

ObjectAL

Generated by Doxygen 1.8.3.1

Tue Apr 16 2013 21:29:24

Contents

1	ObjectAL for iPhone	1
1.1	Contents	1
1.2	Introduction	1
1.3	ObjectAL and OpenAL	2
1.4	Adding ObjectAL to your project	3
1.4.1	Installing the ObjectAL Documentation into XCode	3
1.5	Compile-Time Configuration	3
1.6	Audio Formats	4
1.6.1	OALAudioTrack Supported Formats	4
1.6.2	OpenAL Supported Formats	4
1.7	Choosing Playback Types	4
1.8	Using OALSimpleAudio	5
1.9	Using the OpenAL Objects and OALAudioTrack	6
1.10	Other Examples	8
1.11	iOS Issues that can impede playback	8
1.11.1	MPMoviePlayerController on iOS 3.x	8
1.11.2	MPMusicPlayerController on iOS 4.0	9
1.12	Simulator Issues	9
1.12.1	Simulator Limitations	9
1.12.2	Error Codes on the Simulator	9
1.12.3	Playback Issues	9
2	Hierarchical Index	11
2.1	Class Hierarchy	11
3	Class Index	13
3.1	Class List	13
4	Class Documentation	15
4.1	ALBuffer Class Reference	15
4.1.1	Detailed Description	16
4.1.2	Method Documentation	16

4.1.2.1	bufferWithName:data:size:format:frequency:	16
4.1.2.2	initWithName:data:size:format:frequency:	16
4.1.2.3	sliceWithName:offset:size:	17
4.1.3	Member Data Documentation	17
4.1.3.1	bufferData	17
4.1.3.2	parentBuffer	17
4.1.4	Property Documentation	17
4.1.4.1	bits	17
4.1.4.2	bufferId	17
4.1.4.3	channels	17
4.1.4.4	device	18
4.1.4.5	duration	18
4.1.4.6	format	18
4.1.4.7	freeDataOnDestroy	18
4.1.4.8	frequency	18
4.1.4.9	name	18
4.1.4.10	size	18
4.2	ALCaptureDevice Class Reference	18
4.2.1	Detailed Description	19
4.2.2	Method Documentation	19
4.2.2.1	deviceWithDeviceSpecifier:frequency:format:bufferSize:	19
4.2.2.2	getProcAddress:	20
4.2.2.3	initWithDeviceSpecifier:frequency:format:bufferSize:	20
4.2.2.4	isExtensionPresent:	20
4.2.2.5	moveSamples:toBuffer:	20
4.2.2.6	startCapture	21
4.2.2.7	stopCapture	21
4.2.3	Property Documentation	21
4.2.3.1	captureSamples	21
4.2.3.2	device	21
4.2.3.3	extensions	21
4.2.3.4	majorVersion	21
4.2.3.5	minorVersion	21
4.3	ALChannelSource Class Reference	21
4.3.1	Detailed Description	24
4.3.2	Method Documentation	24
4.3.2.1	addChannel:	24
4.3.2.2	addSource:	25
4.3.2.3	channelWithSources:	25
4.3.2.4	clearUnusedBuffers	25

4.3.2.5	initWithSources:	25
4.3.2.6	removeBuffersNamed:	25
4.3.2.7	removeSource:	26
4.3.2.8	resetToDefault	26
4.3.2.9	setDefaultFromSource:	26
4.3.2.10	splitChannelWithSources:	26
4.3.3	Member Data Documentation	26
4.3.3.1	currentFadeCallbackCount	26
4.3.3.2	currentPanCallbackCount	26
4.3.3.3	currentPitchCallbackCount	26
4.3.3.4	defaultConeInnerAngle	26
4.3.3.5	defaultConeOuterAngle	27
4.3.3.6	defaultConeOuterGain	27
4.3.3.7	defaultDirection	27
4.3.3.8	defaultGain	27
4.3.3.9	defaultLooping	27
4.3.3.10	defaultMaxDistance	27
4.3.3.11	defaultMaxGain	27
4.3.3.12	defaultMinGain	27
4.3.3.13	defaultPitch	27
4.3.3.14	defaultPosition	27
4.3.3.15	defaultReferenceDistance	27
4.3.3.16	defaultReverbObstruction	27
4.3.3.17	defaultReverbOcclusion	28
4.3.3.18	defaultReverbSendLevel	28
4.3.3.19	defaultRolloffFactor	28
4.3.3.20	defaultsInitialized	28
4.3.3.21	defaultSourceRelative	28
4.3.3.22	defaultSourceType	28
4.3.3.23	defaultVelocity	28
4.3.3.24	expectedFadeCallbackCount	28
4.3.3.25	expectedPanCallbackCount	28
4.3.3.26	expectedPitchCallbackCount	28
4.3.3.27	fadeCompleteSelector	28
4.3.3.28	fadeCompleteTarget	28
4.3.3.29	panCompleteSelector	29
4.3.3.30	panCompleteTarget	29
4.3.3.31	pitchCompleteSelector	29
4.3.3.32	pitchCompleteTarget	29
4.3.4	Property Documentation	29

4.3.4.1	context	29
4.3.4.2	reservedSources	29
4.3.4.3	sourcePool	29
4.4	ALContext Class Reference	29
4.4.1	Detailed Description	31
4.4.2	Method Documentation	31
4.4.2.1	clearBuffers	31
4.4.2.2	contextOnDevice:attributes:	31
4.4.2.3	contextOnDevice:outputFrequency:refreshIntervals:synchronousContext:mono-Sources:stereoSources:	31
4.4.2.4	ensureContextIsCurrent	32
4.4.2.5	getProcAddress:	32
4.4.2.6	initOnDevice:attributes:	32
4.4.2.7	initOnDevice:outputFrequency:refreshIntervals:synchronousContext:mono-Sources:stereoSources:	33
4.4.2.8	isExtensionPresent:	33
4.4.2.9	process	33
4.4.2.10	stopAllSounds	33
4.4.3	Member Data Documentation	33
4.4.3.1	attributes	33
4.4.3.2	sources	34
4.4.3.3	suspendHandler	34
4.4.4	Property Documentation	34
4.4.4.1	alVersion	34
4.4.4.2	attributes	34
4.4.4.3	context	34
4.4.4.4	device	34
4.4.4.5	distanceModel	34
4.4.4.6	dopplerFactor	34
4.4.4.7	extensions	34
4.4.4.8	listener	35
4.4.4.9	renderer	35
4.4.4.10	sources	35
4.4.4.11	speedOfSound	35
4.4.4.12	vendor	35
4.5	ALContext() Category Reference	35
4.6	ALDevice Class Reference	35
4.6.1	Detailed Description	36
4.6.2	Method Documentation	37
4.6.2.1	clearBuffers	37

