
3.168.4.2 VoiceDetector . . . . .	165
3.169 VoiceLogger Class Reference . . . . .	165
3.170 VoiceNetworkObject Class Reference . . . . .	166
3.170.1 Member Data Documentation . . . . .	166
3.170.1.1 IsRecording . . . . .	166
3.170.1.2 IsSpeaking . . . . .	167
3.170.2 Property Documentation . . . . .	167
3.170.2.1 RecorderInUse . . . . .	167

---

3.170.2.2 SpeakerInUse . . . . .	167
3.171 AudioUtil.WaveformAudioPusher< T > Class Template Reference . . . . .	167
3.171.1 Detailed Description . . . . .	168
3.172 AudioUtil.WaveformAudioReader< T > Class Template Reference . . . . .	168
3.172.1 Detailed Description . . . . .	168
3.173 WaveWriter Class Reference . . . . .	168
3.174 WebRtcAudioDsp Class Reference . . . . .	169
3.174.1 Member Data Documentation . . . . .	169
3.174.1.1 IsSupported . . . . .	169
3.175 WebRTCAudioLib Class Reference . . . . .	170
3.176 WebRTCAudioProcessor Class Reference . . . . .	170
3.177 WindowsAudioInPusher Class Reference . . . . .	171
<b>Index</b>	<b>173</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 [ArrayPool](#)  
*Pool of Arrays with components of type T, with ObjectPool info being the array's size.*
- class [ArrayPoolSet](#)  
*ArrayPool set of limited size.*
- class [ArraySegmentPool](#)  
*Set of ArrayPool's of N^2-sized arrays used to produce an ArraySegment with the requested size.*
- class [AudioDesc](#)
- class [AudioInChangeNotifierNotSupported](#)
- class [AudioInEnumeratorNotSupported](#)
- class [AudioOutDelayControl](#)
- class [AudioOutDummy](#)
- class [AudioSyncBuffer](#)
- class [AudioUtil](#)  
*Collection of Audio Utility functions and classes.*
- class [BufferReaderPushAdapterAsyncPool](#)  
*BufferReaderPushAdapter<T> implementation using asynchronous LocalVoiceFramed<T>.PushDataAsync.*
- class [BufferReaderPushAdapterAsyncPoolFloatToShort](#)  
*BufferReaderPushAdapter<T> implementation using asynchronous LocalVoiceFramed<T>.PushDataAsync, converting float samples to short.*
- class [BufferReaderPushAdapterAsyncPoolShortToFloat](#)  
*BufferReaderPushAdapter<T> implementation using asynchronous LocalVoiceFramed<T>.PushDataAsync, converting short samples to float.*
- class [BufferReaderPushAdapterBase](#)  
*Adapter base reading data from IDataReader<T>.Read and pushing it to LocalVoice.*
- class [ByteStreamDecoder](#)
- class [ByteStreamEncoder](#)
- class [DecoderConfigFrame](#)  
*Stores the config frame and prevents other frames decoding until the decoder is ready.*
- class [DeviceEnumeratorBase](#)

- class **DeviceEnumeratorNotSupported**
- class **DeviceEnumeratorSingleDevice**
- class **DeviceFeatures**
- struct **DeviceInfo**
- struct **Flip**
- struct **FrameBuffer**
- class **FrameOut**
- class **Framer**

*Utility class to re-frame packets.*
- class **FramerResampler**
- interface **IAudioDesc**

*Audio Source interface.*
- interface **IAudioInChangeNotifier**
- 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 **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**
- struct **ImageBufferInfo**
- class **ImageBufferNative**
- class **ImageBufferNativeAlloc**
- class **ImageBufferNativeGCHandleBytes**
- class **ImageBufferNativeGCHandleSinglePlane**
- class **ImageBufferNativePool**
- interface **IProcessor**

*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 **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< T > for float audio*
- class [LocalVoiceAudioShort](#)  
*Specialization of LocalVoiceAudio< T > for short audio*
- class [LocalVoiceFramed](#)  
*Typed re-framing LocalVoice*
- class [MonoPInvokeCallbackAttribute](#)
- interface [ObjectFactory](#)  
*Object factory with optional info useful in combination with ObjectPool< TType, TInfo >.*
- 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 [PhotonAppSettings](#)  
*Collection of connection-relevant settings, used internally by PhotonNetwork.ConnectUsingSettings.*
- class [PhotonTransportProtocol](#)
- class [Platform](#)
- 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).*
- struct [SendFrameParams](#)
- class [SpacingProfile](#)
- class [UnsupportedCodecException](#)  
*Exception thrown if an unsupported codec is encountered.*
- class [UnsupportedPlatformException](#)  
*Exception thrown if an unsupported platform is encountered.*
- class [UnsupportedSampleTypeException](#)  
*Exception thrown if an unsupported audio sample type is encountered.*
- class [Util](#)
- class [Utility](#)
- class [VideoInEnumeratorNotSupportedException](#)
- class [VoiceClient](#)  
*Voice client interact with other clients on network via IVoiceTransport.*
- struct [VoiceCreateOptions](#)  
*Used to initialize optional properties of the LocalVoice instance at creation time.*
- class [VoiceEvent](#)
- class [VoiceFollowClient](#)  
*This class can be used to automatically sync client states between Leader and Voice clients.*
- struct [VoicelInfo](#)  
*Describes stream properties.*
- class [WebRTCAudioLib](#)
- class [WebRTCAudioProcessor](#)

## Enumerations

- enum **CameraFacing**
- enum **AudioSampleType**

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

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

### 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 [VoicelInfo](#). Do not change the values of this Enum!

Enumerator

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

## 2.3 Photon.Voice.FMOD Namespace Reference

## Classes

- class [AudioInEnumerator](#)
- class [AudioInReader](#)
- class [AudioOut](#)
- class [AudioOutEvent](#)

## 2.4 Photon.Voice.Fusion Namespace Reference

### Classes

- class [FusionVoiceClient](#)
- class [VoiceNetworkObject](#)

### Typedefs

- using **PhotonAppSettings** = global::Fusion.Photon.Realtime.PhotonAppSettings
- using **LogLevel** = Photon.Voice.LogLevel

## 2.5 Photon.Voice.IOS Namespace Reference

### Classes

- class [AudioInChangeNotifier](#)
- class [AudioInPusher](#)
- class [AudioInReader](#)
- struct [AudioSessionParameters](#)
- class [AudioSessionParametersPresets](#)

### Enumerations

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

#### 2.5.1 Enumeration Type Documentation

##### 2.5.1.1 AudioSessionCategory

```
enum AudioSessionCategory [strong]
```

###### Enumerator

Ambient	Use this category for background sounds such as rain, car engine noise, etc. Mixes with other music. API_AVAILABLE(ios(3.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);
SoloAmbient	Use this category for background sounds. Other music will stop playing. API_AVAILABLE(ios(3.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);
Playback	Use this category for music tracks. API_AVAILABLE(ios(3.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);
Record	Use this category when recording audio. API_AVAILABLE(ios(3.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);

## Enumerator

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);

2.5.1.2 **AudioSessionCategoryOption**

```
enum AudioSessionCategoryOption [strong]
```

## Enumerator

MixWithOthers	This allows an application to set whether or not other active audio apps will be interrupted or mixed with when your app's audio session goes active. The typical cases are: (1) AVAudioSessionCategoryPlayAndRecord or AVAudioSessionCategoryMultiRoute this will default to false, but can be set to true. This would allow other applications to play in the background while an app had both audio input and output enabled (2) AVAudioSessionCategoryPlayback this will default to false, but can be set to true. This would allow other applications to play in the background, but an app will still be able to play regardless of the setting of the ringer switch (3) Other categories this defaults to false and cannot be changed (that is, the mix with others setting of these categories cannot be overridden. An application must be prepared for setting this property to fail as behaviour may change in future releases. If an application changes their category, they should reassert the option (it is not sticky across category changes). MixWithOthers is only valid with AVAudioSessionCategoryPlayAndRecord, AVAudioSessionCategoryPlayback, and AVAudioSessionCategoryMultiRoute
DuckOthers	This allows an application to set whether or not other active audio apps will be ducked when when your app's audio session goes active. An example of this is the Nike app, which provides periodic updates to its user (it reduces the volume of any music currently being played while it provides its status). This defaults to off. Note that the other audio will be ducked for as long as the current session is active. You will need to deactivate your audio session when you want full volume playback of the other audio. If your category is AVAudioSessionCategoryPlayback, AVAudioSessionCategoryPlayAndRecord, or AVAudioSessionCategoryMultiRoute, by default the audio session will be non-mixable and non-duking. Setting this option will also make your category mixable with others (AVAudioSessionCategoryOptionMixWithOthers will be set). DuckOthers is only valid with AVAudioSessionCategoryAmbient, AVAudioSessionCategoryPlayAndRecord, AVAudioSessionCategoryPlayback, and AVAudioSessionCategoryMultiRoute

## Enumerator

AllowBluetooth	This allows an application to change the default behaviour of some audio session categories with regards to showing bluetooth Hands-Free Profile (HFP) devices as available routes. The current category behavior is: (1) AVAudioSessionCategoryPlayAndRecord this will default to false, but can be set to true. This will allow a paired bluetooth HFP device to show up as an available route for input, while playing through the category-appropriate output (2) AVAudioSessionCategoryRecord this will default to false, but can be set to true. This will allow a paired bluetooth HFP device to show up as an available route for input (3) Other categories this defaults to false and cannot be changed (that is, enabling bluetooth for input in these categories is not allowed) An application must be prepared for setting this option to fail as behaviour may change in future releases. If an application changes their category or mode, they should reassert the override (it is not sticky across category and mode changes). AllowBluetooth is only valid with AVAudioSessionCategoryRecord and AVAudioSessionCategoryPlayAndRecord
DefaultToSpeaker	This allows an application to change the default behaviour of some audio session categories with regards to the audio route. The current category behavior is: (1) AVAudioSessionCategoryPlayAndRecord category this will default to false, but can be set to true. this will route to Speaker (instead of Receiver) when no other audio route is connected. (2) Other categories this defaults to false and cannot be changed (that is, the default to speaker setting of these categories cannot be overridden An application must be prepared for setting this property to fail as behaviour may change in future releases. If an application changes their category, they should reassert the override (it is not sticky across category and mode changes). DefaultToSpeaker is only valid with AVAudioSessionCategoryPlayAndRecord

## 2.5.1.3 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 <a href="#">Voice</a> over IP (VoIP) applications. Reduces the number of allowable audio routes to be only those that are appropriate for VoIP applications and may engage appropriate system-supplied signal processing. Has the side effect of setting AVAudioSessionCategoryOptionAllowBluetooth API_AVAILABLE(ios(5.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);
VideoRecording	Only valid with AVAudioSessionCategoryPlayAndRecord or AVAudioSessionCategoryRecord. Modifies the audio routing options and may engage appropriate system-supplied signal processing. API_AVAILABLE(ios(5.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);
Measurement	Appropriate for applications that wish to minimize the effect of system-supplied signal processing for input and/or output audio signals. API_AVAILABLE(ios(5.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);
MoviePlayback	Engages appropriate output signal processing for movie playback scenarios. Currently only applied during playback over built-in speaker. API_AVAILABLE(ios(6.0), watchos(2.0), tvos(9.0)) API_UNAVAILABLE(macos);

**Enumerator**

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

**2.6 Photon.Voice.MacOS Namespace Reference****Classes**

- class [AudioInChangeNotifier](#)
- class [AudioInEnumerator](#)  
*Enumerates microphones available on device.*
- class [AudioInPusher](#)
- class [AudioInReader](#)

**2.7 Photon.Voice.PUN Namespace Reference****Classes**

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

**2.8 Photon.Voice.PUN.UtilityScripts Namespace Reference****Classes**

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

## 2.9 Photon.Voice.Unity Namespace Reference

### Classes

- class [AndroidAudioInAEC](#)
- struct [AndroidAudioInParameters](#)
- class [AudioChangesHandler](#)

*This component is useful to handle audio device and config changes.*

- class [AudioClipWrapper](#)
- class [AudioInEnumerator](#)
- class [AudioOutCapture](#)
- class [Logger](#)
- class [MicWrapper](#)
- class [MicWrapperPusher](#)
- class [MicWrapperPusherOnAudioFilterRead](#)
- class [PhotonVoiceCreatedParams](#)
- class [Recorder](#)

*Component representing outgoing audio stream in scene.*

- class [RecorderPreset](#)
- class [RemoteVoiceLink](#)
- class [Speaker](#)
- class [SpeakerAudioFilterRead](#)
- class [TimerWorker](#)
- class [UnityAudioOut](#)
- class [UnityLogger](#)
- class [UnityMicrophone](#)

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

- class [UnityVoiceClient](#)

*Component that represents a [Voice](#) client and manages a simple [Unity](#) integration: a single [Recorder](#) and multiple remote speakers.*

- class [VideoInEnumerator](#)
- class [VoiceComponent](#)
- class [VoiceComponentImpl](#)
- class [VoiceConnection](#)

*Component that represents a [Voice](#) client.*

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

## 2.10 Photon.Voice.Unity.FMOD Namespace Reference

### Classes

- class [FMODRecorderSetup](#)
- class [SpeakerFMOD](#)

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

### Classes

- class [ConnectAndJoin](#)
- class [MicAmplifier](#)
- class [MicAmplifierFloat](#)
- class [MicAmplifierShort](#)
- class [MicrophonePermission](#)

*Helper to request Microphone permission on Android or iOS.*
- 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 [WaveWriter](#)

## 2.12 Photon.Voice.UWP Namespace Reference

### Classes

- class [AudioInEnumerator](#)
- class [AudioInPusher](#)
- class [CaptureDevice](#)
- class [DeviceEnumerator](#)
- class [VideoInEnumerator](#)

### Functions

- delegate void **MediaCaptureInitCompleted** (MediaCapture mediaCapture, bool ok)

## 2.13 Photon.Voice.Windows Namespace Reference

### Classes

- class [AudioInEnumerator](#)

*Enumerates microphones available on device.*
- class [WindowsAudioInPusher](#)

# 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, int optimalFrameSize)
- void **Reset** ()
- void **Dispose** ()

#### Properties

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

### 3.2 AndroidAudioInParameters Struct Reference

#### Public Attributes

- bool **EnableAEC**
- bool **EnableAGC**
- bool **EnableNS**

#### Static Public Attributes

- static [AndroidAudioInParameters](#) **Default** = new [AndroidAudioInParameters\(\)](#) { EnableAEC = true, EnableAGC = true, EnableNS = true }

### 3.3 ArrayPool< 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

- **ArrayPool** (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

##### 3.3.1 Detailed Description

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

#### Template Parameters

T	Array element type.
---	---------------------

### 3.4 ArrayPoolSet< T > Class Template Reference

[ArrayPool](#) set of limited size.

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

#### Public Member Functions

- **ArrayPoolSet** (int capacity, string name, int defaultInfo, int setSize)
- T[] **New** ()
- bool **Free** (T[] obj)
- T[] **New** (int info)
- bool **Free** (T[] obj, int info)
- void **Dispose** ()

##### 3.4.1 Detailed Description

[ArrayPool](#) set of limited size.

**Template Parameters**

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

## 3.5 ArraySegmentPool< T > Class Template Reference

Set of [ArrayPool](#)'s of  $N^2$ -sized arrays used to produce an [ArraySegment](#) with the requested size.

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

### Public Member Functions

- [\*\*ArraySegmentPool\*\*](#) (int capacity, string name, int defaultInfo)
- [\*\*ArraySegment< T > New\*\*](#) ()
- [\*\*ArraySegment< T > New\*\*](#) (int info)
- bool [\*\*Free\*\*](#) ([ArraySegment< T >](#) obj, int info)
- bool [\*\*Free\*\*](#) ([ArraySegment< T >](#) obj)
- void [\*\*Dispose\*\*](#) ()

### Static Public Member Functions

- static void [\*\*test\*\*](#) ()

#### 3.5.1 Detailed Description

Set of [ArrayPool](#)'s of  $N^2$ -sized arrays used to produce an [ArraySegment](#) with the requested size.

**Template Parameters**

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

## 3.6 AudioChangesHandler Class Reference

This component is useful to handle audio device and config changes.

Inherits [VoiceComponent](#).

### Public Attributes

- bool [\*\*HandleDeviceChange\*\*](#) = true
  - Try to react to device change notification when [Recorder](#) is started.*
- bool [\*\*HandleDeviceChangeIOS\*\*](#)
  - iOS: Try to react to device change notification when [Recorder](#) is started.*
- bool [\*\*HandleDeviceChangeAndroid\*\*](#)
  - Android: Try to react to device change notification when [Recorder](#) is started.*

## Protected Member Functions

- override void **Awake** ()

## Additional Inherited Members

### 3.6.1 Detailed Description

This component is useful to handle audio device and config changes.

### 3.6.2 Member Data Documentation

#### 3.6.2.1 HandleDeviceChange

```
bool HandleDeviceChange = true
```

Try to react to device change notification when [Recorder](#) is started.

#### 3.6.2.2 HandleDeviceChangeAndroid

```
bool HandleDeviceChangeAndroid
```

Android: Try to react to device change notification when [Recorder](#) is started.

#### 3.6.2.3 HandleDeviceChangeIOS

```
bool HandleDeviceChangeIOS
```

iOS: Try to react to device change notification when [Recorder](#) is started.

## 3.7 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]

## 3.8 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]

## 3.9 AudioInChangeNotifier Class Reference

Inherits [IAudioInChangeNotifier](#).

### Public Member Functions

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

*Disposes enumerator. Call it to free native resources.*

### Public Attributes

- bool **IsSupported** => true

## Properties

- string **Error** [get]  
*If not null, the enumerator is in invalid state.*

### 3.9.1 Member Function Documentation

### 3.9.1.1 Dispose()

```
void Dispose ( )
```

Disposes enumerator. Call it to free native resources.

## 3.9.2 Property Documentation

### 3.9.2.1 Error

```
string Error [get]
```

If not null, the enumerator is in invalid state.

## 3.10 AudioInChangeNotifier Class Reference

Inherits [IAudioInChangeNotifier](#).

### Public Member Functions

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

*Disposes enumerator. Call it to free native resources.*

### Public Attributes

- bool **IsSupported** => true

### Properties

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

## 3.10.1 Member Function Documentation

### 3.10.1.1 Dispose()

```
void Dispose ( )
```

Disposes enumerator. Call it to free native resources.

### 3.10.2 Property Documentation

#### 3.10.2.1 Error

```
string Error [get]
```

If not null, the enumerator is in invalid state.

## 3.11 AudioInChangeNotifierNotSupported Class Reference

Inherits [IAudioInChangeNotifier](#).

### Public Member Functions

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

### Public Attributes

- bool **IsSupported** => false

### Properties

- string **Error** [get ]

## 3.12 AudioInEnumerator Class Reference

Enumerates microphones available on device.

Inherits [DeviceEnumeratorBase](#).

### Public Member Functions

- **AudioInEnumerator** ([ILogger](#) logger)
- override void **Refresh** ()  
*Refreshes the microphones list.*
- override void **Dispose** ()  
*Disposes enumerator. Call it to free native resources.*

## Additional Inherited Members

### 3.12.1 Detailed Description

Enumerates microphones available on device.

### 3.12.2 Member Function Documentation

#### 3.12.2.1 Dispose()

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

Disposes enumerator. Call it to free native resources.

Implements [DeviceEnumeratorBase](#).

#### 3.12.2.2 Refresh()

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

Refreshes the microphones list.

Implements [DeviceEnumeratorBase](#).

## 3.13 AudioInEnumerator Class Reference

Inherits [DeviceEnumeratorBase](#).

### Public Member Functions

- **AudioInEnumerator** (FMODLib.System coreSystem, [ILogger](#) logger)
- override void **Refresh** ()
- override void **Dispose** ()

## Additional Inherited Members

## 3.14 AudioInEnumerator Class Reference

Inherits [DeviceEnumeratorBase](#).

## Public Member Functions

- **AudioInEnumerator** ([ILogger](#) logger)
- override void **Refresh** ()
- override void **Dispose** ()

## Public Attributes

- override bool **IsSupported** => false

## Properties

- override string **Error** [get]

## Additional Inherited Members

### 3.15 AudioInEnumerator Class Reference

Inherits [Device Enumerator](#).

## Public Member Functions

- **AudioInEnumerator** ([ILogger](#) logger)

## Additional Inherited Members

### 3.16 AudioInEnumerator Class Reference

Enumerates microphones available on device.

Inherits [Device Enumerator Base](#).

## Public Member Functions

- **AudioInEnumerator** ([ILogger](#) logger)
- override void **Refresh** ()  
*Refreshes the microphones list.*
- override void **Dispose** ()  
*Disposes enumerator. Call it to free native resources.*

## Additional Inherited Members

### 3.16.1 Detailed Description

Enumerates microphones available on device.

### 3.16.2 Member Function Documentation

#### 3.16.2.1 Dispose()

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

Disposes enumerator. Call it to free native resources.

Implements [DeviceEnumeratorBase](#).

#### 3.16.2.2 Refresh()

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

Refreshes the microphones list.

Implements [DeviceEnumeratorBase](#).