4.6.2.2	deviceWithDeviceSpecifier:	37
4.6.2.3	getProcAddress:	37
4.6.2.4	initWithDeviceSpecifier:	37
4.6.2.5	isExtensionPresent:	37
4.6.3	Member Data Documentation	38
4.6.3.1	contexts	38
4.6.3.2	suspendHandler	38
4.6.4	Property Documentation	38
4.6.4.1	contexts	38
4.6.4.2	device	38
4.6.4.3	extensions	38
4.6.4.4	majorVersion	38
4.6.4.5	minorVersion	38
4.7	ALListener Class Reference	38
4.7.1	Detailed Description	39
4.7.2	Member Data Documentation	39
4.7.2.1	suspendHandler	39
4.7.3	Property Documentation	39
4.7.3.1	context	39
4.7.3.2	gain	40
4.7.3.3	globalReverbLevel	40
4.7.3.4	muted	40
4.7.3.5	orientation	40
4.7.3.6	position	40
4.7.3.7	reverbEQBandwidth	40
4.7.3.8	reverbEQFrequency	40
4.7.3.9	reverbEQGain	40
4.7.3.10	reverbOn	40
4.7.3.11	reverbRoomType	41
4.7.3.12	velocity	41
4.8	ALOrientation Struct Reference	41
4.8.1	Detailed Description	41
4.8.2	Member Data Documentation	41
4.8.2.1	at	41
4.8.2.2	up	41
4.9	ALPoint Struct Reference	42
4.9.1	Detailed Description	42
4.9.2	Member Data Documentation	42
4.9.2.1	x	42
4.9.2.2	y	42

4.9.2.3	z	42
4.10	<ALSoundSource> Protocol Reference	42
4.10.1	Detailed Description	44
4.10.2	Method Documentation	44
4.10.2.1	clear	44
4.10.2.2	fadeTo:duration:target:selector:	45
4.10.2.3	panTo:duration:target:selector:	45
4.10.2.4	pitchTo:duration:target:selector:	45
4.10.2.5	play:	45
4.10.2.6	play:gain:pitch:pan:loop:	45
4.10.2.7	play:loop:	46
4.10.2.8	rewind	46
4.10.2.9	stop	46
4.10.2.10	stopActions	46
4.10.2.11	stopFade	46
4.10.2.12	stopPan	46
4.10.2.13	stopPitch	46
4.10.3	Property Documentation	47
4.10.3.1	coneInnerAngle	47
4.10.3.2	coneOuterAngle	47
4.10.3.3	coneOuterGain	47
4.10.3.4	direction	47
4.10.3.5	gain	47
4.10.3.6	interruptible	47
4.10.3.7	looping	47
4.10.3.8	maxDistance	47
4.10.3.9	maxGain	47
4.10.3.10	minGain	47
4.10.3.11	muted	47
4.10.3.12	pan	48
4.10.3.13	paused	48
4.10.3.14	pitch	48
4.10.3.15	playing	48
4.10.3.16	position	48
4.10.3.17	referenceDistance	48
4.10.3.18	reverbObstruction	48
4.10.3.19	reverbOcclusion	48
4.10.3.20	reverbSendLevel	48
4.10.3.21	rolloffFactor	48
4.10.3.22	sourceRelative	48

4.10.3.23 sourceType	49
4.10.3.24 velocity	49
4.10.3.25 volume	49
4.11 ALSoundSourcePool Class Reference	49
4.11.1 Detailed Description	50
4.11.2 Method Documentation	50
4.11.2.1 addSource:	50
4.11.2.2 getFreeSource:	50
4.11.2.3 pool	50
4.11.2.4 removeSource:	50
4.11.3 Member Data Documentation	50
4.11.3.1 sources	51
4.11.4 Property Documentation	51
4.11.4.1 sources	51
4.12 ALSoundSourcePool(Private) Category Reference	51
4.12.1 Detailed Description	51
4.12.2 Method Documentation	51
4.12.2.1 moveToHead:	51
4.13 ALSource Class Reference	51
4.13.1 Detailed Description	53
4.13.2 Method Documentation	53
4.13.2.1 initOnContext:	53
4.13.2.2 play	53
4.13.2.3 queueBuffer:	54
4.13.2.4 queueBuffer:repeats:	54
4.13.2.5 queueBuffers:	54
4.13.2.6 queueBuffers:repeats:	54
4.13.2.7 registerNotification:callback:userData:	55
4.13.2.8 source	55
4.13.2.9 sourceOnContext:	55
4.13.2.10 unqueueBuffer:	55
4.13.2.11 unqueueBuffers:	56
4.13.2.12 unregisterAllNotifications	56
4.13.2.13 unregisterNotification:	56
4.13.3 Member Data Documentation	56
4.13.3.1 abortPlaybackResume	56
4.13.3.2 gainAction	56
4.13.3.3 panAction	56
4.13.3.4 pitchAction	56
4.13.3.5 shadowState	56

4.13.3.6	suspendHandler	57
4.13.4	Property Documentation	57
4.13.4.1	buffer	57
4.13.4.2	buffersProcessed	57
4.13.4.3	buffersQueued	57
4.13.4.4	context	57
4.13.4.5	offsetInBytes	57
4.13.4.6	offsetInSamples	57
4.13.4.7	offsetInSeconds	57
4.13.4.8	sourceId	57
4.13.4.9	state	57
4.14	ALVector Struct Reference	58
4.14.1	Detailed Description	58
4.14.2	Member Data Documentation	58
4.14.2.1	x	58
4.14.2.2	y	58
4.14.2.3	z	58
4.15	ALWrapper Class Reference	58
4.15.1	Detailed Description	63
4.15.2	Method Documentation	64
4.15.2.1	addNotification:onSource:callback:userData:	64
4.15.2.2	asaGetListenerb:	64
4.15.2.3	asaGetListenerf:	64
4.15.2.4	asaGetListeneri:	64
4.15.2.5	asaGetSourcecb:property:	65
4.15.2.6	asaGetSourcecf:property:	65
4.15.2.7	asaGetSourceci:property:	65
4.15.2.8	asaListenerb:value:	65
4.15.2.9	asaListenerf:value:	66
4.15.2.10	asaListeneri:value:	66
4.15.2.11	asaSourcecb:property:value:	66
4.15.2.12	asaSourcecf:property:value:	66
4.15.2.13	asaSourceci:property:value:	67
4.15.2.14	buffer3f:parameter:v1:v2:v3:	67
4.15.2.15	buffer3i:parameter:v1:v2:v3:	67
4.15.2.16	bufferData:format:data:size:frequency:	67
4.15.2.17	bufferDataStatic:format:data:size:frequency:	68
4.15.2.18	bufferf:parameter:value:	68
4.15.2.19	bufferfv:parameter:values:	68
4.15.2.20	bufferi:parameter:value:	69

4.15.2.21 bufferiv:parameter:values:	69
4.15.2.22 captureSamples:buffer:numSamples:	69
4.15.2.23 closeCaptureDevice:	69
4.15.2.24 closeDevice:	69
4.15.2.25 createContext:attributes:	70
4.15.2.26 deleteBuffer:	70
4.15.2.27 deleteBuffers:numBuffers:	70
4.15.2.28 deleteSource:	70
4.15.2.29 deleteSources:numSources:	71
4.15.2.30 destroyContext:	71
4.15.2.31 disable:	71
4.15.2.32 distanceModel:	71
4.15.2.33 dopplerFactor:	72
4.15.2.34 enable:	72
4.15.2.35 genBuffer	72
4.15.2.36 genBuffers:numBuffers:	72
4.15.2.37 genSource	72
4.15.2.38 genSources:numSources:	73
4.15.2.39 getBoolean:	73
4.15.2.40 getBooleanv:values:	73
4.15.2.41 getBuffer3f:parameter:v1:v2:v3:	73
4.15.2.42 getBuffer3i:parameter:v1:v2:v3:	74
4.15.2.43 getBufferf:parameter:	74
4.15.2.44 getBufferfv:parameter:values:	74
4.15.2.45 getBufferi:parameter:	74
4.15.2.46 getBufferiv:parameter:values:	75
4.15.2.47 getContextsDevice:	75
4.15.2.48 getContextsDevice:deviceReference:	75
4.15.2.49 getCurrentContext	75
4.15.2.50 getDouble:	75
4.15.2.51 getDoublev:values:	76
4.15.2.52 getEnumValue:	76
4.15.2.53 getEnumValue:name:	76
4.15.2.54 getFloat:	76
4.15.2.55 getFloatv:values:	77
4.15.2.56 getInteger:	77
4.15.2.57 getInteger:attribute:	77
4.15.2.58 getIntegerv:attribute:size:data:	77
4.15.2.59 getIntegerv:values:	78
4.15.2.60 getListener3f:v1:v2:v3:	78

4.15.2.61 getListener3i:v1:v2:v3:	78
4.15.2.62 getListenerf:	78
4.15.2.63 getListenerfv:values:	79
4.15.2.64 getListeneri:	79
4.15.2.65 getListeneriv:values:	79
4.15.2.66 getMixerOutputDataRate	79
4.15.2.67 getNullSeparatedStringList:	79
4.15.2.68 getNullSeparatedStringList:attribute:	80
4.15.2.69 getProcAddress:	80
4.15.2.70 getProcAddress:name:	80
4.15.2.71 getRenderingQuality	80
4.15.2.72 getSource3f:parameter:v1:v2:v3:	81
4.15.2.73 getSource3i:parameter:v1:v2:v3:	81
4.15.2.74 getSourcef:parameter:	81
4.15.2.75 getSourcefv:parameter:values:	81
4.15.2.76 getSourcei:parameter:	82
4.15.2.77 getSourceiv:parameter:values:	82
4.15.2.78 getSpaceSeparatedStringList:	82
4.15.2.79 getSpaceSeparatedStringList:attribute:	82
4.15.2.80 getString:	83
4.15.2.81 getString:attribute:	83
4.15.2.82 isBuffer:	83
4.15.2.83 isEnabled:	83
4.15.2.84 isExtensionPresent:	84
4.15.2.85 isExtensionPresent:name:	84
4.15.2.86 isSource:	84
4.15.2.87 listener3f:v1:v2:v3:	84
4.15.2.88 listener3i:v1:v2:v3:	85
4.15.2.89 listenerf:value:	85
4.15.2.90 listenerfv:values:	85
4.15.2.91 listeneri:value:	85
4.15.2.92 listeneriv:values:	86
4.15.2.93 makeContextCurrent:	86
4.15.2.94 makeContextCurrent:deviceReference:	86
4.15.2.95 openCaptureDevice:frequency:format:bufferSize:	86
4.15.2.96 openDevice:	87
4.15.2.97 processContext:	87
4.15.2.98 removeNotification:onSource:callback:userData:	87
4.15.2.99 setMixerOutputDataRate:	87
4.15.2.100setRenderingQuality:	88

4.15.2.101source3f:parameter:v1:v2:v3:	88
4.15.2.102source3i:parameter:v1:v2:v3:	88
4.15.2.103sourcef:parameter:value:	88
4.15.2.104sourcefv:parameter:values:	89
4.15.2.105sourcei:parameter:value:	89
4.15.2.106sourceiv:parameter:values:	89
4.15.2.107sourcePause:	89
4.15.2.108sourcePausev:numSources:	90
4.15.2.109sourcePlay:	90
4.15.2.110sourcePlayv:numSources:	90
4.15.2.111sourceQueueBuffers:numBuffers:bufferIds:	90
4.15.2.112sourceRewind:	91
4.15.2.113sourceRewindv:numSources:	91
4.15.2.114sourceStop:	91
4.15.2.115sourceStopv:numSources:	91
4.15.2.116sourceUnqueueBuffers:numBuffers:bufferIds:	92
4.15.2.117speedOfSound:	92
4.15.2.118startCapture:	92
4.15.2.119stopCapture:	92
4.15.2.120suspendContext:	92
4.16 ALWrapper(Private) Category Reference	93
4.16.1 Detailed Description	93
4.16.2 Method Documentation	93
4.16.2.1 checkIfSuccessful	93
4.16.2.2 checkIfSuccessfulWithDevice	94
4.16.2.3 decodeNullSeparatedStringList:	94
4.16.2.4 decodeSpaceSeparatedStringList:	94
4.17 IOSVersion Class Reference	94
4.17.1 Detailed Description	95
4.17.2 Method Documentation	95
4.17.2.1 SYNTHESIZE_SINGLETON_FOR_CLASS_HEADER	95
4.17.3 Property Documentation	95
4.17.3.1 version	95
4.18 NSMutableArray(WeakReferences) Category Reference	95
4.18.1 Detailed Description	96
4.18.2 Method Documentation	96
4.18.2.1 mutableArrayUsingWeakReferences	96
4.18.2.2 mutableArrayUsingWeakReferencesWithCapacity:	96
4.18.2.3 newMutableArrayUsingWeakReferences	96
4.18.2.4 newMutableArrayUsingWeakReferencesWithCapacity:	96

4.19	NSMutableDictionary(WeakReferences) Category Reference	96
4.19.1	Method Documentation	97
4.19.1.1	mutableDictionaryUsingWeakReferences	97
4.19.1.2	mutableDictionaryUsingWeakReferencesWithCapacity:	97
4.19.1.3	newMutableDictionaryUsingWeakReferences	97
4.19.1.4	newMutableDictionaryUsingWeakReferencesWithCapacity:	97
4.20	OALAction Class Reference	97
4.20.1	Detailed Description	99
4.20.2	Method Documentation	99
4.20.2.1	initWithDuration:	99
4.20.2.2	prepareWithTarget:	99
4.20.2.3	runWithTarget:	99
4.20.2.4	startAction	99
4.20.2.5	stopAction	99
4.20.2.6	updateCompletion:	100
4.20.3	Member Data Documentation	100
4.20.3.1	runningInManager_	100
4.20.4	Property Documentation	100
4.20.4.1	duration	100
4.20.4.2	elapsed	100
4.20.4.3	running	100
4.20.4.4	target	100
4.21	OALActionManager Class Reference	100
4.21.1	Detailed Description	101
4.21.2	Method Documentation	101
4.21.2.1	stopAllActions	101
4.21.2.2	SYNTHESIZE_SINGLETON_FOR_CLASS_HEADER	101
4.21.3	Member Data Documentation	101
4.21.3.1	actionsToAdd	101
4.21.3.2	actionsToRemove	101
4.21.3.3	lastTimestamp	102
4.21.3.4	stepTimer	102
4.21.3.5	targetActions	102
4.21.3.6	targets	102
4.22	OALAudioFile Class Reference	102
4.22.1	Detailed Description	103
4.22.2	Method Documentation	103
4.22.2.1	audioDataWithStartFrame:numFrames:bufferSize:	103
4.22.2.2	bufferFromUrl:reduceToMono:	103
4.22.2.3	bufferNamed:startFrame:numFrames:	104