## 3.17 AudioInPusher Class Reference

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

### Public Member Functions

- [AudioInPusher \(AudioSessionParameters sessParam, ILogger logger\)](#)
- void [SetCallback \(Action< float\[\]> callback, ObjectFactory< float\[\], int > bufferFactory, int optimalFrameSize\)](#)
- void [Reset \(\)](#)
- void [Dispose \(\)](#)

### Properties

- int [Channels \[get\]](#)
- int [SamplingRate \[get\]](#)
- string [Error \[get\]](#)

## 3.18 AudioInPusher Class Reference

Inherits [IAudioPusher< float >](#).

## Public Member Functions

- **AudioInPusher** (int deviceID, [ILogger](#) logger)
- void **SetCallback** (Action< float[]> callback, [ObjectFactory](#)< float[], int > bufferFactory, int optimalFrameSize)
- void **Dispose** ()

## Properties

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

## 3.19 AudioInPusher Class Reference

Inherits [IAudioPusher< short >](#).

## Public Member Functions

- **AudioInPusher** ([ILogger](#) logger, int samplingRate, int channels, string deviceID)
- void **SetCallback** (Action< short[]> callback, [ObjectFactory](#)< short[], int > bufferFactory, int optimalFrameSize)
- **ArraySegment< byte > DequeueOutput** (out FrameFlags flags)
- void **EndOfStream** ()
- **I GetPlatformAPI< I >** ()
- void **Dispose** ()

## Properties

- int **SamplingRate** [get]
- int **Channels** [get]  
*Number of channels in the audio signal.*
- string **Error** [get]
- bool **ErrorAccess** [get]

### 3.19.1 Property Documentation

#### 3.19.1.1 Channels

```
int Channels [get]
```

Number of channels in the audio signal.

## 3.20 AudioInReader Class Reference

Inherits [IAudioReader< float >](#), and [IResettable](#).

### Public Member Functions

- **AudioInReader** ([AudioSessionParameters](#) sessParam, [ILogger](#) logger)
- void **Reset** ()
- void **Dispose** ()
- bool **Read** ([float\[\]](#) buf)

### Properties

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

## 3.21 AudioInReader Class Reference

Inherits [IAudioReader< float >](#).

### Public Member Functions

- **AudioInReader** (int deviceID, [ILogger](#) logger)
- void **Dispose** ()
- bool **Read** ([float\[\]](#) buf)

### Properties

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

## 3.22 AudioInReader< T > Class Template Reference

Inherits [IAudioReader< T >](#).

### Public Member Functions

- **AudioInReader** ([FMODLib.System](#) coreSystem, int device, int suggestedFrequency, [ILogger](#) logger)
- void **Dispose** ()
- bool **Read** ([T\[\]](#) readBuf)

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

## Public Attributes

- bool **isRecording**

## Properties

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

## 3.22.1 Member Function Documentation

### 3.22.1.1 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 >](#).

## 3.23 AudioOut< T > Class Template Reference

Inherits [AudioOutDelayControl< T >](#).

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

## Public Member Functions

- **AudioOut** (FMODLib.System coreSystem, PlayDelayConfig playDelayConfig, [ILogger](#) logger, string logPrefix, bool debugInfo)
- override void **OutCreate** (int samplingRate, int channels, int bufferSamples)
- override void **OutStart** ()
- override void **OutWrite** (T[ ] frame, int offsetSamples)
- override void **Stop** ()

## Protected Attributes

- int **channels**
- int **frequency**

## Properties

- FMODLib.Sound **Sound** [get]
- FMODLib.Channel **Channel** [get]
- override long **OutPos** [get]
- string **Error** [get]

## 3.24 AudioOutCapture Class Reference

Inherits MonoBehaviour.

## Events

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

## 3.25 AudioOutDelayControl Class Reference

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

## Classes

- struct [PlayDelayConfig](#)

## 3.26 AudioOutDelayControl Class Reference

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

## Classes

- struct [PlayDelayConfig](#)

## 3.27 AudioOutDummy< T > Class Template Reference

Inherits [IAudioOut< T >](#).

## Public Member Functions

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

## Public Attributes

- bool **IsPlaying** => false
- int **Lag** => 0

## Additional Inherited Members

### 3.28 AudioOutEvent< T > Class Template Reference

Inherits [AudioOut< T >](#).

## Public Member Functions

- [AudioOutEvent](#) (FMODLib.System coreSystem, FMODLib.Studio.EventInstance fmodEvent, PlayDelayConfig playDelayConfig, [ILogger](#) logger, string logPrefix, bool debugInfo)
- override void **OutStart** ()
- override void **Stop** ()

## Properties

- override long **OutPos** [get]

## Additional Inherited Members

### 3.29 AudioSessionParameters Struct Reference

## Public Member Functions

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

## Public Attributes

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

## 3.30 AudioSessionParametersPresets Class Reference

### Static Public Attributes

- static `AudioSessionParameters Game`
- static `AudioSessionParameters VoIP`

#### 3.30.1 Member Data Documentation

##### 3.30.1.1 Game

`AudioSessionParameters Game [static]`

###### Initial value:

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

##### 3.30.1.2 VoIP

`AudioSessionParameters VoIP [static]`

###### Initial value:

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

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

## 3.31 AudioSyncBuffer< T > Class Template Reference

Inherits `AudioOutDelayControl< T >`.

### Public Member Functions

- `AudioSyncBuffer` (`PlayDelayConfig playDelayConfig, ILogger logger, string logPrefix, bool debugInfo`)
- override void `OutCreate` (`int frequency, int channels, int bufferSamples`)
- override void `OutStart` ()
- override void `OutWrite` (`T[] data, int offsetSamples`)
- override void `Stop` ()
- void `Read` (`T[] outBuf, int outChannels, int outSampleRate`)

## Public Attributes

- override long **OutPos** => readPosSamples

## 3.32 AudioUtil Class Reference

Collection of Audio [Utility](#) functions and classes.

### Classes

- class [GeneratorPusher](#)  
*IAudioPusher that provides a constant tone signal.*
- class [GeneratorReader](#)
- 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 [TempoUp](#)
- 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.*
- class [WaveformAudioPusher](#)  
*IAudioPusher that provides the given waveform.*
- class [WaveformAudioReader](#)  
*IAudioReader that provides the given waveform.*

## Static Public Member Functions

- static int **ToneToBuf**< T > (T[ ] buf, long timeSamples, int channels, double amp, double k, double phase← Mod=0)
  - static int **ToneToBuf**< T > (T[ ] buf, int offset, int length, long timeSamples, int channels, double amp, double k, double phaseMod=0)
  - static int **WaveformToBuf**< T > (T[ ] buf, T[ ] waveform, long timePos)
  - 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.*

### 3.32.1 Detailed Description

Collection of Audio [Utility](#) functions and classes.

### 3.32.2 Member Function Documentation

#### 3.32.2.1 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.

### 3.32.2.2 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.

### 3.32.2.3 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.

### 3.32.2.4 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.

**3.32.2.5 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.

**3.32.2.6 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.

## 3.33 BufferReaderPushAdapterAsyncPool< T > Class Template Reference

BufferReaderPushAdapter<T> implementation using asynchronous [LocalVoiceFramed<T>.PushDataAsync](#).

Inherits [BufferReaderPushAdapterBase< T >](#).

### Public Member Functions

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

### Additional Inherited Members

#### 3.33.1 Detailed Description

BufferReaderPushAdapter<T> implementation using asynchronous [LocalVoiceFramed<T>.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.

#### 3.33.2 Constructor & Destructor Documentation

##### 3.33.2.1 BufferReaderPushAdapterAsyncPool()

```
BufferReaderPushAdapterAsyncPool (
    IDatasource< T > reader )
```

Create a new BufferReaderPushAdapter instance

##### Parameters

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

#### 3.33.3 Member Function Documentation

### 3.33.3.1 Service()

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

Do the actual data read/push.

#### Parameters

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

Implements `BufferReaderPushAdapterBase< T >`.

## 3.34 BufferReaderPushAdapterAsyncPoolFloatToShort Class Reference

`BufferReaderPushAdapter<T>` implementation using asynchronous `LocalVoiceFramed<T>.PushDataAsync`, converting float samples to short.

Inherits `BufferReaderPushAdapterBase< float >`.

### Public Member Functions

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

### Additional Inherited Members

#### 3.34.1 Detailed Description

`BufferReaderPushAdapter<T>` implementation using asynchronous `LocalVoiceFramed<T>.PushDataAsync`, converting float samples to short.

This adapter works exactly like `BufferReaderPushAdapterAsyncPool<T>`, 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.

#### 3.34.2 Constructor & Destructor Documentation

##### 3.34.2.1 BufferReaderPushAdapterAsyncPoolFloatToShort()

```
BufferReaderPushAdapterAsyncPoolFloatToShort (
    IDataReader< float > reader )
```

Create a new BufferReaderPushAdapter instance

**Parameters**

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

### 3.34.3 Member Function Documentation

#### 3.34.3.1 Service()

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

Do the actual data read/push.

**Parameters**

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

Implements [BufferReaderPushAdapterBase< float >](#).

## 3.35 BufferReaderPushAdapterAsyncPoolShortToFloat Class Reference

BufferReaderPushAdapter<T> implementation using asynchronous [LocalVoiceFramed<T>.PushDataAsync](#), converting short samples to float.

Inherits [BufferReaderPushAdapterBase< short >](#).

### Public Member Functions

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

### Additional Inherited Members

#### 3.35.1 Detailed Description

BufferReaderPushAdapter<T> implementation using asynchronous [LocalVoiceFramed<T>.PushDataAsync](#), converting short samples to float.

This adapter works exactly like BufferReaderPushAdapterAsyncPool<T>, 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.

### 3.35.2 Constructor & Destructor Documentation

#### 3.35.2.1 `BufferReaderPushAdapterAsyncPoolShortToFloat()`

```
BufferReaderPushAdapterAsyncPoolShortToFloat (
    IDataReader< short > reader )
```

Create a new BufferReaderPushAdapter instance

##### Parameters

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

### 3.35.3 Member Function Documentation

#### 3.35.3.1 `Service()`

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

Do the actual data read/push.

##### Parameters

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

Implements `BufferReaderPushAdapterBase< short >`.

## 3.36 `BufferReaderPushAdapterBase< T >` Class Template Reference

Adapter base reading data from `IDataReader<T>.Read` and pushing it to `LocalVoice`.

Inherits `IServiceable`.

Inherited by `BufferReaderPushAdapterAsyncPool< T >`.

### Public Member Functions

- abstract void `Service (LocalVoice localVoice)`

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

*Create a new BufferReaderPushAdapterBase instance*
- void `Dispose ()`

*Release resources associated with this instance.*

## Protected Attributes

- `IDataReader< T > reader`

### 3.36.1 Detailed Description

Adapter base reading data from `IDataReader<T>.Read` and pushing it to `LocalVoice`.

Use this with a `LocalVoice` of same T type.

### 3.36.2 Constructor & Destructor Documentation

#### 3.36.2.1 BufferReaderPushAdapterBase()

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

Create a new `BufferReaderPushAdapterBase` instance

Parameters

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

### 3.36.3 Member Function Documentation

#### 3.36.3.1 Dispose()

```
void Dispose ( )
```

Release resources associated with this instance.

#### 3.36.3.2 Service()

```
abstract void Service (
    LocalVoice localVoice ) [pure virtual]
```

Do the actual data read/push.

**Parameters**

<i>localVoice</i>	<code>LocalVoice</code> instance to push data to.
-------------------	---

Implements [IServiceable](#).

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

## 3.37 ByteStreamDecoder Class Reference

Inherits [IDecoder](#).

### Public Member Functions

- delegate void **OutputDelegate** (ref `FrameBuffer` buf)
- **ByteStreamDecoder** (`OutputDelegate` output, Action onMissingFrame)
- void **Input** (ref `FrameBuffer` buf)
 

*Consumes the given encoded data.*
- void **Open** (`VoiceInfo` info)
 

*Open (initialize) the decoder.*
- void **Dispose** ()

### Public Attributes

- string **Error** => ""

### Additional Inherited Members

#### 3.37.1 Member Function Documentation

##### 3.37.1.1 Input()

```
void Input (
    ref FrameBuffer buf )
```

Consumes the given encoded data.

The callee can call `buf.Retain()` to prevent the caller from disposing the buffer. In this case, the callee should call `buf.Release()` when buffer is no longer needed.

Implements [IDecoder](#).

##### 3.37.1.2 Open()

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

Open (initialize) the decoder.

#### Parameters

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

Implements [IDecoder](#).

## 3.38 ByteStreamEncoder Class Reference

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

### Public Member Functions

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

### Public Attributes

- string **Error** => ""

### Properties

- Action< ArraySegment< byte >, FrameFlags > **Output** [ set ]

## 3.39 CaptureDevice Class Reference

### Public Types

- enum **Media**

### Public Member Functions

- **CaptureDevice** ([ILogger](#) logger, Media media, string deviceID)
- void **Initialize** ()
- void **InitializeAsync** ()
- async Task [CleanUpAsync](#) ()  
*Asynchronous method cleaning up resources and stopping recording if necessary.*
- async Task< IMediaEncodingProperties > [SelectPreferredCameraStreamSettingAsync](#) (MediaStreamType mediaStreamType, Func< IMediaEncodingProperties, bool > filterSettings)  
*Allow selection of camera settings.*
- async Task [StartRecordingAsync](#) (MediaEncodingProfile encodingProfile, Action< byte[], FrameFlags > encoderCallback)  
*Starts media recording asynchronously*
- async Task [StopRecordingAsync](#) ()  
*Stops recording asynchronously*

## Static Public Member Functions

- static async Task< bool > **CheckForRecordingDeviceAsync ()**

## Properties

- MediaCapture **CaptureSource** [get]  
*Creates url object from MediaCapture*

## Events

- EventHandler< MediaCaptureFailedEventArgs > **CaptureFailed**

### 3.39.1 Member Function Documentation

#### 3.39.1.1 CleanUpAsync()

```
async Task CleanUpAsync ( )
```

Asynchronous method cleaning up resources and stopping recording if necessary.

#### 3.39.1.2 SelectPreferredCameraStreamSettingAsync()

```
async Task<IMediaEncodingProperties> SelectPreferredCameraStreamSettingAsync (
    MediaStreamType mediaStreamType,
    Func< IMediaEncodingProperties, bool > filterSettings )
```

Allow selection of camera settings.

#### Parameters

<i>mediaStreamType</i>	Type of a the media stream.
<i>filterSettings</i>	A predicate function, which will be called to filter the correct settings.

#### 3.39.1.3 StartRecordingAsync()

```
async Task StartRecordingAsync (
    MediaEncodingProfile encodingProfile,
    Action< byte[ ], FrameFlags > encoderCallback )
```

Starts media recording asynchronously

**Parameters**

<i>encodingProfile</i>	Encoding profile used for the recording session
------------------------	---

**3.39.1.4 StopRecordingAsync()**

```
async Task StopRecordingAsync ( )
```

Stops recording asynchronously

**3.39.2 Property Documentation****3.39.2.1 CaptureSource**

```
MediaCapture CaptureSource [get]
```

Creates url object from MediaCapture

**3.40 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]

## 3.41 VoiceClient.CreateOptions Struct Reference

### Public Attributes

- byte **VoiceIDMin**
- byte **VoiceIDMax**

### Static Public Attributes

- static [CreateOptions Default](#)

### 3.41.1 Member Data Documentation

#### 3.41.1.1 Default

[CreateOptions Default](#) [static]

##### Initial value:

```
= new CreateOptions()
{
    VoiceIDMin = 1,
    VoiceIDMax = 15
}
```

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

Inherits [IDecoder](#).

### Public Member Functions

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

*Open (initialize) the decoder.*
- void **Dispose** ()
- void **Input** (ref [FrameBuffer](#) buf)
 

*Consumes the given encoded data.*

## Protected Attributes

- OpusDecoder< T > **decoder**

## Properties

- string **Error** [get]

### 3.42.1 Member Function Documentation

#### 3.42.1.1 Input()

```
void Input (
    ref FrameBuffer buf )
```

Consumes the given encoded data.

The callee can call buf.Retain() to prevent the caller from disposing the buffer. In this case, the callee should call buf.Release() when buffer is no longer needed.

Implements [IDecoder](#).

#### 3.42.1.2 Open()

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

Open (initialize) the decoder.

Parameters

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

Implements [IDecoder](#).

## 3.43 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](#) (ref [FrameBuffer](#) byteBuf)  
*Consumes the given encoded data.*
- void [Dispose](#) ()

## Properties

- string [Error](#) [get]

### 3.43.1 Member Function Documentation

#### 3.43.1.1 [Input\(\)](#)

```
void Input (
    ref FrameBuffer buf )
```

Consumes the given encoded data.

The callee can call buf.Retain() to prevent the caller from disposing the buffer. In this case, the callee should call buf.Release() when buffer is no longer needed.

Implements [IDecoder](#).

#### 3.43.1.2 [Open\(\)](#)

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

Open (initialize) the decoder.

#### Parameters

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

Implements [IDecoder](#).

## 3.44 DecoderConfigFrame Class Reference

Stores the config frame and prevents other frames decoding until the decoder is ready.

Inherits [IDisposable](#).

## Public Member Functions

- **DecoderConfigFrame** ([ILogger](#) logger, [IDecoder](#) decoder)
- bool [TryConfigure](#) (ref [FrameBuffer](#) buf, bool decoderReady)  
*Call it in Input().*
- void [Dispose](#) ()

### 3.44.1 Detailed Description

Stores the config frame and prevents other frames decoding until the decoder is ready.

### 3.44.2 Member Function Documentation

#### 3.44.2.1 TryConfigure()

```
bool TryConfigure (
    ref FrameBuffer buf,
    bool decoderReady )
```

Call it in Input().

##### Parameters

<i>buf</i>	Data frame.
<i>decoderReady</i>	True if the decoder is ready.

##### Returns

True if decoder is allowed to decode the current frame.

## 3.45 DeviceEnumerator Class Reference

Inherits [DeviceEnumeratorBase](#).

Inherited by [AudioInEnumerator](#), and [VideoInEnumerator](#).

## Public Member Functions

- **DeviceEnumerator** ([ILogger](#) logger, Windows.Devices.Enumeration.DeviceClass deviceClass)
- override void [Refresh](#) ()
- override void [Dispose](#) ()

## Additional Inherited Members

### 3.46 DeviceEnumeratorBase Class Reference

Inherits [IDeviceEnumerator](#).

Inherited by [DeviceEnumeratorNotSupported](#), [DeviceEnumeratorSingleDevice](#), [AudioInEnumerator](#), [AudioInEnumerator](#), [AudioInEnumerator](#), [VideoInEnumerator](#), [DeviceEnumerator](#), and [AudioInEnumerator](#).

#### Public Member Functions

- **DeviceEnumeratorBase** ([ILogger](#) logger)
- **IEnumerator< DevicelInfo > GetEnumerator ()**
- abstract void **Refresh ()**
- abstract void **Dispose ()**

#### Public Attributes

- virtual bool **IsSupported** => true

#### Protected Attributes

- List< [DevicelInfo](#) > **devices** = new List<[DevicelInfo](#)>()
- [ILogger](#) **logger**

#### Properties

- virtual string **Error** [get, protected set]
- Action **OnReady** [protected get, set]

### 3.47 DeviceFeatures Class Reference

#### Public Member Functions

- **DeviceFeatures** (CameraFacing facing)

#### Properties

- CameraFacing **CameraFacing** [get]

## 3.48 DeviceInfo Struct Reference

### Public Member Functions

- `DeviceInfo` (int id, string name, `DeviceFeatures` features=null)
- `DeviceInfo` (string id, string name, `DeviceFeatures` features=null)
- `DeviceInfo` (string name, `DeviceFeatures` features=null)
- override bool `Equals` (object obj)
- override int `GetHashCode` ()
- override string `ToString` ()

### Static Public Member Functions

- static bool `operator==` (`DeviceInfo` d1, `DeviceInfo` d2)
- static bool `operator!=` (`DeviceInfo` d1, `DeviceInfo` d2)

### Public Attributes

- `DeviceFeatures Features` => features == null ? DeviceFeatures.Default : features
- `DeviceFeatures features`

### Static Public Attributes

- static readonly `DeviceInfo Default` = new `DeviceInfo`(true, -128, "", "[Default]")

### Properties

- bool `IsDefault` [get]
- int `IDInt` [get]
- string `IDString` [get]
- string `Name` [get]

## 3.49 RecorderPreset.DSP Struct Reference

### Public Attributes

- bool `AEC`
- bool `VAD`

## 3.50 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** (VoicelInfo i, ILogger logger)
- abstract void **encodeTyped** (T[ ] buf)

## Protected Attributes

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

## Properties

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

## 3.51 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]

## 3.52 OpusCodec.EncoderFloat Class Reference

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

## Protected Member Functions

- override void **encodeTyped** (float[] buf)

## Additional Inherited Members

### 3.53 OpusCodec.EncoderShort Class Reference

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

## Protected Member Functions

- override void **encodeTyped** (short[] buf)

## Additional Inherited Members

### 3.54 OpusCodec.Factory Class Reference

## Static Public Member Functions

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

### 3.55 Flip Struct Reference

## Public Member Functions

- override bool **Equals** (object obj)
- override int **GetHashCode** ()

## Static Public Member Functions

- static bool **operator==** ([Flip](#) f1, [Flip](#) f2)
- static bool **operator!=** ([Flip](#) f1, [Flip](#) f2)
- static [Flip operator\\*](#) ([Flip](#) f1, [Flip](#) f2)

## Static Public Attributes

- static [Flip None](#)
- static [Flip Vertical](#) = new [Flip\(\)](#) { IsVertical = true }
- static [Flip Horizontal](#) = new [Flip\(\)](#) { IsHorizontal = true }
- static [Flip Both](#) = Vertical \* Horizontal

## Properties

- bool **IsVertical** [get]
- bool **IsHorizontal** [get]

## 3.56 FMODRecorderSetup Class Reference

Inherits [VoiceComponent](#).