4.22.2.4	fileWithURL:reduceToMono:	104
4.22.2.5	initWithUrl:reduceToMono:	104
4.22.3	Member Data Documentation	104
4.22.3.1	fileHandle	104
4.22.3.2	originalChannelsPerFrame	105
4.22.3.3	streamDescription	105
4.22.4	Property Documentation	105
4.22.4.1	reduceToMono	105
4.22.4.2	streamDescription	105
4.22.4.3	totalFrames	105
4.22.4.4	url	105
4.23	OALAudioSession Class Reference	105
4.23.1	Detailed Description	106
4.23.2	Method Documentation	107
4.23.2.1	forceEndInterruption	107
4.23.2.2	SYNTHESIZE_SINGLETON_FOR_CLASS_HEADER	107
4.23.3	Member Data Documentation	107
4.23.3.1	audioSessionWasActive	107
4.23.3.2	handlingErrorNotification	107
4.23.3.3	lastResetTime	107
4.23.3.4	suspendHandler	107
4.23.4	Property Documentation	107
4.23.4.1	allowIpod	107
4.23.4.2	audioRoute	108
4.23.4.3	audioSessionActive	108
4.23.4.4	audioSessionCategory	108
4.23.4.5	handleInterruptions	108
4.23.4.6	hardwareMuted	108
4.23.4.7	hardwareVolume	108
4.23.4.8	honorSilentSwitch	108
4.23.4.9	ipodDucking	108
4.23.4.10	ipodPlaying	108
4.23.4.11	preferredIOBufferDuration	109
4.23.4.12	useHardwareIfAvailable	109
4.24	OALAudioTrack Class Reference	109
4.24.1	Detailed Description	112
4.24.2	Method Documentation	112
4.24.2.1	averagePowerForChannel:	112
4.24.2.2	clear	112
4.24.2.3	fadeTo:duration:target:selector:	112

4.24.2.4	panTo:duration:target:selector:	112
4.24.2.5	peakPowerForChannel:	113
4.24.2.6	play	113
4.24.2.7	playAfterTrack:	113
4.24.2.8	playAfterTrack:timeAdjust:	113
4.24.2.9	playAtTime:	114
4.24.2.10	playFile:	114
4.24.2.11	playFile:loops:	114
4.24.2.12	playFileAsync:loops:target:selector:	114
4.24.2.13	playFileAsync:target:selector:	115
4.24.2.14	playUrl:	115
4.24.2.15	playUrl:loops:	115
4.24.2.16	playUrlAsync:loops:target:selector:	115
4.24.2.17	playUrlAsync:target:selector:	116
4.24.2.18	preloadFile:	116
4.24.2.19	preloadFile:seekTime:	116
4.24.2.20	preloadFileAsync:seekTime:target:selector:	116
4.24.2.21	preloadFileAsync:target:selector:	117
4.24.2.22	preloadUrl:	117
4.24.2.23	preloadUrl:seekTime:	117
4.24.2.24	preloadUrlAsync:seekTime:target:selector:	117
4.24.2.25	preloadUrlAsync:target:selector:	118
4.24.2.26	stop	118
4.24.2.27	stopActions	118
4.24.2.28	stopFade	118
4.24.2.29	stopPan	118
4.24.2.30	track	118
4.24.2.31	updateMeters	119
4.24.3	Member Data Documentation	119
4.24.3.1	gainAction	119
4.24.3.2	operationQueue	119
4.24.3.3	panAction	119
4.24.3.4	simulatorPlayerRef	119
4.24.3.5	suspendHandler	119
4.24.4	Property Documentation	119
4.24.4.1	autoPreload	119
4.24.4.2	currentlyLoadedUrl	119
4.24.4.3	currentTime	119
4.24.4.4	delegate	120
4.24.4.5	deviceCurrentTime	120

4.24.4.6	duration	120
4.24.4.7	gain	120
4.24.4.8	meteringEnabled	120
4.24.4.9	muted	120
4.24.4.10	numberOfChannels	120
4.24.4.11	numberOfLoops	120
4.24.4.12	pan	121
4.24.4.13	paused	121
4.24.4.14	player	121
4.24.4.15	playing	121
4.24.4.16	preloaded	121
4.24.4.17	volume	121
4.25	OALAudioTracks Class Reference	121
4.25.1	Detailed Description	122
4.25.2	Method Documentation	122
4.25.2.1	stopAllTracks	122
4.25.2.2	SYNTHESIZE_SINGLETON_FOR_CLASS_HEADER	122
4.25.3	Member Data Documentation	122
4.25.3.1	deviceTimePoller	122
4.25.3.2	suspendHandler	122
4.25.3.3	tracks	123
4.25.4	Property Documentation	123
4.25.4.1	muted	123
4.25.4.2	paused	123
4.25.4.3	tracks	123
4.26	OALCallAction Class Reference	123
4.26.1	Detailed Description	124
4.26.2	Method Documentation	124
4.26.2.1	actionWithCallTarget:selector:	124
4.26.2.2	actionWithCallTarget:selector:withObject:	124
4.26.2.3	actionWithCallTarget:selector:withObject:withObject:	125
4.26.2.4	initWithCallTarget:selector:	125
4.26.2.5	initWithCallTarget:selector:withObject:	125
4.26.2.6	initWithCallTarget:selector:withObject:withObject:	125
4.26.3	Member Data Documentation	126
4.26.3.1	callTarget_	126
4.26.3.2	numObjects_	126
4.26.3.3	object1_	126
4.26.3.4	object2_	126
4.26.3.5	selector_	126

4.27	OALConcurrentActions Class Reference	126
4.27.1	Detailed Description	127
4.27.2	Method Documentation	127
4.27.2.1	actions:	127
4.27.2.2	actionsFromArray:	127
4.27.2.3	initWithActions:	128
4.27.3	Property Documentation	128
4.27.3.1	actions	128
4.28	OALEaseAction Class Reference	128
4.28.1	Detailed Description	129
4.28.2	Method Documentation	129
4.28.2.1	actionWithShape:phase:action:	129
4.28.2.2	easeFunctionForShape:phase:	129
4.28.2.3	initWithShape:phase:action:	129
4.29	OALEaseAction() Category Reference	130
4.30	OALMoveByAction Class Reference	130
4.30.1	Detailed Description	131
4.30.2	Method Documentation	131
4.30.2.1	actionWithDuration:delta:	131
4.30.2.2	actionWithUnitsPerSecond:delta:	131
4.30.2.3	initWithDuration:delta:	131
4.30.2.4	initWithUnitsPerSecond:delta:	131
4.30.3	Member Data Documentation	132
4.30.3.1	startPoint	132
4.30.4	Property Documentation	132
4.30.4.1	delta	132
4.30.4.2	unitsPerSecond	132
4.31	OALMoveToAction Class Reference	132
4.31.1	Detailed Description	133
4.31.2	Method Documentation	133
4.31.2.1	actionWithDuration:position:	133
4.31.2.2	actionWithUnitsPerSecond:position:	133
4.31.2.3	initWithDuration:position:	134
4.31.2.4	initWithUnitsPerSecond:position:	134
4.31.3	Member Data Documentation	134
4.31.3.1	delta	134
4.31.3.2	startPoint	134
4.31.4	Property Documentation	134
4.31.4.1	position	134
4.31.4.2	unitsPerSecond	134

4.32 OALPlaceAction Class Reference	135
4.32.1 Detailed Description	135
4.32.2 Method Documentation	135
4.32.2.1 actionWithPosition:	135
4.32.2.2 initWithPosition:	136
4.32.3 Property Documentation	136
4.32.3.1 position	136
4.33 OALPropertyAction Class Reference	136
4.33.1 Method Documentation	137
4.33.1.1 actionWithDuration:propertyKey:endValue:	137
4.33.1.2 actionWithDuration:propertyKey:startValue:endValue:	137
4.33.1.3 initWithDuration:propertyKey:endValue:	137
4.33.1.4 initWithDuration:propertyKey:startValue:endValue:	138
4.33.2 Property Documentation	138
4.33.2.1 endValue	138
4.33.2.2 startValue	138
4.34 OALPropertyAction() Category Reference	138
4.35 OALPropertyAction(Audio) Category Reference	138
4.36 OALSequentialActions Class Reference	139
4.36.1 Detailed Description	140
4.36.2 Method Documentation	140
4.36.2.1 actions:	140
4.36.2.2 actionsFromArray:	140
4.36.2.3 initWithActions:	140
4.36.3 Member Data Documentation	140
4.36.3.1 actionIndex_	141
4.36.3.2 pCurrentActionComplete_	141
4.36.3.3 pCurrentActionDuration_	141
4.36.3.4 pLastComplete_	141
4.36.4 Property Documentation	141
4.36.4.1 actions	141
4.37 OALSimpleAudio Class Reference	141
4.37.1 Detailed Description	144
4.37.2 Method Documentation	144
4.37.2.1 playBg	144
4.37.2.2 playBg:	144
4.37.2.3 playBg:loop:	144
4.37.2.4 playBg:volume:pan:loop:	145
4.37.2.5 playBgWithLoop:	145
4.37.2.6 playBuffer:volume:pitch:pan:loop:	145

4.37.2.7	playEffect:	146
4.37.2.8	playEffect:loop:	146
4.37.2.9	playEffect:volume:pitch:pan:loop:	146
4.37.2.10	preloadBg:	146
4.37.2.11	preloadBg:seekTime:	147
4.37.2.12	preloadEffect:	147
4.37.2.13	preloadEffect:reduceToMono:	147
4.37.2.14	resetToDefault	147
4.37.2.15	sharedInstanceWithReservedSources:monoSources:stereoSources:	147
4.37.2.16	sharedInstanceWithSources:	148
4.37.2.17	stopAllEffects	148
4.37.2.18	stopBg	148
4.37.2.19	stopEverything	148
4.37.2.20	SYNTHESIZE_SINGLETON_FOR_CLASS_HEADER	148
4.37.2.21	unloadAllEffects	149
4.37.2.22	unloadEffect:	149
4.37.3	Member Data Documentation	149
4.37.3.1	pendingLoadCount	149
4.37.3.2	preloadCache	149
4.37.4	Property Documentation	149
4.37.4.1	allowIpod	149
4.37.4.2	backgroundTrack	149
4.37.4.3	backgroundTrackURL	150
4.37.4.4	bgMuted	150
4.37.4.5	bgPaused	150
4.37.4.6	bgPlaying	150
4.37.4.7	bgVolume	150
4.37.4.8	channel	150
4.37.4.9	context	150
4.37.4.10	device	150
4.37.4.11	effectsMuted	150
4.37.4.12	effectsPaused	150
4.37.4.13	effectsVolume	150
4.37.4.14	honorSilentSwitch	151
4.37.4.15	interrupted	151
4.37.4.16	manuallySuspended	151
4.37.4.17	muted	151
4.37.4.18	paused	151
4.37.4.19	preloadCacheCount	151
4.37.4.20	preloadCacheEnabled	151

4.37.4.21 reservedSources	151
4.37.4.22 suspended	151
4.37.4.23 useHardwareIfAvailable	151
4.38 OALSuspendHandler Class Reference	152
4.38.1 Detailed Description	153
4.38.2 Method Documentation	153
4.38.2.1 addSuspendListener:	153
4.38.2.2 handlerWithTarget:selector:	153
4.38.2.3 initWithTarget:selector:	154
4.38.2.4 removeSuspendListener:	154
4.38.3 Member Data Documentation	154
4.38.3.1 interruptLock	154
4.38.3.2 listeners	154
4.38.3.3 manualSuspendLock	154
4.38.3.4 manualSuspendStates	154
4.38.3.5 suspendStatusChangeSelector	154
4.38.4 Property Documentation	154
4.38.4.1 interrupted	154
4.38.4.2 manuallySuspended	155
4.38.4.3 suspended	155
4.39 <OALSuspendListener> Protocol Reference	155
4.39.1 Detailed Description	156
4.39.2 Property Documentation	156
4.39.2.1 interrupted	156
4.39.2.2 manuallySuspended	156
4.40 <OALSuspendManager> Protocol Reference	156
4.40.1 Detailed Description	157
4.40.2 Method Documentation	157
4.40.2.1 addSuspendListener:	157
4.40.2.2 removeSuspendListener:	158
4.40.3 Property Documentation	158
4.40.3.1 suspended	158
4.41 OALTargetedAction Class Reference	158
4.41.1 Detailed Description	159
4.41.2 Method Documentation	159
4.41.2.1 actionWithTarget:action:	159
4.41.2.2 initWithTarget:action:	159
4.41.3 Member Data Documentation	159
4.41.3.1 action_	159
4.41.4 Property Documentation	159

4.41.4.1	forcedTarget	159
4.42	OALTools Class Reference	159
4.42.1	Detailed Description	160
4.42.2	Method Documentation	160
4.42.2.1	defaultBundle	160
4.42.2.2	notifyAudioSessionError:function:description:	160
4.42.2.3	notifyExtAudioError:function:description:	161
4.42.2.4	setDefaultBundle:	161
4.42.2.5	urlForPath:	161
4.42.2.6	urlForPath:bundle:	161
4.43	OpenALManager Class Reference	162
4.43.1	Detailed Description	163
4.43.2	Method Documentation	163
4.43.2.1	bufferAsyncFromFile:reduceToMono:target:selector:	163
4.43.2.2	bufferAsyncFromFile:target:selector:	164
4.43.2.3	bufferAsyncFromUrl:reduceToMono:target:selector:	164
4.43.2.4	bufferAsyncFromUrl:target:selector:	164
4.43.2.5	bufferFromFile:	165
4.43.2.6	bufferFromFile:reduceToMono:	165
4.43.2.7	bufferFromUrl:	165
4.43.2.8	bufferFromUrl:reduceToMono:	166
4.43.2.9	clearAllBuffers	166
4.43.2.10	SYNTHESIZE_SINGLETON_FOR_CLASS_HEADER	166
4.43.3	Member Data Documentation	166
4.43.3.1	devices	166
4.43.3.2	operationQueue	166
4.43.3.3	suspendHandler	166
4.43.4	Property Documentation	166
4.43.4.1	availableCaptureDevices	166
4.43.4.2	availableDevices	167
4.43.4.3	currentContext	167
4.43.4.4	defaultCaptureDeviceSpecifier	167
4.43.4.5	defaultDeviceSpecifier	167
4.43.4.6	devices	167
4.43.4.7	mixerOutputFrequency	167
4.43.4.8	renderingQuality	167

Chapter 1

ObjectAL for iPhone

iOS Audio development, minus the headache.