### Protected Member Functions

- override void **Awake** ()

### Additional Inherited Members

## 3.57 FrameBuffer Struct Reference

### Public Member Functions

- **FrameBuffer** (byte[] array, int offset, int count, FrameFlags flags, byte frameNum, IDisposable disposer)
- **FrameBuffer** (byte[] array, FrameFlags flags, byte frameNum)
- **FrameBuffer** ([FrameBuffer](#) from, int offset, int count, FrameFlags flags, byte frameNum)
- void **Retain** ()
- void **Release** ()
- override string **ToString** ()

### Public Attributes

- readonly byte[] **array**
- readonly int **offset**
- readonly int **count**
- readonly IDisposable **disposer**
- bool **disposed**
- int **refCnt**
- GCHandle **gcHandle**
- IntPtr **ptr**
- bool **pinned**
- bool **IsFEC** => (Flags & FrameFlags.FEC) != 0
- bool **IsConfig** => (Flags & FrameFlags.Config) != 0
- bool **IsKeyframe** => (Flags & FrameFlags.KeyFrame) != 0

## Properties

- IntPtr **Ptr** [get]
- byte[] **Array** [get]
- int **Length** [get]
- int **Offset** [get]
- FrameFlags **Flags** [get]
- byte **FrameNum** [get]

## 3.58 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]

## 3.59 Framer< T > Class Template Reference

Utility class to re-frame packets.

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

### Public Member Functions

- [Framer](#) (int frameSize)  
*Create new Framer instance.*
- virtual I`Enumerable< T[]>` [Frame](#) (T[] buf)  
*Append arbitrary-sized buffer and return available full frames.*

### Protected Attributes

- T[] **frame**
- int **sizeofT**
- int **framePos** = 0

### 3.59.1 Detailed Description

Utility class to re-frame packets.

### 3.59.2 Constructor & Destructor Documentation

#### 3.59.2.1 Framer()

```
Framer (
    int frameSize )
```

Create new Framer instance.

### 3.59.3 Member Function Documentation

#### 3.59.3.1 Frame()

```
virtual IEnumerable<T[]> Frame (
    T[] buf ) [virtual]
```

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).

Reimplemented in [FramerResampler< T >](#).

## 3.60 FramerResampler< T > Class Template Reference

Inherits [Framer< T >](#).

### Public Member Functions

- [FramerResampler](#) (int frameSize, int channels, int resampleNum, int resampleDen, bool interpolate)
- override [IEnumerable< T\[\]> Frame](#) (T[] bufT)  
*Append arbitrary-sized buffer and return available full frames.*

### Protected Attributes

- bool **TisFloat**
- bool **interpolate**
- int **channels**
- int **resampleNum**
- int **resampleDen**
- float **resampleRatioInv**
- int **delta**

### 3.60.1 Member Function Documentation

#### 3.60.1.1 Frame()

```
override IEnumerable<T[]> Frame (
    T[] buf ) [virtual]
```

Append arbitrary-sized buffer and return available full frames.

**Parameters**

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

**Returns**

Enumerator of full frames (might be none).

Reimplemented from [Framer< T >](#).

## 3.61 FusionVoiceClient Class Reference

Inherits [VoiceFollowClient](#), and [INetworkRunnerCallbacks](#).

### Public Attributes

- bool [UseFusionAppSettings](#) = true  
*Whether or not to use the `Voice` AppId and all the other AppSettings from `Fusion`'s RealtimeAppSettings Scriptable Object singleton in the `Voice` client/app.*
- bool [UseFusionAuthValues](#) = true  
*Whether or not to use the same AuthenticationValues used in `Fusion` client/app in `Voice` client/app as well. This means that the same UserID will be used in both clients. If custom authentication is used and setup in `Fusion` AppId from dashboard, the same configuration should be done for the `Voice` AppId.*

### Protected Member Functions

- override void [Start](#) ()
- override void [Awake](#) ()
- override void [OnDestroy](#) ()
- override [Speaker](#) [InstantiateSpeakerForRemoteVoice](#) (int playerId, byte voiceId, object userData)
- override string [GetVoiceRoomName](#) ()
- override bool [ConnectVoice](#) ()

### Protected Attributes

- override bool [LeaderInRoom](#) => this.networkRunner.SessionInfo.IsValid
- override bool [LeaderOfflineMode](#) => networkRunner.GameMode == GameMode.Single

### Additional Inherited Members

#### 3.61.1 Member Data Documentation

### 3.61.1.1 UseFusionAppSettings

```
bool UseFusionAppSettings = true
```

Whether or not to use the [Voice](#) AppId and all the other AppSettings from [Fusion](#)'s RealtimeAppSettings Scriptable Object singleton in the [Voice](#) client/app.

### 3.61.1.2 UseFusionAuthValues

```
bool UseFusionAuthValues = true
```

Whether or not to use the same AuthenticationValues used in [Fusion](#) client/app in [Voice](#) client/app as well. This means that the same UserID will be used in both clients. If custom authentication is used and setup in [Fusion](#) AppId from dashboard, the same configuration should be done for the [Voice](#) AppId.

## 3.62 AudioUtil.GeneratorPusher< T > Class Template Reference

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

Inherits [IAudioPusher< T >](#).

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

### Public Member Functions

- **GeneratorPusher** (int bufSizeMs=100, int samplingRate=48000, int channels=1)
- void **SetCallback** (Action< T[]> callback, [ObjectFactory](#)< T[]>, int > bufferFactory, int optimalFrameSize)  
*Set the callback function used for pushing data*
- void **Dispose** ()

### Protected Member Functions

- abstract int **Gen** (T[] buf, long timeSamples)

### Protected Attributes

- long **timeSamples**

### Properties

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

### 3.62.1 Detailed Description

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

### 3.62.2 Member Function Documentation

#### 3.62.2.1 SetCallback()

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

Set the callback function used for pushing data

#### Parameters

<i>callback</i>	Callback function to use
<i>bufferFactory</i>	Buffer factory used to create the buffer that is pushed to the callback

Implements [IAudioPusher< T >](#).

## 3.63 AudioUtil.GeneratorReader< T > Class Template Reference

Inherits [IAudioReader< T >](#).

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

### Public Member Functions

- **GeneratorReader** (*Func< double > clockSec=null, int samplingRate=48000, int channels=1*)
- void **Dispose** ()
- bool **Read** (*T[] buf*)

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

### Protected Member Functions

- abstract int **Gen** (*T[] buf, long timeSamples*)

### Properties

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

### 3.63.1 Member Function Documentation

#### 3.63.1.1 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 >](#).

## 3.64 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.*

#### 3.64.1 Detailed Description

Audio Source interface.

#### 3.64.2 Property Documentation

### 3.64.2.1 Channels

```
int Channels [get]
```

Number of channels in the audio signal.

### 3.64.2.2 Error

```
string Error [get]
```

If not null, audio object is in invalid state.

### 3.64.2.3 SamplingRate

```
int SamplingRate [get]
```

Sampling rate of the audio signal (in Hz).

## 3.65 IAudioInChangeNotifier Interface Reference

Inherits IDisposable.

Inherited by [AudioInChangeNotifierNotSupportedException](#), [AudioInChangeNotifier](#), and [AudioInChangeNotifier](#).

### Properties

- bool **IsSupported** [get]
- string **Error** [get]

## 3.66 IAudioOut< T > Interface Template Reference

Inherited by [AudioOutDelayControl< T >](#), and [AudioOutDummy< 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]

## 3.67 IAudioPusher< T > Interface Template Reference

Audio Pusher interface.

Inherits [IAudioDesc](#).

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

### Public Member Functions

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

### Additional Inherited Members

#### 3.67.1 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,

#### 3.67.2 Member Function Documentation

##### 3.67.2.1 SetCallback()

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

Set the callback function used for pushing data.

#### Parameters

<i>callback</i>	Callback function to use.
<i>bufferFactory</i>	Buffer factory used to create the buffer that is pushed to the callback

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

## 3.68 IAudioReader< T > Interface Template Reference

Audio Reader interface.

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

Inherited by [AudioUtil.GeneratorReader< T >](#), and [AudioInReader< T >](#).

### Additional Inherited Members

#### 3.68.1 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).

## 3.69 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.*

#### 3.69.1 Detailed Description

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

#### 3.69.2 Member Function Documentation

##### 3.69.2.1 Read()

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

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

**Parameters**

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

**Returns**

True if buffer was filled successfully, false otherwise.

Implemented in [AudioInReader< T >](#), and [AudioUtil.GeneratorReader< T >](#).

## 3.70 IDecoder Interface Reference

Generic decoder interface.

Inherits [IDisposable](#).

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

### Public Member Functions

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

### Properties

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

#### 3.70.1 Detailed Description

Generic decoder interface.

#### 3.70.2 Member Function Documentation

##### 3.70.2.1 Input()

```
void Input (
    ref FrameBuffer buf )
```

Consumes the given encoded data.

The callee can call buf.Retain() to prevent the caller from disposing the buffer. In this case, the callee should call buf.Release() when buffer is no longer needed.

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

##### 3.70.2.2 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 >](#), [ByteStreamDecoder](#), and [OpusCodec.Decoder< T >](#).

### 3.70.3 Property Documentation

#### 3.70.3.1 Error

`string Error [get]`

If not null, the object is in invalid state.

## 3.71 IDecoderDirect< B > Interface Template Reference

Interface for an decoder which outputs data via explicit call.

Inherits [IDecoder](#).

### Properties

- `Action< B > Output [set]`  
*Callback to call when a new decoded data buffer is available.*

### Additional Inherited Members

#### 3.71.1 Detailed Description

Interface for an decoder which outputs data via explicit call.

#### 3.71.2 Property Documentation

##### 3.71.2.1 Output

`Action<B> Output [set]`

Callback to call when a new decoded data buffer is available.

## 3.72 IDeviceEnumerator Interface Reference

Inherits IDisposable, and IEnumerable< DeviceInfo >.

Inherited by [DeviceEnumeratorBase](#).

### Public Member Functions

- void **Refresh** ()

### Properties

- bool **IsSupported** [get]
- Action **OnReady** [set]
- string **Error** [get]

## 3.73 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 >** ()  
*Returns an platform-specific interface.*

### 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).*

### 3.73.1 Detailed Description

Generic encoder interface.

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

## 3.73.2 Member Function Documentation

### 3.73.2.1 DequeueOutput()

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

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

### 3.73.2.2 EndOfStream()

```
void EndOfStream ( )
```

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

### 3.73.2.3 GetPlatformAPI< I >()

```
I GetPlatformAPI< I > ( )
```

Returns an platform-specific interface.

Type Constraints

*I*: class

## 3.73.3 Property Documentation

### 3.73.3.1 Error

```
string Error [get]
```

If not null, the object is in invalid state.

### 3.73.3.2 Output

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

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

## 3.74 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

#### 3.74.1 Detailed Description

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

#### 3.74.2 Member Function Documentation

##### 3.74.2.1 [Input\(\)](#)

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

Consumes the given raw data.

###### Parameters

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

## 3.75 IEncoderDirectImage Interface Reference

Interface for an encoder which consumes images via explicit call.

Inherits [IEncoderDirect< ImageBufferNative >](#).

### Properties

- ImageFormat [ImageFormat](#) [get]  
*Recommended encoder input image format. Encoder may support other formats.*

## Additional Inherited Members

### 3.75.1 Detailed Description

Interface for an encoder which consumes images via explicit call.

### 3.75.2 Property Documentation

#### 3.75.2.1 ImageFormat

```
ImageFormat ImageFormat [get]
```

Recommended encoder input image format. Encoder may support other formats.

## 3.76 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.*

### 3.76.1 Detailed Description

Audio Level Metering interface.

### 3.76.2 Member Function Documentation

### 3.76.2.1 ResetAccumAvgPeakAmp()

```
void ResetAccumAvgPeakAmp ( )
```

Reset [AccumAvgPeakAmp](#).

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

## 3.76.3 Property Documentation

### 3.76.3.1 AccumAvgPeakAmp

```
float AccumAvgPeakAmp [get]
```

Average of CurrentPeakAmps since last reset.

### 3.76.3.2 CurrentAvgAmp

```
float CurrentAvgAmp [get]
```

Average amplitude value over last half second.

### 3.76.3.3 CurrentPeakAmp

```
float CurrentPeakAmp [get]
```

Maximum amplitude value over last half second sec.

## 3.77 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.*

### 3.77.1 Detailed Description

Interface for an outgoing audio stream.

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

### 3.77.2 Member Function Documentation

#### 3.77.2.1 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

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

Implemented in `LocalVoiceAudioDummy`, and `LocalVoiceAudio< T >`.

### 3.77.3 Property Documentation

#### 3.77.3.1 LevelMeter

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

The `LevelMeter` utility in use.

### 3.77.3.2 VoiceDetector

`AudioUtil.IVoiceDetector` `VoiceDetector` [get]

The VoiceDetector in use.

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

### 3.77.3.3 VoiceDetectorCalibrating

`bool` `VoiceDetectorCalibrating` [get]

If true, voice detector calibration is in progress.

## 3.78 ILogger Interface Reference

Inherited by [Logger](#).

### Public Member Functions

- `void Log (LogLevel level, string fmt, params object[] args)`

### Properties

- `LogLevel Level` [get]

## 3.79 ImageBufferInfo Struct Reference

### Classes

- struct [StrideSet](#)

### Public Member Functions

- `ImageBufferInfo (int width, int height, StrideSet stride, ImageFormat format)`

### Properties

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

## 3.80 ImageBufferNative Class Reference

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

### Classes

- struct [PlaneSet](#)

### Public Member Functions

- [ImageBufferNative](#) ([ImageBufferInfo](#) info)
- [ImageBufferNative](#) (IntPtr buf, int width, int height, int stride, [ImageFormat](#) imageFormat)
- void [Retain](#) ()
- void [Retain](#) (int times)
- void [Release](#) ()
- virtual void [Dispose](#) ()

### Public Attributes

- [ImageBufferInfo](#) [Info](#)
- [PlaneSet](#) [Planes](#)

### Protected Member Functions

- virtual void [Reset](#) ()
- virtual void [Free](#) ()

## 3.81 ImageBufferNativeAlloc Class Reference

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

### Public Member Functions

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

### Protected Member Functions

- override void [Free](#) ()

### Additional Inherited Members

## 3.82 ImageBufferNativeGCHandleBytes Class Reference

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

## Public Member Functions

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

## Public Attributes

- byte[ ][ ] **PlaneBytes** => planeBytes

## Protected Member Functions

- override void **Free** ()

## 3.83 ImageBufferNativeGCHandleSinglePlane Class Reference

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

## Public Member Functions

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

## Protected Member Functions

- override void **Free** ()

## Additional Inherited Members

## 3.84 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, [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

### 3.85 IP<sub>T</sub> Processor< T > Interface Template Reference

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 data.*

##### 3.85.1 Detailed Description

Processor interface.

##### 3.85.2 Member Function Documentation

###### 3.85.2.1 Process()

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

Process a frame of data.

###### Parameters

<code>buf</code>	Buffer containing input data
------------------	------------------------------

###### Returns

Buffer containing output 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 >](#).

### 3.86 IResettable Interface Reference

Inherited by [AudioInPusher](#), [AudioInReader](#), and [AndroidAudioInAEC](#).

## Public Member Functions

- void **Reset** ()

## 3.87 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.*

### 3.87.1 Detailed Description

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

### 3.87.2 Member Function Documentation

#### 3.87.2.1 Service()

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

Service function that should be called regularly.

Implemented in `BufferReaderPushAdapterAsyncPool< T >`, and `BufferReaderPushAdapterBase< T >`.

## 3.88 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.*

### 3.88.1 Detailed Description

[Voice](#) Activity Detector interface.

### 3.88.2 Property Documentation

#### 3.88.2.1 ActivityDelayMs

int [ActivityDelayMs](#) [get], [set]

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

#### 3.88.2.2 Detected

bool [Detected](#) [get]

If true, voice detected.

#### 3.88.2.3 DetectedTime

DateTime [DetectedTime](#) [get]

Last time when switched to detected state.

### 3.88.2.4 On

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

If true, voice detection enabled.

### 3.88.2.5 Threshold

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

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

## 3.88.3 Event Documentation

### 3.88.3.1 OnDetected

```
Action OnDetected
```

Called when switched to detected state.

## 3.89 IVoiceTransport Interface Reference

Inherited by [LoadBalancingTransport](#).

### Public Member Functions

- bool [IsChannelJoined](#) (int channelId)
- void [SendVoiceInfo](#) ([LocalVoice](#) voice, int channelId, bool targetMe, int[] targetPlayers)
- void [SendVoiceRemove](#) ([LocalVoice](#) voice, int channelId, bool targetMe, int[] targetPlayers)
- void [SendFrame](#) (ArraySegment< byte > data, FrameFlags flags, ushort evNumber, byte frNumber, byte voiceld, int channelId, [SendFrameParams](#) par)
- string [ChannelIdStr](#) (int channelId)
- string [PlayerIdStr](#) (int playerId)
- int [GetPayloadFragmentSize](#) ([SendFrameParams](#) par)

## 3.90 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 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]

### 3.90.1 Detailed Description

Audio Level Meter.

### 3.90.2 Member Function Documentation

#### 3.90.2.1 Process()

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

Process a frame of data.

##### Parameters

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

**Returns**

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

Implements [IProcessor< T >](#).

### 3.90.2.2 ResetAccumAvgPeakAmp()

```
void ResetAccumAvgPeakAmp ( )
```

Reset AccumAvgPeakAmp.

Implements [AudioUtil.ILevelMeter](#).

## 3.91 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]

### 3.91.1 Detailed Description

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

### 3.91.2 Member Function Documentation

#### 3.91.2.1 ResetAccumAvgPeakAmp()

```
void ResetAccumAvgPeakAmp ( )
```

Reset AccumAvgPeakAmp.

Implements [AudioUtil.ILevelMeter](#).

## 3.92 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

#### 3.92.1 Detailed Description

[LevelMeter](#) specialization for float audio.

#### 3.92.2 Constructor & Destructor Documentation

##### 3.92.2.1 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.

## 3.93 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

### 3.93.1 Detailed Description

[LevelMeter](#) specialization for short audio.

### 3.93.2 Constructor & Destructor Documentation

#### 3.93.2.1 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.

## 3.94 LoadBalancingTransport Class Reference

Extends LoadBalancingClient with media streaming functionality.

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

Inherited by [LoadBalancingTransport2](#).

### Public Member Functions

- virtual int [GetPayloadFragmentSize](#) ([SendFrameParams](#) par)
- bool [IsChannelJoined](#) (int channelId)
- [LoadBalancingTransport](#) ([ILogger](#) logger=null, [ConnectionProtocol](#) connectionProtocol=[ConnectionProtocol.Udp](#), bool cppCompatibilityMode=false)
 

*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 [SendVoiceInfo](#) ([LocalVoice](#) voice, int channelId, bool targetMe, int[] targetPlayers)
- void [SendVoiceRemove](#) ([LocalVoice](#) voice, int channelId, bool targetMe, int[] targetPlayers)
- void [SendFrame](#) ([ArraySegment< byte >](#) data, [FrameFlags](#) flags, ushort evNumber, byte frNumber, byte voiceId, int channelId, [SendFrameParams](#) par)
- string [ChannelIdStr](#) (int channelId)
- string [PlayerIdStr](#) (int playerId)
- void [Dispose](#) ()
 

*Releases all resources used by the LoadBalancingTransport instance.*

## Protected Member Functions

- virtual object **buildFrameMessage** (byte voiceId, ushort evNumber, byte frNumber, ArraySegment< byte > data, FrameFlags flags)
- virtual void **onEventActionVoiceClient** (EventData ev)

## Protected Attributes

- [VoiceClient](#) **voiceClient**
- readonly bool **cppCompatibilityMode**
- readonly [ILogger](#) **logger**
- virtual byte **FrameCode** => [VoiceEvent](#).Code

## Properties

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

### 3.94.1 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.

### 3.94.2 Constructor & Destructor Documentation

#### 3.94.2.1 LoadBalancingTransport()

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

Initializes a new [LoadBalancingTransport](#).

#### Parameters

<i>logger</i>	<a href="#">ILogger</a> instance. If null, this instance LoadBalancingClient.DebugReturn implementation is used. <a href="#">ConnectionProtocol</a>
<i>connectionProtocol</i>	Connection protocol (UDP or TCP). <a href="#">ConnectionProtocol</a>
<i>cppCompatibilityMode</i>	Use a protocol compatible with <a href="#">Voice</a> C++ API.