Version 2.2

Copyright 2009-2013 Karl Stenerud

Released under the [Apache License v2.0](#)

1.1 Contents

- [Introduction](#)
- [ObjectAL and OpenAL](#)
- [Adding ObjectAL to your project](#) (also, installing the documentation into XCode)
- [Compile-Time Configuration](#)
- [Audio Formats](#)
- [Choosing Playback Types](#)
- [Using OALSimpleAudio](#)
- [Using the OpenAL Objects and OALAudioTrack](#)
- [Other Examples](#)
- [iOS Issues that can impede playback](#)
- [Simulator Issues](#)

1.2 Introduction

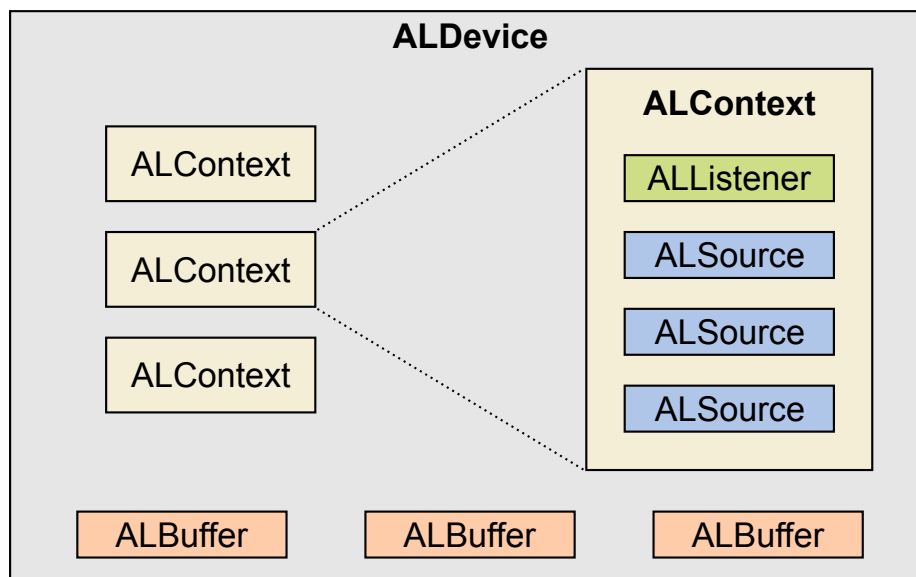
ObjectAL for iPhone is designed to be a simpler, more intuitive interface to OpenAL and AVAudioPlayer. There are four main parts to **ObjectAL for iPhone**:

OALSimpleAudio (Simpler Interface)			
ObjectAL (Sound Effects)		OALAudioSession (Session Management)	OALAudioTrack (Long-play Audio)
OpenAL	ExtAudio	AudioSession	AVAudioPlayer

- **ObjectAL** gives you full access to the OpenAL system without the hassle of the C API. All OpenAL operations can be performed using first class objects and properties, without needing to muddle around with arrays of data, maintain IDs, or pass around pointers to basic types. ObjectALManager also provides sound loading routines.
- **OALAudioTrack** provides a simpler interface to AVAudioPlayer, allowing you to play, stop, pause, fade, and mute background music tracks.
- **OALAudioSession** handles audio session management in iOS devices, and provides an easy way to configure session behavior such as how to handle iPod-style music and the silent switch.
- **OALSimpleAudio** layers on top of the other three, providing an even simpler interface for playing background music and sound effects.

1.3 ObjectAL and OpenAL

ObjectAL follows the same basic principles as the **OpenAL API by Creative Labs**.



- **OpenALManager** provides some overall controls that affect everything, manages the current context, and provides audio loading routines.
- **ALDevice** represents a physical audio device.
Each device can have one or more contexts (**ALContext**) created on it, and can have multiple buffers (**ALBuffer**) associated with it.
- **ALContext** controls the overall sound environment, such as distance model, doppler effect, and speed of sound.
Each context has one listener (**ALListener**), and can have multiple sources (**ALSource**) opened on it (up to a maximum of 32 overall on iPhone).
- **ALListener** represents the listener of sounds originating on its context (one listener per context). It has position, orientation, and velocity.
- **ALSource** is a sound emitting source that plays sound data from an **ALBuffer**. It has position, direction, velocity, as well as other properties which determine how the sound is emitted.

- [ALChannelSource](#) allows you to reserve a certain number of sources for special purposes.
- [ALBuffer](#) is simply a container for sound data. Only linear PCM is supported directly, but [OpenALManager](#) load methods, and [OALSimpleAudio](#) effect preload and play methods, will automatically convert any formats that don't require hardware decoding (though conversion results in a longer loading time).

Note: While OpenAL allows for multiple devices and contexts, in practice you'll only use one device and one context when using OpenAL under iOS.

Further information regarding the more advanced features of OpenAL (such as distance models) are available via the [OpenAL Documentation at Creative Labs](#).

In particular, read up on the various property values for sources and listeners (such as Doppler Shift) in the [OpenAL Programmer's Guide](#), and distance models in section 3 of the [OpenAL Specification](#).

1.4 Adding ObjectAL to your project

To add ObjectAL to your project, do the following:

1. Copy ObjectAL/ObjectAL from this project into your project. You can simply drag it into the "Groups & Files" section in xcode if you like (be sure to select "Copy items into destination group's folder").
Alternatively, you can build ObjectAL as a static library (as it's configured to do in the ObjectAL demo project).
2. Add the following frameworks to your project:
 - OpenAL.framework
 - AudioToolbox.framework
 - AVFoundation.framework
3. Start using ObjectAL!

Note: The demos in this project use [Cocos2d](#), a very nice 2d game engine. However, ObjectAL doesn't require it. You can just as easily use ObjectAL in your Cocoa app or anything you wish.

Note #2: You do NOT have to provide a link to the Apache license from within your application. Simply including a copy of the license in your project is sufficient.

1.4.1 Installing the ObjectAL Documentation into XCode

By installing the ObjectAL documentation into XCode's Developer Documentation system, you gain the ability to look up ObjectAL classes and methods just like you'd look up Apple classes and methods. You can install the ObjectAL documentation into XCode's Developer Documentation system by doing the following:

1. Install [Doxygen](#). You can either use the OSX installer or [Homebrew](#).
2. Build the "Documentation" target in this project.
3. Open the developer documentation and type "ObjectAL" into the search box.

1.5 Compile-Time Configuration

[ObjectALConfig.h](#) contains configuration defines that will affect at a high level how ObjectAL behaves. Look inside [ObjectALConfig.h](#) to see what can be configured, and what each configuration value does.

The recommended values are fine for most users, but Cocos2D users may want to set `OBJECTAL_CFG_USE_COCOS2D_ACTIONS` so that the audio actions (such as fade) use the Cocos2D action manager.

1.6 Audio Formats

The audio formats officially supported by Apple are [defined here](#).

1.6.1 OALAudioTrack Supported Formats

[OALAudioTrack](#) supports all hardware and software decoded formats.

1.6.2 OpenAL Supported Formats

OpenAL officially supports 8 or 16 bit PCM data only. However, Apple's implementation only seems to work with 16 bit data.

The effects preloading/playing methods in [OALSimpleAudio](#) and the buffer loading methods in [OpenALManager](#) can load any audio file that can be software decoded. However, there is a cost incurred at load time converting to a native OpenAL format. To avoid this, convert all of your samples to a CAFF container with 16-bit little endian integer PCM format and the same sample rate as "mixerOutputFrequency" in [OpenALManager](#) (by default, 44100Hz). Note, however, that uncompressed files can get quite large.

Convert to iOS native uncompressed format using Apple's "afconvert" command line tool:

```
afconvert -f caff -d LEI16@44100 sourcefile.wav destfile.caf
```

Alternatively, if sound file load time is not an issue for you, you can lower your app footprint size (for over-the-air app download) by using a compressed format.

Convert to AAC compressed format with CAFF container using Apple's "afconvert" command line tool:

```
afconvert -f caff -d aac sourcefile.wav destfile.caf
```

1.7 Choosing Playback Types

OpenAL ([ALSource](#), or effects in [OALSimpleAudio](#)) and **AVAudioPlayer** ([OALAudioTrack](#), or background audio in [OALSimpleAudio](#)) are playback technologies built for different purposes. OpenAL is designed for game-style short sound effects that have no playback delay. AVAudioPlayer is designed for music playback. You can of course mix and match as you please.

	OpenAL	AVAudioPlayer
Playback Delay	None	Small delay if not preloaded
Format on Disk	Any software decodable format	Any software decodable format, or any hardware format if using hardware
Decoding	During load	During playback
Memory Use	Entire file loaded and decompressed into memory	File streamed realtime (very low memory use)
Max Simult. Sources	32	As many as the CPU can handle
Playback Performance	Good	Excellent with 1 track (if using hardware). Good with 2 tracks. Not so good with more (each non-hardware track taxes the CPU significantly, especially if the files are compressed).

Looped Playback	Yes (on or off)	Yes (specify number of loops or -1 = forever)
Panning	Yes (mono files only)	Yes (iOS 4.0+ only)
Positional Audio	Yes (mono files only)	No
Modify Pitch	Yes	No
Audio Power Metering	No	Yes

1.8 Using OALSimpleAudio

By far, the easiest component to use is [OALSimpleAudio](#). You sacrifice some power for ease-of-use, but for many projects it is more than sufficient. You can also use your own instances of [OALAudioTrack](#), [ALSource](#), [ALBuffer](#) and such alongside of [OALSimpleAudio](#) if you want (just be sure to set [OALSimpleAudio](#)'s reservedSources to less than 32 if you want to make your own instances of [ALSource](#)).

Here is a code example using purely [OALSimpleAudio](#):

```
// OALSimpleAudioSample.h

@interface OALSimpleAudioSample : NSObject
{
    // No objects to keep track of...
}

@end

// OALSimpleAudioSample.m

#import "OALSimpleAudioSample.h"
#import "ObjectAL.h"

#define SHOOT_SOUND @"shoot.caf"
#define EXPLODE_SOUND @"explode.caf"

#define INGAME_MUSIC_FILE @"bg_music.mp3"
#define GAMEOVER_MUSIC_FILE @"gameover_music.mp3"

@implementation OALSimpleAudioSample

- (id) init
{
    if(nil != (self = [super init]))
    {
        // We don't want ipod music to keep playing since
        // we have our own bg music.
        [OALSimpleAudio sharedInstance].allowIpod = NO;

        // Mute all audio if the silent switch is turned on.
        [OALSimpleAudio sharedInstance].honorSilentSwitch = YES;

        // This loads the sound effects into memory so that
        // there's no delay when we tell it to play them.
        [[OALSimpleAudio sharedInstance] preloadEffect:SHOOT_SOUND];
        [[OALSimpleAudio sharedInstance] preloadEffect:EXPLODE_SOUND];
    }
    return self;
}

- (void) onGameStart
{
    // Play the BG music and loop it.
    [[OALSimpleAudio sharedInstance] playBg:INGAME_MUSIC_FILE loop:YES];
}

- (void) onGamePause
{
    [OALSimpleAudio sharedInstance].paused = YES;
}

- (void) onGameResume
{
    [OALSimpleAudio sharedInstance].paused = NO;
}

- (void) onGameOver
{
}
```

```

    // Could use stopEverything here if you want
    [[OALSimpleAudio sharedInstance] stopAllEffects];

    // We only play the game over music through once.
    [[OALSimpleAudio sharedInstance] playBg:GAMEOVER_MUSIC_FILE];
}

- (void) onShipShotABullet
{
    [[OALSimpleAudio sharedInstance] playEffect:SHOOT_SOUND];
}

- (void) onShipGotHit
{
    [[OALSimpleAudio sharedInstance] playEffect:EXPLODE_SOUND];
}

- (void) onQuitToMainMenu
{
    // Stop all music and sound effects.
    [[OALSimpleAudio sharedInstance] stopEverything];

    // Unload all sound effects and bg music so that it doesn't fill
    // memory unnecessarily.
    [[OALSimpleAudio sharedInstance] unloadAllEffects];
}

@end

```

1.9 Using the OpenAL Objects and OALAudioTrack

The OpenAL objects and [OALAudioTrack](#) offer you much more power at the cost of complexity. Here's the same thing as above, done using OpenAL components and [OALAudioTrack](#):

```

// OpenALAudioTrackSample.h

#import <Foundation/Foundation.h>
#import "ObjectAL.h"

@interface OpenALAudioTrackSample : NSObject
{
    // Sound Effects
    ALDevice* device;
    ALContext* context;
    ALChannelSource* channel;
    ALBuffer* shootBuffer;
    ALBuffer* explosionBuffer;

    // Background Music
    OALAudioTrack* musicTrack;
}

@end

// OpenALAudioTrackSample.m

#import "OpenALAudioTrackSample.h"

#define SHOOT_SOUND @"shoot.caf"
#define EXPLODE_SOUND @"explode.caf"

#define INGAME_MUSIC_FILE @"bg_music.mp3"
#define GAMEOVER_MUSIC_FILE @"gameover_music.mp3"

@implementation OpenALAudioTrackSample

- (id) init
{
    if (nil != (self = [super init]))
    {
        // Create the device and context.
        // Note that it's easier to just let OALSimpleAudio handle
        // these rather than make and manage them yourself.
        device = [[ALDevice deviceWithDeviceSpecifier:nil] retain];
        context = [[ALContext contextOnDevice:device
            attributes:nil] retain];
        [OpenALManager sharedInstance].currentContext = context;
    }
}