### 3.94.3 Member Function Documentation

#### 3.94.3.1 Dispose()

```
void Dispose ( )
```

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

#### 3.94.3.2 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).

### 3.94.4 Property Documentation

#### 3.94.4.1 VoiceClient

```
VoiceClient VoiceClient [get]
```

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

## 3.95 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](#), bool cppCompatibilityMode=false)
- override int **GetPayloadFragmentSize** ([SendFrameParams](#) par)

### Protected Member Functions

- override object **buildFrameMessage** (byte voiceld, ushort evNumber, byte frNumber, [ArraySegment< byte >](#) data, [FrameFlags](#) flags)
- override void **onEventActionVoiceClient** ([EventData](#) ev)

## Protected Attributes

- override byte **FrameCode** => VoiceEvent.FrameCode

## Additional Inherited Members

### 3.95.1 Detailed Description

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

## 3.96 LocalVoice Class Reference

Represents outgoing data stream.

Inherits [IDisposable](#).

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

## Public Member Functions

- void **SendSpacingProfileStart** ()
- void **RemoveSelf** ()  
*Remove this voice from its VoiceClient (using [VoiceClient.RemoveLocalVoice](#))*
- virtual void **Dispose** ()

## Public Attributes

- int **EventBufferSize** => EV\_BUF\_SIZE

## Static Public Attributes

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

## Protected Member Functions

- bool **targetExists** (bool targetMe, int[ ] targetPlayers)
- void **sendVoiceInfoAndConfigFrame** (bool targetMe, int[ ] targetPlayers)
- void **sendVoiceRemove** (bool targetMe, int[ ] targetPlayers)

## Protected Attributes

- int[ ] **targetPlayers\_**
- [VoiceInfo](#) **info**
- [IEncoder](#) **encoder**
- [VoiceClient](#) **voiceClient**
- bool **threadingEnabled**
- [ArraySegment< byte >](#) **configFrame**
- volatile bool **disposed**
- object **disposeLock** = new object()
- bool **isJoined** => voiceClient != null && voiceClient.transport.IsChannelJoined(this.channelId)

## Properties

- **VoiceInfo Info** [get]
 

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

*If true, stream data broadcasted.*
- **bool IsCurrentlyTransmitting** [get]
 

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

*Sent frames counter.*
- **int FramesSentFragmented** [get]
 

*Sent fragmented frames counter.*
- **int FramesSentFragments** [get]
 

*Sent frames fragments counter.*
- **int FramesSentBytes** [get]
 

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

*Send data reliable. See also [VoiceCreateOptions.Reliable](#).*
- **bool Encrypt** [get, set]
 

*Send data encrypted. See also [VoiceCreateOptions.Encrypt](#).*
- **bool Fragment** [get, set]
 

*Split frames into fragments according to the size provided by the Transport. See also [VoiceCreateOptions.Fragment](#).*
- **int FEC** [get, set]
 

*Forward Error Correction control. See also [VoiceCreateOptions.FEC](#).*
- **IServiceable LocalUserServiceable** [get, set]
 

*Optional user object attached to [LocalVoice](#). its Service() will be called at each [VoiceClient.Service\(\)](#) call.*
- **byte Group** [get, set]
 

*If InterestGroup != 0, streaming only to the players subscribed to this group (if supported by the transport). See also [VoiceCreateOptions.InterestGroup](#).*
- **byte InterestGroup** [get, set]
 

*If InterestGroup != 0, streaming only to the players subscribed to this group (if supported by the transport). See also [VoiceCreateOptions.InterestGroup](#).*
- **bool DebugEchoMode** [get, set]
 

*If true, outgoing stream routed back to client via server same way as for remote client's streams. See also [VoiceCreateOptions.DebugEchoMode](#).*
- **int[]? TargetPlayers** [get, set]
 

*If TargetPlayers is not null, sending voice info and streaming only to clients having player numbers specified in the array (if supported by transport). See also [VoiceCreateOptions.TargetPlayers](#).*
- **string SendSpacingProfileDump** [get]
 

*Logs input frames time spacing profiling results. Do not call frequently.*
- **byte ID** [get]
- **string shortName** [get]
- **string Name** [get]
- **string LogPrefix** [get]

### 3.96.1 Detailed Description

Represents outgoing data stream.

### 3.96.2 Member Function Documentation

### 3.96.2.1 RemoveSelf()

```
void RemoveSelf( )
```

Remove this voice from its [VoiceClient](#) (using [VoiceClient.RemoveLocalVoice](#))

## 3.96.3 Property Documentation

### 3.96.3.1 DebugEchoMode

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

If true, outgoing stream routed back to client via server same way as for remote client's streams. See also [VoiceCreateOptions.DebugEchoMode](#).

This functionality availability depends on transport.

### 3.96.3.2 Encrypt

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

Send data encrypted. See also [VoiceCreateOptions.Encrypt](#).

### 3.96.3.3 FEC

```
int FEC [get], [set]
```

Forward Error Correction control. See also [VoiceCreateOptions.FEC](#).

### 3.96.3.4 Fragment

```
bool Fragment [get], [set]
```

Split frames into fragments according to the size provided by the Transport. See also [VoiceCreateOptions.Fragment](#).

### 3.96.3.5 FramesSent

```
int FramesSent [get]
```

Sent frames counter.

### 3.96.3.6 FramesSentBytes

```
int FramesSentBytes [get]
```

Sent frames bytes counter.

### 3.96.3.7 FramesSentFragmented

```
int FramesSentFragmented [get]
```

Sent fragmented frames counter.

### 3.96.3.8 FramesSentFragments

```
int FramesSentFragments [get]
```

Sent frames fragments counter.

### 3.96.3.9 Info

```
VoiceInfo Info [get]
```

Returns Info structure assigned on local voice creation.

### 3.96.3.10 InterestGroup

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

If InterestGroup != 0, streaming only to the players subscribed to this group (if supported by the transport). See also [VoiceCreateOptions.InterestGroup](#).

A remote voice is created even if the remote player is not subscribed to the InterestGroup. Use [VoiceCreateOptions.TargetPlayers](#) and [TargetPlayers](#) to completely hide the existence of [LocalVoice](#) from the remote player./>

### 3.96.3.11 IsCurrentlyTransmitting

```
bool IsCurrentlyTransmitting [get]
```

Returns true if stream broadcasts.

### 3.96.3.12 LocalUserServiceable

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

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

### 3.96.3.13 Reliable

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

Send data reliable. See also [VoiceCreateOptions.Reliable](#).

### 3.96.3.14 SendSpacingProfileMax

```
int SendSpacingProfileMax [get]
```

Logs input frames time spacing profiling results. Do not call frequently.

### 3.96.3.15 TargetPlayers

```
int []?? TargetPlayers [get], [set]
```

If TargetPlayers is not null, sending voice info and streaming only to clients having player numbers specified in the array (if supported by transport). See also [VoiceCreateOptions.TargetPlayers](#).

TargetPlayers update triggers the sending of voice info to added players and voice remove to removed. Depending on the transport, TargetPlayers may disregard [InterestGroup](#): the remote player whos number is in TargetPlayers, receives the stream even if not subscribed to the interest group. If the local player number is in TargetPlayers, it works like [DebugEchoMode](#). Using both DebugEchoMode and TargetPlayers at the same time to route the stream back to the sender leads to an inconsistent state after one of the options is switched off: the voice is removed but the frames are still delivered.

### 3.96.3.16 TransmitEnabled

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

If true, stream data broadcasted.

## 3.97 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.*

## Protected Member Functions

- void **initBuiltInProcessors** ()

## Protected Attributes

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

## Properties

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

*True if the VoiceDetector is currently calibrating.*

## Additional Inherited Members

### 3.97.1 Detailed Description

Outgoing audio stream.

### 3.97.2 Member Function Documentation

#### 3.97.2.1 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 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](#).

### 3.97.3 Property Documentation

#### 3.97.3.1 VoiceDetectorCalibrating

bool VoiceDetectorCalibrating [get]

True if the VoiceDetector is currently calibrating.

## 3.98 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

#### 3.98.1 Detailed Description

Dummy [LocalVoiceAudio](#)

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

#### 3.98.2 Member Function Documentation

##### 3.98.2.1 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](#).

### 3.98.3 Member Data Documentation

#### 3.98.3.1 Dummy

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

A Dummy [LocalVoiceAudio](#) instance.

## 3.99 LocalVoiceAudioFloat Class Reference

Specialization of LocalVoiceAudio<T> for float audio

Inherits [LocalVoiceAudio< float >](#).

### Additional Inherited Members

#### 3.99.1 Detailed Description

Specialization of LocalVoiceAudio<T> for float audio

## 3.100 LocalVoiceAudioShort Class Reference

Specialization of LocalVoiceAudio<T> for short audio

Inherits [LocalVoiceAudio< short >](#).

### Additional Inherited Members

#### 3.100.1 Detailed Description

Specialization of LocalVoiceAudio<T> for short audio

## 3.101 LocalVoiceFramed< T > Class Template Reference

Typed re-framing [LocalVoice](#)

Inherits [LocalVoice](#).

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 [RemoveProcessor](#) (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 LocalVoiceFramed< T > instance. Buffers used for asynchronous push will be disposed in encoder thread's 'finally'.*

### Protected Member Functions

- [T\[\] processFrame](#) ([T\[\]](#) buf, int p0, int p1)

### Properties

- int [OptimalSourceFrameSize](#) [get]
- [ObjectFactory< T\[\], int > BufferFactory](#) [get]
 

*<see cref="PushData(T[])" and />.*
- bool [PushDataAsyncReady](#) [get]

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

### Additional Inherited Members

#### 3.101.1 Detailed Description

Typed re-framing [LocalVoice](#)

Consumes data in array buffers of arbitrary length. Repacks them in frames of [VoiceInfo.FrameSize](#) length for further processing and encoding.

### 3.101.2 Member Function Documentation

#### 3.101.2.1 AddPostProcessor()

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

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

**Parameters**

<i>processors</i>	<input type="button" value=""/>
-------------------	---------------------------------

**3.101.2.2 AddPreProcessor()**

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

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

**Parameters**

<i>processors</i>	<input type="button" value=""/>
-------------------	---------------------------------

**3.101.2.3 ClearProcessors()**

```
void ClearProcessors ( )
```

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

**3.101.2.4 Dispose()**

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

Releases resources used by the LocalVoiceFramed<T> instance. Buffers used for asynchronous push will be disposed in encoder thread's 'finally'.

Reimplemented from [LocalVoice](#).

**3.101.2.5 PushData()**

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

Synchronously push data into this stream.

### 3.101.2.6 PushDataAsync()

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

Asynchronously push data into this stream.

### 3.101.2.7 RemoveProcessor()

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

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

#### Parameters

<i>processors</i>	<input type="button" value=""/>
-------------------	---------------------------------

## 3.101.3 Property Documentation

### 3.101.3.1 BufferFactory

`ObjectFactory<T[ ], int> BufferFactory [get]`

<see cref="PushData(T[])" and />.

### 3.101.3.2 PushDataAsyncReady

`bool PushDataAsyncReady [get]`

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

## 3.102 Logger Class Reference

Inherits [ILogger](#).

### Public Member Functions

- **Logger** (LogLevel level=LogLevel.Debug)
- void **Log** (LogLevel level, string fmt, params object[ ] args)

## Properties

- LogLevel **Level** [get, set]

## 3.103 MicAmplifier Class Reference

Inherits [VoiceComponent](#).

## Properties

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

## Additional Inherited Members

## 3.104 MicAmplifierFloat Class Reference

Inherits [IProcessor< float >](#).

## Public Member Functions

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

## Properties

- float **AmplificationFactor** [get, set]
- bool **Disabled** [get, set]

## 3.105 MicAmplifierShort Class Reference

Inherits [IProcessor< short >](#).

## Public Member Functions

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

## Properties

- float **AmplificationFactor** [get, set]
- bool **Disabled** [get, set]

## 3.106 MicrophonePermission Class Reference

Helper to request Microphone permission on Android or iOS.

Inherits [VoiceComponent](#).

### Public Member Functions

- void **InitVoice** ()

### Protected Member Functions

- override void **Awake** ()

### Properties

- bool? **HasPermission** [get]

### Events

- static Action< bool > **MicrophonePermissionCallback**

### Additional Inherited Members

#### 3.106.1 Detailed Description

Helper to request Microphone permission on Android or iOS.

## 3.107 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]

## 3.108 MicWrapperPusher Class Reference

Inherits [IAudioPusher< float >](#).

### Public Member Functions

- **MicWrapperPusher** (GameObject parent, string device, int suggestedFrequency, [ILogger](#) logger)
- void **SetCallback** (Action< float[]> callback, [ObjectFactory< float\[\]>](#), int > bufferFactory, int optimalFrame←Size)
- void **Dispose** ()

### Properties

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

## 3.109 MicWrapperPusherOnAudioFilterRead Class Reference

Inherits MonoBehaviour.

### Events

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

## 3.110 MonoPInvokeCallbackAttribute Class Reference

Inherits Attribute.

### Public Member Functions

- **MonoPInvokeCallbackAttribute** (Type t)

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

Object factory with optional info useful in combination with ObjectPool< TType, TInfo >.

Inherits IDisposable.

Inherited by [ObjectPool< TType, TInfo >](#).

### Public Member Functions

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

### 3.111.1 Detailed Description

Object factory with optional info useful in combination with ObjectPool< TType, TInfo >.

### Template Parameters

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

## 3.112 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 [ObjectFactory< TType, TInfo >](#).

### Public Member Functions

- [ObjectPool](#) (int capacity, string name, *TInfo* info)
 

*Create a new ObjectPool instance with the given info structure.*
- *TType* [New](#) ()
 

*Acquire an existing object, or create a new one if none are available.*
- *TType* [New](#) (*TInfo* info)
 

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

*Returns object to pool.*
- virtual bool [Free](#) (*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.*

#### 3.112.1 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).

**3.112.2 Constructor & Destructor Documentation****3.112.2.1 ObjectPool()**

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

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

## 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.

**3.112.3 Member Function Documentation****3.112.3.1 Dispose()**

```
void Dispose ( )
```

Free resources associated with this [ObjectPool](#)

**3.112.3.2 Free() [1/2]**

```
virtual bool Free (
    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.
------------	-----------------------------------

Implements [ObjectFactory< TType, TInfo >](#).

**3.112.3.3 Free() [2/2]**

```
virtual bool Free (
    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.

Implements [ObjectFactory< TType, TInfo >](#).

**3.112.3.4 New() [1/2]**

```
TType New ( )
```

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.

Implements [ObjectFactory< TType, TInfo >](#).

**3.112.3.5 New() [2/2]**

```
TType New (
    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.
-------------	---

Implements [ObjectFactory< TType, TInfo >](#).

### 3.112.4 Property Documentation

#### 3.112.4.1 Info

`TInfo Info [get]`

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

## 3.113 OpusCodec Class Reference

### Classes

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

### Public Types

- enum [FrameDuration](#)

### Properties

- static string [Version](#) [get]

## 3.114 PhotonAppSettings Class Reference

Collection of connection-relevant settings, used internally by `PhotonNetwork.ConnectUsingSettings`.

Inherits `ScriptableObject`.

### Public Member Functions

- void [UseCloud](#) (string cloudAppid, string code="")  
*Sets appid and region code in the AppSettings. Used in Editor.*
- override string [ToString](#) ()  
*String summary of the AppSettings.*

## Static Public Member Functions

- static void **LoadOrCreateSettings ()**

## Public Attributes

- AppSettings **AppSettings**

## Properties

- static [PhotonAppSettings](#) **Instance** [get]

### 3.114.1 Detailed Description

Collection of connection-relevant settings, used internally by PhotonNetwork.ConnectUsingSettings.

Includes the AppSettings class from the Realtime APIs plus some other, PUN-relevant, settings.

### 3.114.2 Member Function Documentation

#### 3.114.2.1 ToString()

```
override string ToString ( )
```

String summary of the AppSettings.

#### 3.114.2.2 UseCloud()

```
void UseCloud (
    string cloudAppid,
    string code = "" )
```

Sets appid and region code in the AppSettings. Used in Editor.

## 3.115 PhotonVoiceCreatedParams Class Reference

## Properties

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

## 3.116 PhotonVoiceLagSimulationGui Class Reference

Inherits MonoBehaviour.

### Public Member Functions

- void **Start ()**

## 3.117 PhotonVoiceStatsGui Class Reference

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

Inherits MonoBehaviour.

### 3.117.1 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).

## 3.118 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](#).

### Protected Member Functions

- override void **Awake ()**

### Properties

- **Recorder RecorderInUse [get]**  
*The Recorder component currently used by this PhotonVoiceView*
- **Speaker SpeakerInUse [get]**  
*The Speaker component currently used by this PhotonVoiceView*
- **bool IsSpeaking [get]**  
*If true, this PhotonVoiceView has a Speaker that is currently playing received audio frames from remote audio source*
- **bool IsRecording [get]**  
*If true, this PhotonVoiceView has a Recorder that is currently transmitting audio stream from local audio source*

## Additional Inherited Members

### 3.118.1 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.

### 3.118.2 Property Documentation

#### 3.118.2.1 IsRecording

```
bool IsRecording [get]
```

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

#### 3.118.2.2 IsSpeaking

```
bool IsSpeaking [get]
```

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

#### 3.118.2.3 RecorderInUse

```
Recorder RecorderInUse [get]
```

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

#### 3.118.2.4 SpeakerInUse

```
Speaker SpeakerInUse [get]
```

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

## 3.119 ImageBufferNative.PlaneSet Struct Reference

### Public Member Functions

- **PlaneSet** (int length, IntPtr p0=default(IntPtr), IntPtr p1=default(IntPtr), IntPtr p2=default(IntPtr), IntPtr p3=default(IntPtr))

### Properties

- IntPtr **this[int key]** [get, set]
- int **Length** [get]

## 3.120 Platform Class Reference

### Static Public Member Functions

- static IDeviceEnumerator **CreateAudioInEnumerator** (ILogger logger)
- static IAudioInChangeNotifier **CreateAudioInChangeNotifier** (Action callback, ILogger logger)
- static IEncoder **CreateDefaultAudioEncoder**< T > (ILogger logger, VoicelInfo info)
- static IAudioDesc **CreateDefault AudioSource** (ILogger logger, DeviceInfo dev, int samplingRate, int channels, object otherParams=null)
- static IAudioOut< float > **CreateUnity AudioOut** (UnityEngine.AudioSource audioSource, AudioOutDelayControl.PlayDelayConfig playDelayConfig, ILogger logger, string logPrefix, bool debugInfo)
- static IDeviceEnumerator **CreateVideoInEnumerator** (ILogger logger)
- static IEncoderDirectImage **CreateDefaultVideoEncoder** (ILogger logger, VoicelInfo info)
- static IDecoderDirect< ImageBufferNative > **CreateDefaultVideoDecoder** (ILogger logger, VoicelInfo info)
- static IVideoRecorder **CreateDefaultVideoRecorder** (ILogger logger, VoicelInfo info, DeviceInfo camDevice, Action< IVideoRecorder > onReady)
- static IVideoPlayer **CreateDefaultVideoPlayer** (ILogger logger, VoicelInfo info, Action< IVideoPlayer > onReady)
- static IPreviewManager **CreateDefaultPreviewManager** (ILogger logger)
- static IVideoRecorder **CreateVideoRecorderUnityTexture** (ILogger logger, VoicelInfo info, DeviceInfo camDevice, Action< IVideoRecorder > onReady)
- static IVideoPlayer **CreateVideoPlayerUnityTexture** (ILogger logger, VoicelInfo info, Action< IVideoPlayer > onReady)
- static IPreviewManager **CreatePreviewManagerUnityTexture** (ILogger logger)

## 3.121 AudioOutDelayControl.PlayDelayConfig Struct Reference

### Public Attributes

- int **Low**
- int **High**
- int **Max**
- int **SpeedUpPerc**

### Static Public Attributes

- static PlayDelayConfig **Default**

## Properties

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

### 3.121.1 Member Data Documentation

#### 3.121.1.1 Default

`PlayDelayConfig Default [static]`

##### Initial value:

```
= new PlayDelayConfig()
{
    Low = 200,
    High = 200,
    Max = 1000,
    SpeedUpPerc = 5,
}
```

## 3.122 PunVoiceClient Class Reference

This class can be used to automatically sync client states between PUN and Voice. It also finds 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 [VoiceFollowClient](#).

### Static Public Attributes

- const string **VoiceRoomNameSuffix** = "\_voice\_"
   
*Suffix for voice room names appended to Leader room names.*

### Protected Member Functions

- override void **Start** ()
- override void **OnDestroy** ()
- override [Speaker](#) **InstantiateSpeakerForRemoteVoice** (int playerId, byte voiceId, object userData)
- override string **GetVoiceRoomName** ()
- override bool **ConnectVoice** ()

### Protected Attributes

- override bool **LeaderInRoom** => PhotonNetwork.InRoom
- override bool **LeaderOfflineMode** => PhotonNetwork.OfflineMode

## Properties

- static `PunVoiceClient Instance` [get]  
*Singleton instance for PunVoiceClient*
- bool `UsePunAppSettings` [get, set]  
*Whether or not to use the Voice AppId and all the other AppSettings from PUN's PhotonServerSettings ScriptableObject singleton in the Voice client/app.*
- bool `UsePunAuthValues` [get, set]  
*Whether or not to use the same PhotonNetwork.AuthValues in PunVoiceClient.Instance.Client.AuthValues. This means that the same UserID will be used in both clients. If custom authentication is used and setup in PUN app, the same configuration should be done for the Voice app.*

## Additional Inherited Members

### 3.122.1 Detailed Description

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

### 3.122.2 Member Data Documentation

#### 3.122.2.1 VoiceRoomNameSuffix

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

Suffix for voice room names appended to Leader room names.

### 3.122.3 Property Documentation

#### 3.122.3.1 Instance

```
PunVoiceClient Instance [static], [get]
```

Singleton instance for `PunVoiceClient`

#### 3.122.3.2 UsePunAppSettings

```
bool UsePunAppSettings [get], [set]
```

Whether or not to use the Voice AppId and all the other AppSettings from PUN's PhotonServerSettings ScriptableObject singleton in the Voice client/app.

### 3.122.3.3 UsePunAuthValues

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

Whether or not to use the same PhotonNetwork.AuthValues in PunVoiceClient.Instance.Client.AuthValues. This means that the same UserID will be used in both clients. If custom authentication is used and setup in PUN app, the same configuration should be done for the [Voice](#) app.

## 3.123 RawCodec Class Reference

### Classes

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

## 3.124 Recorder Class Reference

Component representing outgoing audio stream in scene.

Inherits [VoiceComponent](#).

### Public Types

- enum [InputSourceType](#)
- enum [MicType](#)

### Public Member Functions

- bool [RestartRecording \(\)](#)  
*Restarts recording if [Recorder.RecordingEnabled](#) is true*
- 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.*
- bool [SetIosAudioSessionParameters \(IOS.AudioSessionParameters asp\)](#)  
*Sets the AudioSessionParameters for iOS audio initialization when [Photon MicrophoneType](#) is used.*
- bool [SetIosAudioSessionParameters \(IOS.AudioSessionCategory category, IOS.AudioSessionMode mode, IOS.AudioSessionCategoryOption\[\] options\)](#)  
*Sets the AudioSessionParameters for iOS audio initialization when [Photon MicrophoneType](#) is used.*
- bool [SetAndroidNativeMicrophoneSettings \(bool aec=false, bool agc=false, bool ns=false\)](#)  
*Sets the native Android audio input settings when the [Photon](#) microphone type is used.*
- 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 Attributes

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

## Protected Member Functions

- virtual void **SendPhotonVoiceCreatedMessage** ()
- void **Update** ()

## Properties

- 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.*
- byte **InterestGroup** [get, set]  
*Target interest group that will receive transmitted audio.*
- int[] **TargetPlayers** [get, set]  
*Target players which 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.*
- **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 `RecordingEnabled` [get, set]  
*Gets or sets whether this [Recorder](#) is recording audio to be transmitted.*
- 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 OnAudioFilterRead callback from a muted local AudioSource.*
- 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.*
- bool `RecordWhenJoined` [get, set]  
*If true, recording starts when joining the room and stops when leaving the room.*
- `DeviceInfo MicrophoneDevice` [get, set]
- bool `AndroidMicrophoneAGC` [get]
- bool `AndroidMicrophoneAEC` [get]
- bool `AndroidMicrophoneNS` [get]

## Additional Inherited Members

### 3.124.1 Detailed Description

Component representing outgoing audio stream in scene.

### 3.124.2 Member Function Documentation

#### 3.124.2.1 `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.

### 3.124.2.2 RestartRecording()

```
bool RestartRecording ( )
```

Restarts recording if [Recorder.RecordingEnabled](#) is true

### 3.124.2.3 SetAndroidNativeMicrophoneSettings()

```
bool SetAndroidNativeMicrophoneSettings (
    bool aec = false,
    bool agc = false,
    bool ns = false )
```

Sets the native Android audio input settings when the [Photon](#) microphone type is used.

#### Parameters

<i>aec</i>	Acoustic Echo Cancellation
<i>agc</i>	Automatic Gain Control
<i>ns</i>	Noise Suppression

#### Returns

If a change has been made.

### 3.124.2.4 SetIosAudioSessionParameters() [1/2]

```
bool SetIosAudioSessionParameters (
    IOS.AudioSessionCategory category,
    IOS.AudioSessionMode mode,
    IOS.AudioSessionCategoryOption[] options )
```

Sets the AudioSessionParameters for iOS audio initialization when [Photon](#) MicrophoneType is used.

#### Parameters

<i>category</i>	Audio session category to be used.
<i>mode</i>	Audio session mode to be used.
<i>options</i>	Audio session category options to be used

#### Returns

If a change has been made.

### 3.124.2.5 SetIosAudioSessionParameters() [2/2]

```
bool SetIosAudioSessionParameters (
    IOS.AudioSessionParameters asp )
```

Sets the AudioSessionParameters for iOS audio initialization when [Photon](#) MicrophoneType is used.

#### Parameters

<code>asp</code>	You can use custom value or one from presets, <a href="#">IOS.AudioSessionParametersPresets</a>
------------------	---

#### Returns

If a change has been made.

### 3.124.2.6 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

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

## 3.124.3 Property Documentation

### 3.124.3.1 AudioClip

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

Source audio clip.

### 3.124.3.2 Bitrate

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

Outgoing audio stream bitrate.

### 3.124.3.3 DebugEchoMode

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

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

Initialized with serialized field and can be updated by Editor at runtime.

### 3.124.3.4 Encrypt

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

If true, voice stream is sent encrypted.

Initialized with serialized field and can be updated by Editor at runtime.

### 3.124.3.5 FrameDuration

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

Outgoing audio stream encoder delay.

### 3.124.3.6 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

### 3.124.3.7 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. Initialized with serialized field and can be updated by Editor at runtime.

### 3.124.3.8 IsCurrentlyTransmitting

```
bool IsCurrentlyTransmitting [get]
```

Returns true if audio stream broadcasts.

### 3.124.3.9 LevelMeter

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

Level meter utility.

### 3.124.3.10 LoopAudioClip

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

Loop playback for audio clip sources.

### 3.124.3.11 MicrophoneType

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

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

### 3.124.3.12 RecordingEnabled

```
bool RecordingEnabled [get], [set]
```

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

### 3.124.3.13 RecordWhenJoined

```
bool RecordWhenJoined [get], [set]
```

If true, recording starts when joining the room and stops when leaving the room.

### 3.124.3.14 ReliableMode

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

If true, stream data sent in reliable mode.

Initialized with serialized field and can be updated by Editor at runtime.

### 3.124.3.15 SamplingRate

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

Outgoing audio stream sampling rate.

### 3.124.3.16 SourceType

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

Audio data source.

### 3.124.3.17 StopRecordingWhenPaused

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

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

### 3.124.3.18 TargetPlayers

```
int [] TargetPlayers [get], [set]
```

Target players which will receive transmitted audio.

Initialized with serialized field and can be updated by Editor at runtime

### 3.124.3.19 TransmitEnabled

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

If true, audio transmission is enabled.

### 3.124.3.20 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.

### 3.124.3.21 UseOnAudioFilterRead

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

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

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

### 3.124.3.22 UserData

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

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

### 3.124.3.23 VoiceDetection

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

If true, voice detection enabled.

### 3.124.3.24 VoiceDetectionDelayMs

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

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

### 3.124.3.25 VoiceDetectionThreshold

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

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

### 3.124.3.26 VoiceDetector

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

Returns voice activity detector for recorder's audio stream.

### 3.124.3.27 VoiceDetectorCalibrating

```
bool VoiceDetectorCalibrating [get]
```

If true, voice detector calibration is in progress.

## 3.125 RecorderPreset Class Reference

Inherits [VoiceComponent](#).

### Classes

- struct [DSP](#)

### Public Attributes

- RuntimePlatform **Platform**
- MicType **MicrophoneType**
- bool **DSPEnabled**
- [DSP](#) **DSPSettings**

### Protected Member Functions

- override void **Awake** ()

### Additional Inherited Members

## 3.126 RemoteVoiceInfo Class Reference

Information about a remote voice (incoming stream).

### Properties

- [VoicelInfo](#) **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).*

### 3.126.1 Detailed Description

Information about a remote voice (incoming stream).

### 3.126.2 Property Documentation

#### 3.126.2.1 ChannelId

```
int ChannelId [get]
```

ID of channel used for transmission.

#### 3.126.2.2 Info

```
VoiceInfo Info [get]
```

Remote voice info.

#### 3.126.2.3 PlayerId

```
int PlayerId [get]
```

Player ID of voice owner.

#### 3.126.2.4 Voiceld

```
byte VoiceId [get]
```

Voice ID (unique in the room).

## 3.127 RemoteVoiceLink Class Reference

### Public Member Functions

- **RemoteVoiceLink** ([VoiceInfo](#) info, int playerId, byte voiceld, int channelld, ref [RemoteVoiceOptions](#) options)
- override string **ToString ()**

## Public Attributes

- readonly [VoicelInfo](#) **VoicelInfo**
- readonly int **PlayerId**
- readonly byte **VoiceId**
- readonly int **ChannelId**

## Events

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

## 3.128 RemoteVoiceOptions Struct Reference

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

### Public Member Functions

- **RemoteVoiceOptions** ([ILogger](#) logger, string logPrefix, [VoicelInfo](#) voiceInfo)
- void **SetOutput** (Action<[FrameOut](#)< float >> output)  
*Create default audio decoder and register a method to be called when a data frame is decoded.*
- void **SetOutput** (Action<[FrameOut](#)< short >> output)  
*Create default audio decoder and register a method to be called when a data frame is decoded.*

### 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.*

### 3.128.1 Detailed Description

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

### 3.128.2 Member Function Documentation

#### 3.128.2.1 SetOutput() [1/2]

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

Create default audio decoder and register a method to be called when a data frame is decoded.

### 3.128.2.2 SetOutput() [2/2]

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

Create default audio decoder and register a method to be called when a data frame is decoded.

## 3.128.3 Property Documentation

### 3.128.3.1 Decoder

[IDecoder](#) Decoder [get], [set]

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

### 3.128.3.2 OnRemoteVoiceRemoveAction

Action OnRemoteVoiceRemoveAction [get], [set]

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

## 3.129 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 data.*
- void [Dispose](#) ()

### Protected Attributes

- T[] [frameResampled](#)

### 3.129.1 Detailed Description

Sample-rate conversion Audio Processor.

This processor converts the sample-rate of the source stream. Internally, it uses [AudioUtil.Resample<T>\(T\[\], T\[\], int, int\)](#).

### 3.129.2 Constructor & Destructor Documentation

#### 3.129.2.1 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.

### 3.129.3 Member Function Documentation

#### 3.129.3.1 Process()

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

Process a frame of data.

##### Parameters

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

##### Returns

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

Implements [IProcessor< T >](#).

### 3.130 SaveIncomingStreamToFile Class Reference

Inherits [VoiceComponent](#).

#### Protected Member Functions

- override void **Awake** ()

#### Additional Inherited Members

### 3.131 SaveOutgoingStreamToFile Class Reference

Inherits [VoiceComponent](#).

#### Additional Inherited Members

### 3.132 SendFrameParams Struct Reference

#### Public Member Functions

- **SendFrameParams** (bool targetMe, int[] targetPlayers, byte interestGroup, bool reliable, bool encrypt)

#### Properties

- bool **TargetMe** [get]
- int[] **TargetPlayers** [get]
- byte **InterestGroup** [get]
- bool **Reliable** [get]
- bool **Encrypt** [get]

### 3.133 RawCodec.ShortToFloat Class Reference

#### Public Member Functions

- **ShortToFloat** (Action< [FrameOut](#)< float >> output)
- void **Output** ([FrameOut](#)< short > shortBuf)

### 3.134 Speaker Class Reference

Inherits [VoiceComponent](#).

Inherited by [SpeakerFMOD](#), and [SpeakerAudioFilterRead](#).

## Public Member Functions

- void **RestartPlayback ()**

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

## Protected Member Functions

- override void **Awake ()**
- virtual **IAudioOut< float > CreateAudioOut ()**
- virtual void **OnDestroy ()**
- void **Update ()**

## Protected Attributes

- **IAudioOut< float > audioOutput**
- **AudioOutDelayControl.PlayDelayConfig playDelayConfig** = AudioOutDelayControl.PlayDelayConfig.Default
- bool **restartOnDeviceChange** = true

## Properties

- bool **IsPlaying** [get]  
*Is the speaker playing right now.*
- int? **Lag** [get]  
*The current difference between positions in the buffer of (jittery) stream writer and (clock-driven) audio output reader in ms.*
- Action< **Speaker** > **OnRemoteVoiceRemoveAction** [get, set]  
*Register a method to be called when remote voice removed.*
- **RemoteVoiceLink RemoteVoice** [get]
- bool **IsLinked** [get]  
*Whether or not this **Speaker** has been linked to a remote voice stream.*
- **AudioOutDelayControl.PlayDelayConfig PlayDelayConfig** [get, set]  
*Gets or sets jitter buffer config.*
- int **PlayDelay** [get, set]  
*Gets or sets jitter buffer size in ms.*
- bool **RestartOnDeviceChange** [get, set]

## Additional Inherited Members

### 3.134.1 Member Function Documentation

#### 3.134.1.1 **RestartPlayback()**

```
void RestartPlayback ( )
```

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

#### Returns

True if playback is successfully restarted.

### 3.134.2 Property Documentation

#### 3.134.2.1 IsLinked

```
bool IsLinked [get]
```

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

#### 3.134.2.2 IsPlaying

```
bool IsPlaying [get]
```

Is the speaker playing right now.

#### 3.134.2.3 Lag

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

The current difference between positions in the buffer of (jittery) stream writer and (clock-driven) audio output reader in ms.

#### 3.134.2.4 OnRemoteVoiceRemoveAction

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

Register a method to be called when remote voice removed.

#### 3.134.2.5 PlayDelay

```
int PlayDelay [get], [set]
```

Gets or sets jitter buffer size in ms.

The method updates `PlayDelayConfig` with reasonable values based on the single value provided. Use [PlayDelayConfig](#) for more precise control.

### 3.134.2.6 PlayDelayConfig

```
AudioOutDelayControl.PlayDelayConfig PlayDelayConfig [get], [set]
```

Gets or sets jitter buffer config.

Make sure that the new value is fully initialized or built from AudioOutDelayControl.PlayDelayConfig.Default.

## 3.135 SpeakerAudioFilterRead Class Reference

Inherits [Speaker](#).

### Protected Member Functions

- override `IAudioOut< float > CreateAudioOut ()`

### Additional Inherited Members

## 3.136 SpeakerFMOD Class Reference

Inherits [Speaker](#).

### Protected Member Functions

- override `IAudioOut< float > CreateAudioOut ()`

### Additional Inherited Members

## 3.137 ImageBufferInfo.StrideSet Struct Reference

### Public Member Functions

- `StrideSet (int length, int s0=0, int s1=0, int s2=0, int s3=0)`

### Properties

- `int this[int key] [get, set]`
- `int Length [get]`

### 3.138 `AudioUtil.TempoUp< T >` Class Template Reference

#### Public Member Functions

- void **Begin** (int channels, int changePerc, int skipGroup)
- int **Process** (T[] s, T[] d)
- int **End** (T[] s)
- int **endFloat** (float[] s)
- int **endShort** (short[] s)

### 3.139 `TestTone` Class Reference

Inherits MonoBehaviour.

### 3.140 `TimerWorker` Class Reference

#### Public Member Functions

- **TimerWorker** (Action callback, int interval)
- void **Stop** ()

#### Static Public Member Functions

- static void **CallbackStatic** (int handle)

### 3.141 `AudioUtil.ToneAudioPusher< T >` Class Template Reference

IAudioPusher that provides a constant tone signal.

Inherits [AudioUtil.GeneratorPusher< T >](#).

#### Public Member Functions

- **ToneAudioPusher** (int frequency=440, int bufSizeMs=100, int samplingRate=48000, int channels=1)  
*Create a new [ToneAudioReader](#) instance*

#### Protected Member Functions

- override int **Gen** (T[] buf, long timeSamples)

## Additional Inherited Members

### 3.141.1 Detailed Description

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

### 3.141.2 Constructor & Destructor Documentation

#### 3.141.2.1 ToneAudioPusher()

```
ToneAudioPusher (
    int frequency = 440,
    int bufSizeMs = 100,
    int samplingRate = 48000,
    int channels = 1 )
```

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.

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

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

Inherits [AudioUtil.GeneratorReader< T >](#).

## Public Member Functions

- [ToneAudioReader](#) (`Func< double > clockSec=null, double frequency=440, int samplingRate=48000, int channels=1)`

*Create a new [ToneAudioReader](#) instance*

## Protected Member Functions

- `override int Gen (T[] buf, long timeSamples)`

## Additional Inherited Members

### 3.142.1 Detailed Description

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

Because of current resampling algorithm, the tone is distorted if SamplingRate does not equal encoder sampling rate.

### 3.142.2 Constructor & Destructor Documentation

#### 3.142.2.1 ToneAudioReader()

```
ToneAudioReader (
    Func< double > clockSec = null,
    double frequency = 440,
    int samplingRate = 48000,
    int channels = 1 )
```

Create a new [ToneAudioReader](#) instance

#### Parameters

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

## 3.143 UnityAudioOut Class Reference

Inherits [AudioOutDelayControl< float >](#).

### Public Member Functions

- **UnityAudioOut** (ILogger logger, string logPrefix, bool debugInfo)
- override void **OutCreate** (int frequency, int channels, int bufferSamples)
- override void **OutStart** ()
- override void **OutWrite** (float[] data, int offsetSamples)
- override void **Stop** ()

### Protected Attributes

- readonly AudioSource **source**
- AudioClip **clip**

## Properties

- override long **OutPos** [get]

## 3.144 UnityLogger Class Reference

### Static Public Member Functions

- static void **Log** (LogLevel level, Object obj, string tag, string objName, string fmt, params object[] args)

## 3.145 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)
- static string **CheckDevice** ([Voice.ILogger](#) logger, string logPref, string device, int suggestedFrequency, out int frequency)

## Properties

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

### 3.145.1 Detailed Description

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

## 3.146 UnityVoiceClient Class Reference

Component that represents a [Voice](#) client and manages a simple [Unity](#) integration: a single [Recorder](#) and multiple remote speakers.

Inherits [VoiceConnection](#).

### Public Member Functions

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

## Public Attributes

- override bool **AlwaysUsePrimaryRecorder** => true
- bool **UseVoiceAppSettings** = false

*Whether or not to use the [Voice AppId](#) and all the other AppSettings from Fusion's RealtimeAppSettings Scriptable Object singleton in the [Voice](#) client/app.*

## Protected Member Functions

- virtual void **Start** ()
- override **Speaker InstantiateSpeakerForRemoteVoice** (int playerId, byte voiceId, object userData)

## Additional Inherited Members

### 3.146.1 Detailed Description

Component that represents a [Voice](#) client and manages a simple [Unity](#) integration: a single [Recorder](#) and multiple remote speakers.

### 3.146.2 Member Function Documentation

#### 3.146.2.1 ConnectUsingSettings()

```
override bool ConnectUsingSettings (
    AppSettings overwriteSettings = null ) [virtual]
```

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

Reimplemented from [VoiceConnection](#).

### 3.146.3 Member Data Documentation

### 3.146.3.1 UseVoiceAppSettings

```
bool UseVoiceAppSettings = false
```

Whether or not to use the [Voice](#) AppId and all the other AppSettings from [Fusion](#)'s RealtimeAppSettings Scriptable Object singleton in the [Voice](#) client/app.

## 3.147 UnsupportedCodecException Class Reference

Exception thrown if an unsupported codec is encountered.

Inherits Exception.

### Public Member Functions

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

### 3.147.1 Detailed Description

Exception thrown if an unsupported codec is encountered.

### 3.147.2 Constructor & Destructor Documentation

#### 3.147.2.1 UnsupportedCodecException()

```
UnsupportedCodecException (
    string info,
    Codec codec )
```

Create a new [UnsupportedCodecException](#).

##### Parameters

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

## 3.148 UnsupportedPlatformException Class Reference

Exception thrown if an unsupported platform is encountered.

Inherits Exception.

## Public Member Functions

- [UnsupportedPlatformException](#) (string subject, string platform=null)  
*Create a new [UnsupportedPlatformException](#).*

### 3.148.1 Detailed Description

Exception thrown if an unsupported platform is encountered.

### 3.148.2 Constructor & Destructor Documentation

#### 3.148.2.1 UnsupportedPlatformException()

```
UnsupportedPlatformException (
    string subject,
    string platform = null )
```

Create a new [UnsupportedPlatformException](#).

##### Parameters

<i>subject</i>	The info prepending standard message.
----------------	---------------------------------------

///

##### Parameters

<i>platform</i>	Optional platform name.
-----------------	-------------------------

## 3.149 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](#).*

### 3.149.1 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.

### 3.149.2 Constructor & Destructor Documentation

#### 3.149.2.1 UnsupportedSampleTypeException()

```
UnsupportedSampleTypeException (
```

Type	<i>t</i>	)
------	----------	---

Create a new [UnsupportedSampleTypeException](#).

##### Parameters

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

## 3.150 OpusCodec.Util Class Reference

### 3.151 Utility Class Reference

#### Static Public Member Functions

- static bool **IsAudio** (this [Codec](#) c)
- static bool **IsVideo** (this [Codec](#) c)

### 3.152 VideoInEnumerator Class Reference

Inherits [DeviceEnumeratorBase](#).

#### Public Member Functions

- **VideoInEnumerator** ([ILogger](#) logger)
- override void **Refresh** ()
- override void **Dispose** ()

#### Properties

- override string **Error** [get]

## Additional Inherited Members

### 3.153 VideoInEnumerator Class Reference

Inherits [DeviceEnumerator](#).

#### Public Member Functions

- [VideoInEnumerator \(ILogger logger\)](#)

## Additional Inherited Members

### 3.154 VoiceClient Class Reference

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

Inherits [IDisposable](#).

#### Classes

- struct [CreateOptions](#)

#### 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 [LogSpacingProfiles](#) ()
- void [LogStats](#) ()
- void [SetRemoteVoiceDelayFrames](#) ([Codec](#) codec, int delayFrames)
- void [SetRemoteVoiceDelayFrames](#) (Func< [Codec](#), bool > predicate, int delayFrames)
- [VoiceClient](#) ([IVoiceTransport](#) transport, [ILogger](#) logger, [CreateOptions](#) opt=default([CreateOptions](#)))
 

*Creates [VoiceClient](#) instance*
- 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, [VoiceCreateOptions](#) options=default([VoiceCreateOptions](#)))
 

*Creates basic outgoing stream w/o data processing support. Provided encoder should generate output data stream.*
- [LocalVoiceAudio< T > CreateLocalVoiceAudio< T >](#) ([VoiceInfo](#) voiceInfo, [IAudioDesc](#) audioSourceDesc, int channelId, [VoiceCreateOptions](#) options=default([VoiceCreateOptions](#)))
- [LocalVoice CreateLocalVoiceAudioFromSource](#) ([VoiceInfo](#) voiceInfo, [IAudioDesc](#) source, [AudioSampleType](#) sampleType, int channelId, [VoiceCreateOptions](#) options=default([VoiceCreateOptions](#)))
 

*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.*
- [LocalVoiceVideo CreateLocalVoiceVideo](#) ([VoiceInfo](#) voiceInfo, [IVideoRecorder](#) recorder, int channelId, [VoiceCreateOptions](#) options=default([VoiceCreateOptions](#)))
 

*Creates outgoing video stream consuming sequence of image buffers.*
- void [RemoveLocalVoice](#) ([LocalVoice](#) voice)
 

*Removes local voice (outgoing data stream).*

**Parameters**

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

- void **onJoinChannel** (int channelld)
- void **onJoinAllChannels** ()
- void **onLeaveChannel** (int channel)
- void **onLeaveAllChannels** ()
- void **onPlayerJoin** (int channelld, int playerId)
- void **onPlayerJoin** (int playerId)
- void **onPlayerLeave** (int channelld, int playerId)
- void **onPlayerLeave** (int playerId)
- void **onVoiceInfo** (int channelld, int playerId, byte voiceld, int eventBufferSize, [VoiceInfo](#) info)
- void **onVoiceRemove** (int playerId, byte[] voicelds)
- void **onFrame** (int playerId, byte voiceld, ushort evNumber, ref [FrameBuffer](#) receivedBytes, bool isLocal← Player)
- void **Dispose** ()

**Properties**

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

*Lost events counter (the number of empty frames sent to the deocder).*
- int **EventsLost** [get, set]
 

*Lost frames counter (the number of empty frames sent to the deocder).*
- int **FramesLost** [get, set]
 

*Recovered frames counter.*
- int **FramesFragPart** [get, set]
 

*The counter of assembled frames, fragments of which are partially missing.*
- int **FramesRecovered** [get, set]
 

*Recovered frames counter.*
- int **FramesLate** [get, set]
 

*Counter of late (incorrectly ordered) frames.*
- int **FramesReceived** [get]
 

*Received frames counter.*
- int **FramesReceivedFEC** [get, set]
 

*Received FEC events counter.*
- int **FramesTryFEC** [get, set]
 

*FEC recovery attempts counter.*
- int **FramesReceivedFragments** [get, set]
 

*Received events for fragmented frames counter.*
- int **FramesReceivedFragmented** [get, set]
 

*Assembled fragmented 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). Method 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.*

### 3.154.1 Detailed Description

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

### 3.154.2 Constructor & Destructor Documentation

#### 3.154.2.1 VoiceClient()

```
VoiceClient (
    IVoiceTransport transport,
    ILogger logger,
    CreateOptions opt = default(CreateOptions) )
```

Creates [VoiceClient](#) instance

### 3.154.3 Member Function Documentation

#### 3.154.3.1 CreateLocalVoice()

```
LocalVoice CreateLocalVoice (
    VoiceInfo voiceInfo,
    int channelId,
    VoiceCreateOptions options = default(VoiceCreateOptions) )
```

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

#### Parameters

<code>voiceInfo</code>	Outgoing stream parameters.
<code>channelId</code>	Transport channel specific to transport.
<code>options</code>	<a href="#">Voice</a> creation options.

**Returns**

Outgoing stream handler.

**3.154.3.2 CreateLocalVoiceAudioFromSource()**

```
LocalVoice CreateLocalVoiceAudioFromSource (
    VoiceInfo voiceInfo,
    IAudioDesc source,
    AudioSampleType sampleType,
    int channelId,
    VoiceCreateOptions options = default(VoiceCreateOptions) )
```

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 stream parameters.
<i>source</i>	Streaming audio source.
<i>sampleType</i>	<i>Voice</i> 's audio sample type. If does not match source audio sample type, conversion will occur.
<i>channelId</i>	Transport channel specific to transport.
<i>options</i>	<i>Voice</i> creation options.

**Returns**

Outgoing stream handler.

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

**3.154.3.3 CreateLocalVoiceVideo()**

```
LocalVoiceVideo CreateLocalVoiceVideo (
    VoiceInfo voiceInfo,
    IVideoRecorder recorder,
    int channelId,
    VoiceCreateOptions options = default(VoiceCreateOptions) )
```

Creates outgoing video stream consuming sequence of image buffers.

**Parameters**

<i>voiceInfo</i>	Outgoing stream parameters.
<i>recorder</i>	Video recorder.
<i>channelId</i>	Transport channel specific to transport.
<i>options</i>	<i>Voice</i> creation options.

**Returns**

Outgoing stream handler.

**3.154.3.4 LocalVoicesInChannel()**

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

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

**3.154.3.5 RemoteVoiceInfoDelegate()**

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

Remote voice info event delegate.

**3.154.3.6 RemoveLocalVoice()**

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

Removes local voice (outgoing data stream).

**Parameters**

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

**3.154.3.7 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).

### 3.154.4 Property Documentation

#### 3.154.4.1 DebugLostPercent

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

Lost frames simulation ratio.

#### 3.154.4.2 EventsLost

```
int EventsLost [get], [set]
```

Lost events counter (the number of empty frames sent to the deocder).

#### 3.154.4.3 FramesFragPart

```
int FramesFragPart [get], [set]
```

The counter of assembled frames, fragments of which are partially missing.

#### 3.154.4.4 FramesLate

```
int FramesLate [get], [set]
```

Counter of late (incorrectly ordered) frames.

#### 3.154.4.5 FramesLost

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

Lost frames counter (the number of empty frames sent to the deocder).

### 3.154.4.6 FramesReceived

```
int FramesReceived [get]
```

Received frames counter.

### 3.154.4.7 FramesReceivedFEC

```
int FramesReceivedFEC [get], [set]
```

Received FEC events counter.

### 3.154.4.8 FramesReceivedFragmented

```
int FramesReceivedFragmented [get], [set]
```

Assembled fragmented frames counter.

### 3.154.4.9 FramesReceivedFragments

```
int FramesReceivedFragments [get], [set]
```

Received events for fragmented frames counter.

### 3.154.4.10 FramesRecovered

```
int FramesRecovered [get], [set]
```

Recovered frames counter.

### 3.154.4.11 FramesSent

```
int FramesSent [get]
```

Sent frames counter.

#### 3.154.4.12 FramesSentBytes

```
int FramesSentBytes [get]
```

Sent frames bytes counter.

#### 3.154.4.13 FramesTryFEC

```
int FramesTryFEC [get], [set]
```

FEC recovery attempts counter.

#### 3.154.4.14 LocalVoices

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

Iterates through copy of all local voices list.

#### 3.154.4.15 OnRemoteVoiceInfoAction

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

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

#### 3.154.4.16 RemoteVoiceInfos

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

Iterates through all remote voices infos.

#### 3.154.4.17 RoundTripTime

```
int RoundTripTime [get]
```

Average time required voice packet to return to sender.

### 3.154.4.18 RoundTripTimeVariance

```
int RoundTripTimeVariance [get]
```

Average round trip time variation.

### 3.154.4.19 SuppressInfoDuplicateWarning

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

Do not log warning when duplicate info received.

## 3.155 VoiceComponent Class Reference

Inherits MonoBehaviour.

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

### Public Attributes

- [VoiceLogger](#) **VoiceLogger** => impl.VoiceLogger

### Protected Member Functions

- virtual void **Awake** ()

### Protected Attributes

- [Voice.ILogger](#) **Logger** => impl.Logger

### Properties

- string **Name** [set]

## 3.156 VoiceComponentImpl Class Reference

### Public Member Functions

- void **Awake** (MonoBehaviour mb)

## Public Attributes

- `Voice.ILogger Logger => logger`
- `VoiceLogger VoiceLogger => voiceLogger`

## Properties

- string `Name [set]`

## 3.157 VoiceConnection Class Reference

Component that represents a [Voice](#) client.

Inherits `ConnectionHandler`.

Inherited by [UnityVoiceClient](#), and [VoiceFollowClient](#).

## Public Member Functions

- virtual bool `ConnectUsingSettings (AppSettings overwriteSettings=null)`  
*Connect to `Photon` server using `Settings`*
- bool `AddSpeaker (Speaker speaker, object userData)`  
*Tries to link local `Speaker` with remote voice stream using `UserData`. Useful if `Speaker` created after stream is started.*
- `Speaker InstantiateSpeakerPrefab (GameObject parent, bool destroyOnRemove)`  
*Instantiates `SpeakerPrefab`, optionally attaches it to the provided parent.*
- bool `AddRecorder (Recorder rec)`
- void `RemoveRecorder (Recorder rec)`

## Public Attributes

- virtual bool `AlwaysUsePrimaryRecorder => false`
- AppSettings `Settings`  
*Settings to be used by this `Voice` Client*
- `Voice.ILogger Logger => voiceComponentImpl.Logger`
- `VoiceLogger VoiceLogger => voiceComponentImpl.VoiceLogger`
- bool `UsePrimaryRecorder => this.usePrimaryRecorder`  
*Use `VoiceConnection.PrimaryRecorder` directly.*

## Static Public Attributes

- const int `ChannelAudio = 1`  
*Recommended `Photon` Transport channel for audio. Chosen not to interfere with video and default channel.*
- const int `ChannelVideo = 2`  
*Recommended `Photon` Transport channel for video. Chosen not to interfere with audio and default channel.*

## Protected Member Functions

- override void **Awake** ()
- virtual void **Update** ()
- virtual void **FixedUpdate** ()
- virtual void **OnDestroy** ()
- virtual **Speaker** **InstantiateSpeakerForRemoteVoice** (int playerId, byte voiceId, object userData)
- virtual void **OnVoiceStateChanged** (**ClientState** fromState, **ClientState** toState)
- void **CalcStatistics** ()
- virtual void **OnOperationResponseReceived** (**OperationResponse** operationResponse)

## Properties

- 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]
 

*Primary Recorder to be used by VoiceConnection implementations directly or via integration objects.*
- string **BestRegionSummaryInPreferences** [get, set]
 

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

## 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*

### 3.157.1 Detailed Description

Component that represents a **Voice** client.

### 3.157.2 Member Function Documentation

#### 3.157.2.1 AddSpeaker()

```
bool AddSpeaker (
    Speaker speaker,
    object userData )
```

Tries to link local **Speaker** with remote voice stream using **Userdata**. Useful if **Speaker** created after stream is started.

## Parameters

<i>speaker</i>	<a href="#">Speaker</a> or try linking.
<i>userData</i>	UserData object used to bind local <a href="#">Speaker</a> with remote voice stream.

## Returns

**3.157.2.2 ConnectUsingSettings()**

```
virtual bool ConnectUsingSettings (
    AppSettings overwriteSettings = null ) [virtual]
```

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

Reimplemented in [UnityVoiceClient](#).

**3.157.2.3 InstantiateSpeakerPrefab()**

```
Speaker InstantiateSpeakerPrefab (
    GameObject parent,
    bool destroyOnRemove )
```

Instantiates [SpeakerPrefab](#), optionally attaches it to the provided parent.

[VoiceConnection](#) manages the instantiated object (destroys on [OnRemoteVoiceRemoveAction](#)).

## Parameters

<i>parent</i>	The object to attach Speaker to.
<i>destroyOnRemove</i>	Automatically destroy instantiated prefab when remote voice is removed (the caller does not manage the instance).

**Returns**

Instantiated [Speaker](#) or null.

### 3.157.3 Member Data Documentation

#### 3.157.3.1 ChannelAudio

```
const int ChannelAudio = 1 [static]
```

Recommended [Photon](#) Transport channel for audio. Chosen not to interfere with video and default channel.

#### 3.157.3.2 ChannelVideo

```
const int ChannelVideo = 2 [static]
```

Recommended [Photon](#) Transport channel for video. Chosen not to interfere with audio and default channel.

#### 3.157.3.3 Settings

AppSettings Settings

Settings to be used by this [Voice](#) Client

#### 3.157.3.4 UsePrimaryRecorder

```
bool UsePrimaryRecorder => this.usePrimaryRecorder
```

Use [VoiceConnection.PrimaryRecorder](#) directly.

### 3.157.4 Property Documentation

#### 3.157.4.1 BestRegionSummaryInPreferences

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

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

### 3.157.4.2 ClientState

```
ClientState ClientState [get]
```

Returns [Photon Voice](#) client state.

### 3.157.4.3 FramesLostPercent

```
float FramesLostPercent [get]
```

Percentage of lost frames.

### 3.157.4.4 FramesLostPerSecond

```
float FramesLostPerSecond [get]
```

Number of frames lost per second.

### 3.157.4.5 FramesReceivedPerSecond

```
float FramesReceivedPerSecond [get]
```

Number of frames received per second.

### 3.157.4.6 PrimaryRecorder

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

Primary [Recorder](#) to be used by [VoiceConnection](#) implementations directly or via integration objects.

### 3.157.4.7 SpeakerPrefab

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

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

### 3.157.4.8 VoiceClient

`VoiceClient VoiceClient [get]`

Returns underlying [Photon Voice](#) client.

## 3.157.5 Event Documentation

### 3.157.5.1 RemoteVoiceAdded

`Action<RemoteVoiceLink> RemoteVoiceAdded`

Fires when a remote voice stream is added

### 3.157.5.2 SpeakerLinked

`Action<Speaker> SpeakerLinked`

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

## 3.158 VoiceCreateOptions Struct Reference

Used to initialize optional properties of the [LocalVoice](#) instance at creation time.

### Public Attributes

- `IEncoder Encoder`  
*Encoder.*
- `byte InterestGroup`  
*See [LocalVoice.InterestGroup](#).*
- `int[] TargetPlayers`  
*See [LocalVoice.TargetPlayers](#) Set to [] to make sure that [LocalVoice](#) does not create Remote voices during creation*
- `bool DebugEchoMode`  
*See [LocalVoice.DebugEchoMode](#).*
- `bool Reliable`  
*See [LocalVoice.Reliable](#).*
- `bool Encrypt`  
*See [LocalVoice.Encrypt](#).*
- `bool Fragment`  
*See [LocalVoice.Fragment](#).*
- `int FEC`  
*See [LocalVoice.FEC](#).*
- `int EventBufSize`  
*Receiver's ring buffer size (default is 256)*

### 3.158.1 Detailed Description

Used to initialize optional properties of the [LocalVoice](#) instance at creation time.

### 3.158.2 Member Data Documentation

#### 3.158.2.1 DebugEchoMode

```
bool DebugEchoMode
```

See [LocalVoice.DebugEchoMode](#).

#### 3.158.2.2 Encoder

```
IEncoder Encoder
```

Encoder.

#### 3.158.2.3 Encrypt

```
bool Encrypt
```

See [LocalVoice.Encrypt](#).

#### 3.158.2.4 EventBufSize

```
int EventBufSize
```

Receiver's ring buffer size (default is 256)

#### 3.158.2.5 FEC

```
int FEC
```

See [LocalVoice.FEC](#).

### 3.158.2.6 Fragment

```
bool Fragment
```

See [LocalVoice.Fragment](#).

### 3.158.2.7 InterestGroup

```
byte InterestGroup
```

See [LocalVoice.InterestGroup](#).

### 3.158.2.8 Reliable

```
bool Reliable
```

See [LocalVoice.Reliable](#).

### 3.158.2.9 TargetPlayers

```
int [] TargetPlayers
```

See [LocalVoice.TargetPlayers](#) Set to [] to make sure that LocalVoice does not create Remote voices during creation

## 3.159 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.RecordingEnabled 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?!).*

### 3.159.1 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.

### 3.159.2 Member Data Documentation

#### 3.159.2.1 DisableVad

```
bool DisableVad
```

Disable recorder.VoiceDetection for easier testing.

#### 3.159.2.2 ForceRecordingAndTransmission

```
bool ForceRecordingAndTransmission
```

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

#### 3.159.2.3 IncreaseLogLevels

```
bool IncreaseLogLevels
```

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

### 3.159.2.4 LocalDebug

bool LocalDebug

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

### 3.159.2.5 TestAudioClip

AudioClip TestAudioClip

Audio file to be broadcast when TestUsingAudioClip is enabled.

### 3.159.2.6 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.

## 3.160 **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 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.*

### 3.160.1 Detailed Description

Simple voice activity detector triggered by signal level.

### 3.160.2 Member Function Documentation

#### 3.160.2.1 Process()

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

Process a frame of data.

##### Parameters

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

##### Returns

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

Implements [IProcessor< T >](#).

### 3.160.3 Property Documentation

### 3.160.3.1 ActivityDelayMs

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

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

### 3.160.3.2 Detected

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

If true, voice detected.

### 3.160.3.3 DetectedTime

```
DateTime DetectedTime [get]
```

Last time when switched to detected state.

### 3.160.3.4 On

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

If true, voice detection enabled.

### 3.160.3.5 Threshold

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

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

## 3.160.4 Event Documentation

### 3.160.4.1 OnDetected

Action OnDetected

Called when switched to detected state.

## 3.161 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 data.*
- void [Dispose](#) ()

### Protected Attributes

- int [calibrateCount](#)

### Properties

- bool [IsCalibrating](#) [get]

#### 3.161.1 Detailed Description

Calibration Utility for Voice Detector

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

#### 3.161.2 Constructor & Destructor Documentation

##### 3.161.2.1 VoiceDetectorCalibration()

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

Create new [VoiceDetectorCalibration](#) instance.

**Parameters**

<i>voiceDetector</i>	Voice 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>channels</i>	Number of channels in the audio signal.

**3.161.3 Member Function Documentation****3.161.3.1 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>	Optional callback that is called after calibration is complete.

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.

**3.161.3.2 Process()**

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

Process a frame of data.

**Parameters**

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

**Returns**

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

Implements [IProcessor< T >](#).

## 3.162 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

#### 3.162.1 Detailed Description

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

## 3.163 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

#### 3.163.1 Detailed Description

[VoiceDetector](#) specialization for float audio.

#### 3.163.2 Constructor & Destructor Documentation

##### 3.163.2.1 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.

## 3.164 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

#### 3.164.1 Detailed Description

[VoiceDetector](#) specialization for float audio.

#### 3.164.2 Constructor & Destructor Documentation

##### 3.164.2.1 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.

## 3.165 VoiceEvent Class Reference

### Static Public Attributes

- const byte **Code** = 202  
*Single event used for voice communications.*
- const byte **FrameCode** = 203

### 3.165.1 Member Data Documentation

#### 3.165.1.1 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.

## 3.166 VoiceFollowClient Class Reference

This class can be used to automatically sync client states between Leader and [Voice](#) clients.

Inherits [VoiceConnection](#).

Inherited by [FusionVoiceClient](#), and [PunVoiceClient](#).

### 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 Leader client is joined to a Leader room*

### Protected Member Functions

- abstract string [GetVoiceRoomName](#) ()
- abstract bool [ConnectVoice](#) ()
- virtual void [Start](#) ()
- override void [OnDestroy](#) ()
- override void [OnOperationResponseReceived](#) (OperationResponse operationResponse)
- void [LeaderStateChanged](#) ([ClientState](#) toState)
- override void [OnVoiceStateChanged](#) ([ClientState](#) fromState, [ClientState](#) toState)
- virtual bool [JoinVoiceRoom](#) (string voiceRoomName)

## Properties

- abstract bool **LeaderInRoom** [get]
- abstract bool **LeaderOfflineMode** [get]

## Additional Inherited Members

### 3.166.1 Detailed Description

This class can be used to automatically sync client states between Leader and [Voice](#) clients.

### 3.166.2 Member Function Documentation

#### 3.166.2.1 ConnectAndJoinRoom()

```
bool ConnectAndJoinRoom ( )
```

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

##### Returns

If true, connection command send from client

#### 3.166.2.2 Disconnect()

```
void Disconnect ( )
```

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

### 3.166.3 Member Data Documentation

#### 3.166.3.1 AutoConnectAndJoin

```
bool AutoConnectAndJoin = true
```

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

## 3.167 VoicelInfo Struct Reference

Describes stream properties.

### Public Member Functions

- override string **ToString ()**

### Static Public Member Functions

- static **VoiceInfo CreateAudioOpus** (POpusCodec.Enums.SamplingRate samplingRate, int channels, OpusCodec.FrameDuration frameDurationUs, int bitrate, object userdata=null)  
*Create stream info for an Opus audio stream.*
- static **VoiceInfo CreateAudio** (**Codec** codec, int samplingRate, int channels, int frameDurationUs, object userdata=null)  
*Create stream info for an audio stream.*
- static **VoiceInfo CreateVideo** (**Codec** codec, int bitrate, int width, int height, int fps, int keyFrameInt, object userdata=null)  
*Create stream info for a video stream.*

### Properties

- **Codec Codec** [get, set]  
    *Audio sampling rate (frequency, in Hz).*
- int **SamplingRate** [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.*

### 3.167.1 Detailed Description

Describes stream properties.

### 3.167.2 Member Function Documentation

#### 3.167.2.1 CreateAudio()

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

Create stream info for an audio stream.

##### Parameters

<i>codec</i>	Audio codec.
<i>samplingRate</i>	Audio sampling rate.
<i>channels</i>	Number of channels.
<i>frameDurationUs</i>	Uncompressed frame (audio packet) size in microseconds.
<i>userdata</i>	Optional user data. Should be serializable by <a href="#">Photon</a> .

##### Returns

[VoiceInfo](#) instance.

#### 3.167.2.2 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**

[VoicelInfo](#) instance.

### 3.167.2.3 CreateVideo()

```
static VoiceInfo CreateVideo (
    Codec codec,
    int bitrate,
    int width,
    int height,
    int fps,
    int keyFrameInt,
    object userdata = null ) [static]
```

Create stream info for a video stream.

**Parameters**

<i>codec</i>	Video codec.
<i>bitrate</i>	Stream bitrate.
<i>width</i>	Streamed video width. If 0, width and height of video source used (no rescaling).
<i>height</i>	Streamed video height. If -1, aspect ratio preserved during rescaling.
<i>fps</i>	Streamed video frames per second.
<i>keyFrameInt</i>	Keyframes interval in frames.

///

**Parameters**

<i>userdata</i>	Optional user data. Should be serializable by <a href="#">Photon</a> .
-----------------	--

**Returns**

[VoicelInfo](#) instance.

## 3.167.3 Property Documentation

### 3.167.3.1 Bitrate

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

Target bitrate (in bits/second).

### 3.167.3.2 Channels

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

Number of channels.

### 3.167.3.3 FPS

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

Video frames per second

### 3.167.3.4 FrameDurationSamples

```
int FrameDurationSamples [get]
```

Uncompressed frame (data packet) size in samples.

### 3.167.3.5 FrameDurationUs

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

Uncompressed frame (audio packet) size in microseconds.

### 3.167.3.6 FrameSize

```
int FrameSize [get]
```

Uncompressed frame (data packet) array size.

### 3.167.3.7 Height

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

Video height

### 3.167.3.8 KeyFrameInt

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

Video keyframe interval in frames

### 3.167.3.9 SamplingRate

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

Audio sampling rate (frequency, in Hz).

### 3.167.3.10 UserData

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

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

### 3.167.3.11 Width

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

Video width.

## 3.168 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 data.*
- void [Dispose](#) ()

## Properties

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

### 3.168.1 Detailed Description

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

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

### 3.168.2 Constructor & Destructor Documentation

#### 3.168.2.1 `VoiceLevelDetectCalibrate()`

```
VoiceLevelDetectCalibrate (
    int samplingRate,
    int channels )
```

Create new [VoiceLevelDetectCalibrate](#) instance

Parameters

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

### 3.168.3 Member Function Documentation

#### 3.168.3.1 `Calibrate()`

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

Start calibration

Parameters

<code>durationMs</code>	Duration of the calibration procedure (in milliseconds).
<code>onCalibrated</code>	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.

### 3.168.3.2 Process()

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

Process a frame of data.

#### Parameters

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

#### Returns

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

Implements [IProcessor< T >](#).

## 3.168.4 Property Documentation

### 3.168.4.1 LevelMeter

```
ILevelMeter LevelMeter [get]
```

The [LevelMeter](#) in use.

### 3.168.4.2 VoiceDetector

```
IVoiceDetector VoiceDetector [get]
```

The [VoiceDetector](#) in use

## 3.169 VoiceLogger Class Reference

Inherits MonoBehaviour.

### Static Public Member Functions

- static [VoiceLogger FindLogger](#) (GameObject gameObject)
- static [VoiceLogger CreateRootLogger](#) ()

## Public Attributes

- LogLevel **LogLevel** = LogLevel.Warning

## 3.170 VoiceNetworkObject Class Reference

Inherits NetworkBehaviour.

### Public Member Functions

- override void **Spawned** ()
- override void **Despawned** (NetworkRunner runner, bool hasState)

### Public Attributes

- VoiceLogger VoiceLogger** => voiceComponentImpl.VoiceLogger
- bool **IsSpeaking** => this.SpeakerInUse != null && this.SpeakerInUse.isPlaying  
*If true, this VoiceNetworkObject has a Speaker that is currently playing received audio frames from remote audio source*
- bool **IsRecording** => this.RecorderInUse != null && this.RecorderInUse.IsCurrentlyTransmitting  
*If true, this VoiceNetworkObject has a Recorder that is currently transmitting audio stream from local audio source*
- bool **IsLocal** => Runner.Topology == SimulationConfig.Topologies.Shared ? this.Object.HasStateAuthority : this.Object.HasInputAuthority

### Protected Attributes

- Voice.ILogger Logger** => voiceComponentImpl.Logger

### Properties

- Recorder RecorderInUse** [get]  
*The Recorder component currently used by this VoiceNetworkObject*
- Speaker SpeakerInUse** [get]  
*The Speaker component currently used by this VoiceNetworkObject*

## 3.170.1 Member Data Documentation

### 3.170.1.1 IsRecording

```
bool IsRecording => this.RecorderInUse != null && this.RecorderInUse.IsCurrentlyTransmitting
```

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

### 3.170.1.2 IsSpeaking

```
bool IsSpeaking => this.SpeakerInUse != null && this.SpeakerInUse.isPlaying
```

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

## 3.170.2 Property Documentation

### 3.170.2.1 RecorderInUse

```
Recorder RecorderInUse [get]
```

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

### 3.170.2.2 SpeakerInUse

```
Speaker SpeakerInUse [get]
```

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

## 3.171 AudioUtil.WaveformAudioPusher< T > Class Template Reference

[IAudioPusher](#) that provides the given waveform.

Inherits [AudioUtil.GeneratorPusher< T >](#).

### Public Member Functions

- **WaveformAudioPusher** (int bufSizeMs=100, int samplingRate=48000, int channels=1)

### Protected Member Functions

- override int **Gen** (T[] buf, long timeSamples)

### Properties

- T[] **Waveform** [set]

## Additional Inherited Members

### 3.171.1 Detailed Description

[IAudioPusher](#) that provides the given waveform.

## 3.172 **AudioUtil.WaveformAudioReader< T >** Class Template Reference

[IAudioReader](#) that provides the given waveform.

Inherits [AudioUtil.GeneratorReader< T >](#).

### Public Member Functions

- **WaveformAudioReader** (`Func< double > clockSec=null, int samplingRate=48000, int channels=1`)

### Protected Member Functions

- override int **Gen** (`T[] buf, long timeSamples`)

### Properties

- `T[] Waveform [set]`

### 3.172.1 Detailed Description

[IAudioReader](#) that provides the given waveform.

## 3.173 **WaveWriter** Class Reference

Inherits [IDisposable](#).

### Public Member Functions

- **WaveWriter** (`string fileName, int sampleRate, int bits, int channels`)
- **WaveWriter** (`Stream stream, int sampleRate, int bitsPerSample, int channels`)
- void **Dispose** ()
- void **WriteSample** (`float sample`)
- void **WriteSamples** (`float[] samples, int offset, int count`)
- void **Write** (`byte[] buffer, int offset, int count`)
- void **Write** (`byte value`)
- void **Write** (`short value`)
- void **Write** (`int value`)
- void **Write** (`float value`)

## Protected Member Functions

- virtual void **Dispose** (bool disposing)

# 3.174 WebRtcAudioDsp Class Reference

Inherits [VoiceComponent](#).

## Public Member Functions

- void **AdjustVoiceInfo** (ref [VoiceInfo](#) voiceInfo, ref [AudioSampleType](#) st)

## Public Attributes

- bool **IsSupported**

## Protected Member Functions

- override void **Awake** ()

## Properties

- bool **AEC** [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]
- int **AgcTargetLevel** [get, set]
- bool **VAD** [get, set]

## Additional Inherited Members

### 3.174.1 Member Data Documentation

#### 3.174.1.1 IsSupported

bool **IsSupported**

**Initial value:**

=>  
    true

## 3.175 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)

## 3.176 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

### 3.177 WindowsAudioInPusher Class Reference

Inherits [IAudioPusher< short >](#).

#### Public Member Functions

- **WindowsAudioInPusher** (int deviceID, [ILogger](#) logger)
- void **SetCallback** (Action< short[]> callback, [ObjectFactory](#)< short[], int > bufferFactory, int optimalFrameSize)
- void **Dispose** ()

#### Properties

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



# Index

AccumAvgPeakAmp  
    AudioUtil.ILevelMeter, 66

ActivityDelayMs  
    AudioUtil.IVoiceDetector, 73  
    AudioUtil.VoiceDetector< T >, 151

AddPostProcessor  
    LocalVoiceFramed< T >, 90

AddPreProcessor  
    LocalVoiceFramed< T >, 91

AddSpeaker  
    VoiceConnection, 142

AllowBluetooth  
    Photon.Voice.IOS, 9

Ambient  
    Photon.Voice.IOS, 7

AndroidAudioInAEC, 13

AndroidAudioInParameters, 13

ArrayPool< T >, 14

ArrayPoolSet< T >, 14

ArraySegmentPool< T >, 15

AudioChangesHandler, 15  
    HandleDeviceChange, 16  
    HandleDeviceChangeAndroid, 16  
    HandleDeviceChangeiOS, 16

AudioClip  
    Recorder, 110

AudioClipWrapper, 16

AudioDesc, 17

AudioInChangeNotifier, 17, 18  
    Dispose, 17, 18  
    Error, 18, 19

AudioInChangeNotifierNotSupportedException, 19

AudioInEnumerator, 19–21  
    Dispose, 20, 22  
    Refresh, 20, 22

AudioInPusher, 22, 23  
    Channels, 23

AudioInReader, 24

AudioInReader< T >, 24  
    Read, 25

AudioOpus  
    Photon.Voice, 6

AudioOut< T >, 25

AudioOutCapture, 26

AudioOutDelayControl, 26

AudioOutDelayControl.PlayDelayConfig, 103  
    Default, 104

AudioOutDummy< T >, 26

AudioOutEvent< T >, 27

AudioProcessing  
    Photon.Voice.IOS, 8

AudioSampleType  
    Photon.Voice, 6

AudioSessionCategory  
    Photon.Voice.IOS, 7

AudioSessionCategoryOption  
    Photon.Voice.IOS, 8

AudioSessionMode  
    Photon.Voice.IOS, 9

AudioSessionParameters, 27

AudioSessionParametersPresets, 28  
    Game, 28  
    VoIP, 28

AudioSyncBuffer< T >, 28

AudioUtil, 29  
    Convert, 30, 31  
    ForceToStereo< T >, 31  
    Resample< T >, 31  
    ResampleAndConvert, 32

AudioUtil.GeneratorPusher< T >, 54  
    SetCallback, 55

AudioUtil.GeneratorReader< T >, 55  
    Read, 56

AudioUtil.ILevelMeter, 65  
    AccumAvgPeakAmp, 66  
    CurrentAvgAmp, 66  
    CurrentPeakAmp, 66  
    ResetAccumAvgPeakAmp, 65

AudioUtil.IVoiceDetector, 72  
    ActivityDelayMs, 73  
    Detected, 73  
    DetectedTime, 73  
    On, 73  
    OnDetected, 74  
    Threshold, 74

AudioUtil.LevelMeter< T >, 74  
    Process, 75  
    ResetAccumAvgPeakAmp, 76

AudioUtil.LevelMeterDummy, 76  
    ResetAccumAvgPeakAmp, 76

AudioUtil.LevelMeterFloat, 77  
    LevelMeterFloat, 77

AudioUtil.LevelMeterShort, 77  
    LevelMeterShort, 78

AudioUtil.Resampler< T >, 118  
    Process, 119  
    Resampler, 119

AudioUtil.TempoUp< T >, 124

**AudioUtil.ToneAudioPusher< T >**, 124  
 ToneAudioPusher, 125  
**AudioUtil.ToneAudioReader< T >**, 125  
 ToneAudioReader, 126  
**AudioUtil.VoiceDetector< T >**, 150  
 ActivityDelayMs, 151  
 Detected, 152  
 DetectedTime, 152  
 On, 152  
 OnDetected, 152  
 Process, 151  
 Threshold, 152  
**AudioUtil.VoiceDetectorCalibration< T >**, 153  
 Calibrate, 154  
 Process, 154  
 VoiceDetectorCalibration, 153  
**AudioUtil.VoiceDetectorDummy**, 155  
**AudioUtil.VoiceDetectorFloat**, 155  
 VoiceDetectorFloat, 155  
**AudioUtil.VoiceDetectorShort**, 156  
 VoiceDetectorShort, 156  
**AudioUtil.VoiceLevelDetectCalibrate< T >**, 163  
 Calibrate, 164  
 LevelMeter, 165  
 Process, 165  
 VoiceDetector, 165  
 VoiceLevelDetectCalibrate, 164  
**AudioUtil.WaveformAudioPusher< T >**, 167  
**AudioUtil.WaveformAudioReader< T >**, 168  
**AutoConnectAndJoin**  
 VoiceFollowClient, 158  
  
**BestRegionSummaryInPreferences**  
 VoiceConnection, 144  
**Bitrate**  
 Recorder, 110  
 VoicelInfo, 161  
**BufferFactory**  
 LocalVoiceFramed< T >, 92  
**BufferReaderPushAdapterAsyncPool**  
 BufferReaderPushAdapterAsyncPool< T >, 33  
**BufferReaderPushAdapterAsyncPool< T >**, 33  
 BufferReaderPushAdapterAsyncPool, 33  
 Service, 33  
**BufferReaderPushAdapterAsyncPoolFloatToShort**, 34  
 BufferReaderPushAdapterAsyncPoolFloatToShort,  
     34  
 Service, 35  
**BufferReaderPushAdapterAsyncPoolShortToFloat**, 35  
 BufferReaderPushAdapterAsyncPoolShortToFloat,  
     36  
 Service, 36  
**BufferReaderPushAdapterBase**  
 BufferReaderPushAdapterBase< T >, 37  
**BufferReaderPushAdapterBase< T >**, 36  
 BufferReaderPushAdapterBase, 37  
 Dispose, 37  
 Service, 37  
**ByteStreamDecoder**, 38  
  
 Input, 38  
 Open, 38  
**ByteStreamEncoder**, 39  
  
**Calibrate**  
 AudioUtil.VoiceDetectorCalibration< T >, 154  
 AudioUtil.VoiceLevelDetectCalibrate< T >, 164  
**CaptureDevice**, 39  
 CaptureSource, 41  
 CleanUpAsync, 40  
 SelectPreferredCameraStreamSettingAsync, 40  
 StartRecordingAsync, 40  
 StopRecordingAsync, 41  
**CaptureSource**  
 CaptureDevice, 41  
**ChannelAudio**  
 VoiceConnection, 144  
**ChannelId**  
 RemoteVoicelInfo, 116  
**Channels**  
 AudioInPusher, 23  
 IAudioDesc, 56  
 VoicelInfo, 161  
**ChannelVideo**  
 VoiceConnection, 144  
**CleanUpAsync**  
 CaptureDevice, 40  
**ClearProcessors**  
 LocalVoiceFramed< T >, 91  
**ClientState**  
 VoiceConnection, 144  
**Code**  
 VoiceEvent, 157  
**Codec**  
 Photon.Voice, 6  
**ConnectAndJoin**, 41  
**ConnectAndJoinRoom**  
 VoiceFollowClient, 158  
**ConnectUsingSettings**  
 UnityVoiceClient, 128  
 VoiceConnection, 143  
**Convert**  
 AudioUtil, 30, 31  
**CreateAudio**  
 VoicelInfo, 160  
**CreateAudioOpus**  
 VoicelInfo, 160  
**CreateLocalVoice**  
 VoiceClient, 134  
**CreateLocalVoiceAudioFromSource**  
 VoiceClient, 135  
**CreateLocalVoiceVideo**  
 VoiceClient, 135  
**CreateVideo**  
 VoicelInfo, 161  
**CurrentAvgAmp**  
 AudioUtil.ILevelMeter, 66  
**CurrentPeakAmp**  
 AudioUtil.ILevelMeter, 66

DebugEchoMode  
    LocalVoice, 83  
    Recorder, 110  
    VoiceCreateOptions, 147  
DebugLostPercent  
    VoiceClient, 137  
Decoder  
    RemoteVoiceOptions, 118  
DecoderConfigFrame, 44  
    TryConfigure, 45  
Default  
    AudioOutDelayControl.PlayDelayConfig, 104  
    Photon.Voice.IOS, 9  
    VoiceClient.CreateOptions, 42  
DefaultToSpeaker  
    Photon.Voice.IOS, 9  
DequeueOutput  
    IEncoder, 63  
Detected  
    AudioUtil.IVoiceDetector, 73  
    AudioUtil.VoiceDetector< T >, 152  
DetectedTime  
    AudioUtil.IVoiceDetector, 73  
    AudioUtil.VoiceDetector< T >, 152  
DeviceEnumerator, 45  
DeviceEnumeratorBase, 46  
DeviceFeatures, 46  
DeviceInfo, 47  
DisableVad  
    VoiceDebugScript, 149  
Disconnect  
    VoiceFollowClient, 158  
Dispose  
    AudioInChangeNotifier, 17, 18  
    AudioInEnumerator, 20, 22  
    BufferReaderPushAdapterBase< T >, 37  
    LoadBalancingTransport, 80  
    LocalVoiceFramed< T >, 91  
    ObjectPool< TType, TInfo >, 97  
DuckOthers  
    Photon.Voice.IOS, 8  
Dummy  
    LocalVoiceAudioDummy, 88  
Encoder  
    VoiceCreateOptions, 147  
Encrypt  
    LocalVoice, 83  
    Recorder, 111  
    VoiceCreateOptions, 147  
EndOfStream  
    IEncoder, 63  
Error  
    AudioInChangeNotifier, 18, 19  
    IAudioDesc, 57  
    IDecoder, 61  
    IEncoder, 63  
EventBufSize  
    VoiceCreateOptions, 147  
EventsLost  
    VoiceClient, 137  
FEC  
    LocalVoice, 83  
    VoiceCreateOptions, 147  
Flip, 49  
FMODRecorderSetup, 50  
ForceRecordingAndTransmission  
    VoiceDebugScript, 149  
ForceToStereo< T >  
    AudioUtil, 31  
FPS  
    VoicelInfo, 162  
Fragment  
    LocalVoice, 83  
    VoiceCreateOptions, 147  
Frame  
    Framer< T >, 52  
    FramerResampler< T >, 52  
FrameBuffer, 50  
FrameDuration  
    Recorder, 111  
FrameDurationSamples  
    VoicelInfo, 162  
FrameDurationUs  
    VoicelInfo, 162  
FrameOut< T >, 51  
Framer  
    Framer< T >, 51  
Framer< T >, 51  
    Frame, 52  
    Framer, 51  
FramerResampler< T >, 52  
    Frame, 52  
FramesFragPart  
    VoiceClient, 137  
FrameSize  
    VoicelInfo, 162  
FramesLate  
    VoiceClient, 137  
FramesLost  
    VoiceClient, 137  
FramesLostPercent  
    VoiceConnection, 145  
FramesLostPerSecond  
    VoiceConnection, 145  
FramesReceived  
    VoiceClient, 137  
FramesReceivedFEC  
    VoiceClient, 138  
FramesReceivedFragmented  
    VoiceClient, 138  
FramesReceivedFragments  
    VoiceClient, 138  
FramesReceivedPerSecond  
    VoiceConnection, 145  
FramesRecovered  
    VoiceClient, 138

FramesSent  
   LocalVoice, 83  
   VoiceClient, 138

FramesSentBytes  
   LocalVoice, 83  
   VoiceClient, 138

FramesSentFragmented  
   LocalVoice, 84

FramesSentFragments  
   LocalVoice, 84

FramesTryFEC  
   VoiceClient, 139

Free  
   ObjectPool< TType, TInfo >, 97, 98

FusionVoiceClient, 53  
   UseFusionAppSettings, 53  
   UseFusionAuthValues, 54

Game  
   AudioSessionParametersPresets, 28

GetPlatformAPI< I >  
   IEncoder, 63

HandleDeviceChange  
   AudioChangesHandler, 16

HandleDeviceChangeAndroid  
   AudioChangesHandler, 16

HandleDeviceChangeiOS  
   AudioChangesHandler, 16

Height  
   VoiceInfo, 162

IAudioDesc, 56  
   Channels, 56  
   Error, 57  
   SamplingRate, 57

IAudioInChangeNotifier, 57

IAudioOut< T >, 57

IAudioPusher< T >, 58  
   SetCallback, 58

IAudioReader< T >, 59

IDataReader< T >, 59  
   Read, 59

IDecoder, 60  
   Error, 61  
   Input, 60  
   Open, 60

IDecoderDirect< B >, 61  
   Output, 61

IDeviceEnumerator, 62

IEncoder, 62  
   DequeueOutput, 63  
   EndOfStream, 63  
   Error, 63  
   GetPlatformAPI< I >, 63  
   Output, 63

IEncoderDirect< B >, 64  
   Input, 64

IEncoderDirectImage, 64

ImageFormat, 65

ILocalVoiceAudio, 66  
   LevelMeter, 67  
   VoiceDetector, 67  
   VoiceDetectorCalibrate, 67  
   VoiceDetectorCalibrating, 68

ILogger, 68

ImageBufferInfo, 68

ImageBufferInfo.StrideSet, 123

ImageBufferNative, 69

ImageBufferNative.PlaneSet, 103

ImageBufferNativeAlloc, 69

ImageBufferNativeGCHandleBytes, 69

ImageBufferNativeGCHandleSinglePlane, 70

ImageBufferNativePool< T >, 70

ImageFormat  
   IEncoderDirectImage, 65

IncreaseLogLevels  
   VoiceDebugScript, 149

Info  
   LocalVoice, 84  
   ObjectPool< TType, TInfo >, 99  
   RemoteVoiceInfo, 116

Input  
   ByteStreamDecoder, 38  
   IDecoder, 60  
   IEncoderDirect< B >, 64  
   OpusCodec.Decoder< T >, 43  
   RawCodec.Decoder< T >, 44

InputFactory  
   Recorder, 111

Instance  
   PunVoiceClient, 105

InstantiateSpeakerPrefab  
   VoiceConnection, 143

InterestGroup  
   LocalVoice, 84  
   Recorder, 111  
   VoiceCreateOptions, 148

IProcessor< T >, 71  
   Process, 71

IResettable, 71

IsCurrentlyTransmitting  
   LocalVoice, 84  
   Recorder, 111

IServiceable, 72  
   Service, 72

IsLinked  
   Speaker, 122

IsPlaying  
   Speaker, 122

IsRecording  
   PhotonVoiceView, 102  
   VoiceNetworkObject, 166

IsSpeaking  
   PhotonVoiceView, 102  
   VoiceNetworkObject, 166

IsSupported

WebRtcAudioDsp, 169  
    IVoiceTransport, 74

KeyFrameInt  
    VoiceInfo, 162

Lag  
    Speaker, 122

LevelMeter  
    AudioUtil.VoiceLevelDetectCalibrate< T >, 165  
    ILocalVoiceAudio, 67  
    Recorder, 111

LevelMeterFloat  
    AudioUtil.LevelMeterFloat, 77

LevelMeterShort  
    AudioUtil.LevelMeterShort, 78

LoadBalancingTransport, 78  
    Dispose, 80  
    LoadBalancingTransport, 79  
    Service, 80  
    VoiceClient, 80

LoadBalancingTransport2, 80

LocalDebug  
    VoiceDebugScript, 149

LocalUserServiceable  
    LocalVoice, 84

LocalVoice, 81  
    DebugEchoMode, 83  
    Encrypt, 83  
    FEC, 83  
    Fragment, 83  
    FramesSent, 83  
    FramesSentBytes, 83  
    FramesSentFragmented, 84  
    FramesSentFragments, 84  
    Info, 84  
    InterestGroup, 84  
    IsCurrentlyTransmitting, 84  
    LocalUserServiceable, 84  
    Reliable, 85  
    RemoveSelf, 82  
    SendSpacingProfileMax, 85  
    TargetPlayers, 85  
    TransmitEnabled, 85

LocalVoiceAudio< T >, 85  
    VoiceDetectorCalibrate, 86  
    VoiceDetectorCalibrating, 87

LocalVoiceAudioDummy, 87  
    Dummy, 88  
    VoiceDetectorCalibrate, 87

LocalVoiceAudioFloat, 88

LocalVoiceAudioShort, 88

LocalVoiceFramed< T >, 89  
    AddPostProcessor, 90  
    AddPreProcessor, 91  
    BufferFactory, 92  
    ClearProcessors, 91  
    Dispose, 91  
    PushData, 91

    PushDataAsync, 91  
    PushDataAsyncReady, 92  
    RemoveProcessor, 92

LocalVoices  
    VoiceClient, 139

LocalVoicesInChannel  
    VoiceClient, 136

Logger, 92

LoopAudioClip  
    Recorder, 112

Measurement  
    Photon.Voice.IOS, 9

MicAmplifier, 93

MicAmplifierFloat, 93

MicAmplifierShort, 93

MicrophonePermission, 94

MicrophoneType  
    Recorder, 112

MicWrapper, 94

MicWrapperPusher, 95

MicWrapperPusherOnAudioFilterRead, 95

MixWithOthers  
    Photon.Voice.IOS, 8

MonoPInvokeCallbackAttribute, 95

MoviePlayback  
    Photon.Voice.IOS, 9

MultiRoute  
    Photon.Voice.IOS, 8

New  
    ObjectPool< TType, TInfo >, 98

ObjectFactory< TType, TInfo >, 95

ObjectPool  
    ObjectPool< TType, TInfo >, 97  
    ObjectPool< TType, TInfo >, 96  
        Dispose, 97  
        Free, 97, 98  
        Info, 99  
        New, 98  
        ObjectPool, 97

On  
    AudioUtil.IVoiceDetector, 73  
    AudioUtil.VoiceDetector< T >, 152

OnDetected  
    AudioUtil.IVoiceDetector, 74  
    AudioUtil.VoiceDetector< T >, 152

OnRemoteVoiceInfoAction  
    VoiceClient, 139

OnRemoteVoiceRemoveAction  
    RemoteVoiceOptions, 118  
    Speaker, 122

Open  
    ByteStreamDecoder, 38  
    IDecoder, 60  
    OpusCodec.Decoder< T >, 43  
    RawCodec.Decoder< T >, 44

    OpusCodec, 99

OpusCodec.Decoder< T >, 42  
     Input, 43  
     Open, 43  
 OpusCodec.Encoder< T >, 47  
 OpusCodec.EncoderFloat, 48  
 OpusCodec.EncoderShort, 49  
 OpusCodec.Factory, 49  
 OpusCodec.Util, 131  
 Output  
     IDecoderDirect< B >, 61  
     IEncoder, 63

Photon, 3  
 Photon.Voice, 3  
     AudioOpus, 6  
     AudioSampleType, 6  
     Codec, 6  
 Photon.Voice.FMOD, 6  
 Photon.Voice.Fusion, 7  
 Photon.Voice.IOS, 7  
     AllowBluetooth, 9  
     Ambient, 7  
     AudioProcessing, 8  
     AudioSessionCategory, 7  
     AudioSessionCategoryOption, 8  
     AudioSessionMode, 9  
     Default, 9  
     DefaultToSpeaker, 9  
     DuckOthers, 8  
     Measurement, 9  
     MixWithOthers, 8  
     MoviePlayback, 9  
     MultiRoute, 8  
     PlayAndRecord, 8  
     Playback, 7  
     Record, 7  
     SoloAmbient, 7  
     VideoChat, 10  
     VideoRecording, 9  
     VoiceChat, 9  
 Photon.Voice.MacOS, 10  
 Photon.Voice.PUN, 10  
 Photon.Voice.PUN.UtilityScripts, 10  
 Photon.Voice.Unity, 11  
 Photon.Voice.Unity.FMOD, 11  
 Photon.Voice.Unity.UtilityScripts, 12  
 Photon.Voice.UWP, 12  
 Photon.Voice.Windows, 12  
 PhotonAppSettings, 99  
     ToString, 100  
     UseCloud, 100  
 PhotonVoiceCreatedParams, 100  
 PhotonVoiceLagSimulationGui, 101  
 PhotonVoiceStatsGui, 101  
 PhotonVoiceView, 101  
     IsRecording, 102  
     IsSpeaking, 102  
     RecorderInUse, 102  
     SpeakerInUse, 102

Platform, 103  
 PlayAndRecord  
     Photon.Voice.IOS, 8  
 Playback  
     Photon.Voice.IOS, 7  
 PlayDelay  
     Speaker, 122  
 PlayDelayConfig  
     Speaker, 122  
 PlayerId  
     RemoteVoiceInfo, 116  
 PrimaryRecorder  
     VoiceConnection, 145  
 Process  
     AudioUtil.LevelMeter< T >, 75  
     AudioUtil.Resampler< T >, 119  
     AudioUtil.VoiceDetector< T >, 151  
     AudioUtil.VoiceDetectorCalibration< T >, 154  
     AudioUtil.VoiceLevelDetectCalibrate< T >, 165  
     IProcessor< T >, 71  
 PunVoiceClient, 104  
     Instance, 105  
     UsePunAppSettings, 105  
     UsePunAuthValues, 105  
     VoiceRoomNameSuffix, 105  
 PushData  
     LocalVoiceFramed< T >, 91  
 PushDataAsync  
     LocalVoiceFramed< T >, 91  
 PushDataAsyncReady  
     LocalVoiceFramed< T >, 92  
 RawCodec, 106  
 RawCodec.Decoder< T >, 43  
     Input, 44  
     Open, 44  
 RawCodec.Encoder< T >, 48  
 RawCodec.ShortToFloat, 120  
 Read  
     AudioInReader< T >, 25  
     AudioUtil.GeneratorReader< T >, 56  
     IDataReader< T >, 59  
 Record  
     Photon.Voice.IOS, 7  
 Recorder, 106  
     AudioClip, 110  
     Bitrate, 110  
     DebugEchoMode, 110  
     Encrypt, 111  
     FrameDuration, 111  
     InputFactory, 111  
     InterestGroup, 111  
     IsCurrentlyTransmitting, 111  
     LevelMeter, 111  
     LoopAudioClip, 112  
     MicrophoneType, 112  
     RecordingEnabled, 112  
     RecordWhenJoined, 112  
     ReliableMode, 112

ResetLocalAudio, 108  
RestartRecording, 108  
SamplingRate, 112  
SetAndroidNativeMicrophoneSettings, 109  
SetIosAudioSessionParameters, 109  
SourceType, 113  
StopRecordingWhenPaused, 113  
TargetPlayers, 113  
TransmitEnabled, 113  
UseMicrophoneTypeFallback, 113  
UseOnAudioFilterRead, 113  
UserData, 114  
VoiceDetection, 114  
VoiceDetectionDelayMs, 114  
VoiceDetectionThreshold, 114  
VoiceDetector, 114  
VoiceDetectorCalibrate, 110  
VoiceDetectorCalibrating, 114  
RecorderInUse  
    PhotonVoiceView, 102  
    VoiceNetworkObject, 167  
RecorderPreset, 115  
RecorderPreset.DSP, 47  
RecordingEnabled  
    Recorder, 112  
RecordWhenJoined  
    Recorder, 112  
Refresh  
    AudioInEnumerator, 20, 22  
Reliable  
    LocalVoice, 85  
    VoiceCreateOptions, 148  
ReliableMode  
    Recorder, 112  
RemoteVoiceAdded  
    VoiceConnection, 146  
RemoteVoicelInfo, 115  
    ChannelId, 116  
    Info, 116  
    PlayerId, 116  
    Voiceld, 116  
RemoteVoicelInfoDelegate  
    VoiceClient, 136  
RemoteVoicelInfos  
    VoiceClient, 139  
RemoteVoiceLink, 116  
RemoteVoiceOptions, 117  
    Decoder, 118  
    OnRemoteVoiceRemoveAction, 118  
    SetOutput, 117  
RemoveLocalVoice  
    VoiceClient, 136  
RemoveProcessor  
    LocalVoiceFramed< T >, 92  
RemoveSelf  
    LocalVoice, 82  
Resample< T >  
    AudioUtil, 31  
    ResampleAndConvert  
        AudioUtil, 32  
    Resampler  
        AudioUtil.Resampler< T >, 119  
    ResetAccumAvgPeakAmp  
        AudioUtil.ILevelMeter, 65  
        AudioUtil.LevelMeter< T >, 76  
        AudioUtil.LevelMeterDummy, 76  
    ResetLocalAudio  
        Recorder, 108  
    RestartPlayback  
        Speaker, 121  
    RestartRecording  
        Recorder, 108  
    RoundTripTime  
        VoiceClient, 139  
    RoundTripTimeVariance  
        VoiceClient, 139  
SamplingRate  
    IAudioDesc, 57  
    Recorder, 112  
    VoicelInfo, 163  
SaveIncomingStreamToFile, 120  
SaveOutgoingStreamToFile, 120  
SelectPreferredCameraStreamSettingAsync  
    CaptureDevice, 40  
SendFrameParams, 120  
SendSpacingProfileMax  
    LocalVoice, 85  
Service  
    BufferReaderPushAdapterAsyncPool< T >, 33  
    BufferReaderPushAdapterAsyncPoolFloatToShort,  
        35  
    BufferReaderPushAdapterAsyncPoolShortToFloat,  
        36  
    BufferReaderPushAdapterBase< T >, 37  
    IServiceable, 72  
    LoadBalancingTransport, 80  
    VoiceClient, 136  
SetAndroidNativeMicrophoneSettings  
    Recorder, 109  
SetCallback  
    AudioUtil.GeneratorPusher< T >, 55  
    IAudioPusher< T >, 58  
SetIosAudioSessionParameters  
    Recorder, 109  
SetOutput  
    RemoteVoiceOptions, 117  
Settings  
    VoiceConnection, 144  
SoloAmbient  
    Photon.Voice.IOS, 7  
SourceType  
    Recorder, 113  
Speaker, 120  
    IsLinked, 122  
    IsPlaying, 122  
    Lag, 122

OnRemoteVoiceRemoveAction, 122  
PlayDelay, 122  
PlayDelayConfig, 122  
RestartPlayback, 121  
SpeakerAudioFilterRead, 123  
SpeakerFMOD, 123  
SpeakerInUse  
    PhotonVoiceView, 102  
    VoiceNetworkObject, 167  
SpeakerLinked  
    VoiceConnection, 146  
SpeakerPrefab  
    VoiceConnection, 145  
StartRecordingAsync  
    CaptureDevice, 40  
StopRecordingAsync  
    CaptureDevice, 41  
StopRecordingWhenPaused  
    Recorder, 113  
SuppressInfoDuplicateWarning  
    VoiceClient, 140

TargetPlayers  
    LocalVoice, 85  
    Recorder, 113  
    VoiceCreateOptions, 148  
TestAudioClip  
    VoiceDebugScript, 150  
TestTone, 124  
TestUsingAudioClip  
    VoiceDebugScript, 150  
Threshold  
    AudioUtil.IVoiceDetector, 74  
    AudioUtil.VoiceDetector< T >, 152  
TimerWorker, 124  
ToneAudioPusher  
    AudioUtil.ToneAudioPusher< T >, 125  
ToneAudioReader  
    AudioUtil.ToneAudioReader< T >, 126  
ToString  
    PhotonAppSettings, 100  
TransmitEnabled  
    LocalVoice, 85  
    Recorder, 113  
TryConfigure  
    DecoderConfigFrame, 45

UnityAudioOut, 126  
UnityLogger, 127  
UnityMicrophone, 127  
UnityVoiceClient, 127  
    ConnectUsingSettings, 128  
    UseVoiceAppSettings, 128  
UnsupportedCodecException, 129  
    UnsupportedCodecException, 129  
UnsupportedPlatformException, 129  
    UnsupportedPlatformException, 130  
UnsupportedSampleTypeException, 130  
    UnsupportedSampleTypeException, 131

UseCloud  
    PhotonAppSettings, 100  
UseFusionAppSettings  
    FusionVoiceClient, 53  
UseFusionAuthValues  
    FusionVoiceClient, 54  
UseMicrophoneTypeFallback  
    Recorder, 113  
UseOnAudioFilterRead  
    Recorder, 113  
UsePrimaryRecorder  
    VoiceConnection, 144  
UsePunAppSettings  
    PunVoiceClient, 105  
UsePunAuthValues  
    PunVoiceClient, 105  
UserData  
    Recorder, 114  
    VoicelInfo, 163  
UseVoiceAppSettings  
    UnityVoiceClient, 128  
Utility, 131

VideoChat  
    Photon.Voice.IOS, 10  
VideoInEnumerator, 131, 132  
VideoRecording  
    Photon.Voice.IOS, 9  
VoiceChat  
    Photon.Voice.IOS, 9  
VoiceClient, 132  
    CreateLocalVoice, 134  
    CreateLocalVoiceAudioFromSource, 135  
    CreateLocalVoiceVideo, 135  
    DebugLostPercent, 137  
    EventsLost, 137  
    FramesFragPart, 137  
    FramesLate, 137  
    FramesLost, 137  
    FramesReceived, 137  
    FramesReceivedFEC, 138  
    FramesReceivedFragmented, 138  
    FramesReceivedFragments, 138  
    FramesRecovered, 138  
    FramesSent, 138  
    FramesSentBytes, 138  
    FramesTryFEC, 139  
    LoadBalancingTransport, 80  
    LocalVoices, 139  
    LocalVoicesInChannel, 136  
    OnRemoteVoiceInfoAction, 139  
    RemoteVoiceInfoDelegate, 136  
    RemoteVoiceInfos, 139  
    RemoveLocalVoice, 136  
    RoundTripTime, 139  
    RoundTripTimeVariance, 139  
    Service, 136  
    SuppressInfoDuplicateWarning, 140  
    VoiceClient, 134

VoiceConnection, 145  
VoiceClient.CreateOptions, 42  
    Default, 42  
VoiceComponent, 140  
VoiceComponentImpl, 140  
VoiceConnection, 141  
    AddSpeaker, 142  
    BestRegionSummaryInPreferences, 144  
    ChannelAudio, 144  
    ChannelVideo, 144  
    ClientState, 144  
    ConnectUsingSettings, 143  
    FramesLostPercent, 145  
    FramesLostPerSecond, 145  
    FramesReceivedPerSecond, 145  
    InstantiateSpeakerPrefab, 143  
    PrimaryRecorder, 145  
    RemoteVoiceAdded, 146  
    Settings, 144  
    SpeakerLinked, 146  
    SpeakerPrefab, 145  
    UsePrimaryRecorder, 144  
    VoiceClient, 145  
VoiceCreateOptions, 146  
    DebugEchoMode, 147  
    Encoder, 147  
    Encrypt, 147  
    EventBufSize, 147  
    FEC, 147  
    Fragment, 147  
    InterestGroup, 148  
    Reliable, 148  
    TargetPlayers, 148  
VoiceDebugScript, 148  
    DisableVad, 149  
    ForceRecordingAndTransmission, 149  
    IncreaseLogLevels, 149  
    LocalDebug, 149  
    TestAudioClip, 150  
    TestUsingAudioClip, 150  
VoiceDetection  
    Recorder, 114  
VoiceDetectionDelayMs  
    Recorder, 114  
VoiceDetectionThreshold  
    Recorder, 114  
VoiceDetector  
    AudioUtil.VoiceLevelDetectCalibrate< T >, 165  
    ILocalVoiceAudio, 67  
    Recorder, 114  
VoiceDetectorCalibrate  
    ILocalVoiceAudio, 67  
    LocalVoiceAudio< T >, 86  
    LocalVoiceAudioDummy, 87  
    Recorder, 110  
VoiceDetectorCalibrating  
    ILocalVoiceAudio, 68  
    LocalVoiceAudio< T >, 87  
Recorder, 114  
VoiceDetectorCalibration  
    AudioUtil.VoiceDetectorCalibration< T >, 153  
VoiceDetectorFloat  
    AudioUtil.VoiceDetectorFloat, 155  
VoiceDetectorShort  
    AudioUtil.VoiceDetectorShort, 156  
VoiceEvent, 157  
    Code, 157  
VoiceFollowClient, 157  
    AutoConnectAndJoin, 158  
    ConnectAndJoinRoom, 158  
    Disconnect, 158  
Voiceld  
    RemoteVoicelInfo, 116  
VoicelInfo, 159  
    Bitrate, 161  
    Channels, 161  
    CreateAudio, 160  
    CreateAudioOpus, 160  
    CreateVideo, 161  
    FPS, 162  
    FrameDurationSamples, 162  
    FrameDurationUs, 162  
    FrameSize, 162  
    Height, 162  
    KeyFrameInt, 162  
    SamplingRate, 163  
    UserData, 163  
    Width, 163  
VoiceLevelDetectCalibrate  
    AudioUtil.VoiceLevelDetectCalibrate< T >, 164  
VoiceLogger, 165  
VoiceNetworkObject, 166  
    IsRecording, 166  
    IsSpeaking, 166  
    RecorderInUse, 167  
    SpeakerInUse, 167  
VoiceRoomNameSuffix  
    PunVoiceClient, 105  
VoIP  
    AudioSessionParametersPresets, 28  
WaveWriter, 168  
WebRtcAudioDsp, 169  
    IsSupported, 169  
WebRTCAudioLib, 170  
WebRTCAudioProcessor, 170  
Width  
    VoicelInfo, 163  
WindowsAudioInPusher, 171