```

```

    // Deal with interruptions for me!
    [OALAudioSession sharedInstance].handleInterruptions = YES;

    // We don't want ipod music to keep playing since
    // we have our own bg music.
    [OALAudioSession sharedInstance].allowIpod = NO;

    // Mute all audio if the silent switch is turned on.
    [OALAudioSession sharedInstance].honorSilentSwitch = YES;

    // Take all 32 sources for this channel.
    // (we probably won't use that many but what the heck!)
    channel = [[ALChannelSource channelWithSources:32] retain];

    // Preload the buffers so we don't have to load and play them later.
    shootBuffer = [[[OpenALManager sharedInstance]
        bufferFromFile:SHOOT_SOUND] retain];
    explosionBuffer = [[[OpenALManager sharedInstance]
        bufferFromFile:EXPLODE_SOUND] retain];

    // Background music track.
    musicTrack = [[OALAudioTrack track] retain];
}
return self;
}

- (void) dealloc
{
    [musicTrack release];

    [channel release];
    [shootBuffer release];
    [explosionBuffer release];

    // Note: You'll likely only have one device and context open throughout
    // your program, so in a real program you'd be better off making a
    // singleton object that manages the device and context, rather than
    // allocating/deallocating it here.
    // Most of the demos just let OALSimpleAudio manage the device and context
    // for them.
    [context release];
    [device release];

    [super dealloc];
}

- (void) onGameStart
{
    // Play the BG music and loop it forever.
    [musicTrack playFile:INGAME_MUSIC_FILE loops:-1];
}

- (void) onGamePause
{
    musicTrack.paused = YES;
    channel.paused = YES;
}

- (void) onGameResume
{
    channel.paused = NO;
    musicTrack.paused = NO;
}

- (void) onGameOver
{
    [channel stop];
    [musicTrack stop];

    // We only play the game over music through once.
    [musicTrack playFile:GAMEOVER_MUSIC_FILE];
}

- (void) onShipShotABullet
{
    [channel play:shootBuffer];
}

- (void) onShipGotHit
{
    [channel play:explosionBuffer];
}

- (void) onQuitToMainMenu
{
    // Stop all music and sound effects.

```

```

    [channel stop];
    [musicTrack stop];
}

@end

```

1.10 Other Examples

The demo scenes in this distribution have been crafted to demonstrate common uses of this library. Try them out and go through the code to see how it's done. I've done my best to keep the code readable. Really!

You can try out the demos by building and running the OALDemo target for iOS or OSX.

The current demos are:

- **SingleSourceDemo:** Demonstrates using a location based source and a listener.
- **TwoSourceDemo:** Demonstrates using two location based sources and a listener.
- **VolumePitchPanDemo:** Demonstrates using gain, pitch, and pan controls.
- **CrossFadeDemo:** Demonstrates crossfading between two sources.
- **ChannelsDemo:** Demonstrates using audio channels.
- **FadeDemo:** Demonstrates realtime fading with [OALAudioTrack](#) and [ALSource](#).
- **AudioTrackDemo:** Demonstrates using multiple [OALAudioTrack](#) objects.
- **PlanetKillerDemo:** Demonstrates using [OALSimpleAudio](#) in a game setting.
- **IntroAndMainTrackDemo:** Demonstrates a short intro track followed by a main loop track.
- **SourceNotificationsDemo:** Demonstrates using OpenAL playback notifications.
- **HardwareDemo:** Demonstrates hardware monitoring features.
- **AudioSessionDemo:** Allows you to play with various audio session settings.

1.11 iOS Issues that can impede playback

Certain versions of iOS have bugs or quirks, requiring workarounds. ObjectAL tries to handle most of these automatically, but there are cases that require specific handling by the developer. These are:

1.11.1 MPMoviePlayerController on iOS 3.x

In iOS 3.x, MPMoviePlayerController doesn't play nice, and takes over the audio session when you play a video. In order to mitigate this, you must manually suspend OpenAL, play the video, and then manually unsuspend once video playback finishes:

```

- (void) playVideo
{
    if ([myMoviePlayer respondsToSelector:@selector(view)])
    {
        [myMoviePlayer setFullscreen:YES animated:YES];
    }
    else
    {
        // No "view" method means we are < 4.0
        // Manually suspend so iOS 3.x doesn't clobber our session!
        [OpenALManager sharedInstance].manuallySuspended = YES;
    }

    [myMoviePlayer play];
}

```

```

[[NSNotificationCenter defaultCenter]
 addObserver:self
 selector:@selector(movieFinishedCallback:)
 name:MPMoviePlayerPlaybackDidFinishNotification
 object:myMoviePlayer];
}

-(void)movieFinishedCallback:(NSNotification *)notification
{
    if([myMoviePlayer respondsToSelector:@selector(view)])
    {
        if (myMoviePlayer.fullscreen)
        {
            [myMoviePlayer setFullscreen:NO animated:YES];
        }
    }
    else
    {
        // No "view" method means we are < 4.0
        // Manually unsuspend
        [OpenALManager sharedInstance].manuallySuspended = NO;
    }
}

```

1.11.2 MPMusicPlayerController on iOS 4.0

On iOS 4.0, MPMusicPlayerController sends an interrupt when it begins playback, but doesn't send a corresponding "end interrupt" when it ends. To work around this, force an "end interrupt" after starting playback:

```
[[OALAudioSession sharedInstance] forceEndInterruption];
```

1.12 Simulator Issues

As you've likely heard time and time again, the simulator is no substitute for the real thing. The simulator is buggy. It can run faster or slower than a real device. It fails system calls that a real device doesn't. It shows graphics glitches that a real device doesn't. Sounds stop working, clicks and static, dogs and cats living together, etc, etc. When things look wrong, try it on a real device before bugging people.

1.12.1 Simulator Limitations

The simulator does not support setting audio modes, so setting allowIpad or honorSilentSwitch in [OALAudioSession](#) will have no effect in the simulator.

1.12.2 Error Codes on the Simulator

From time to time, the simulator can get confused, and start spitting out spurious errors. When this happens, check on a real device to make sure it's not just a simulator issue. Usually quitting and restarting the simulator will fix it, but sometimes you may have to reboot your machine as well.

1.12.3 Playback Issues

The simulator is notoriously finicky when it comes to audio playback. Any number of programs you've installed on your mac can cause the simulator to stop playing bg music, or effects, or both!

Some things to check when sound stops working:

- Try resetting and restarting the simulator.
- Try restarting XCode, cleaning, and recompiling your project.
- Try rebooting your computer.

- Open "Audio MIDI Setup" (type "midi" into spotlight to find it) and make sure "Built-in Output" is set to 44100.0 Hz.
- Go to System Preferences -> Sound -> Output, and ensure that "Play sound effects through" is set to "- Internal Speakers"
- Go to System Preferences -> Sound -> Input, and ensure that it is using internal sound devices.
- Go to System Preferences -> Sound -> Sound Effects, and ensure "Play user interface sound effects" is checked.
- Some codecs may cause problems with sound playback. Try removing them.
- Programs that redirect audio can wreak havoc on the simulator. Try removing them.

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

ALContext()	35
ALOrientation	41
ALPoint	42
ALSoundSourcePool(Private)	51
ALVector	58
ALWrapper(Private)	93
<AVAudioPlayerDelegate>	
OALAudioTrack	109
NSMutableArray(WeakReferences)	95
NSMutableDictionary(WeakReferences)	96
NSObject	
ALBuffer	15
ALCaptureDevice	18
ALChannelSource	21
ALContext	29
ALDevice	35
ALListener	38
ALSoundSourcePool	49
ALSource	51
ALWrapper	58
IOSVersion	94
OALAction	97
OALCallAction	123
OALConcurrentActions	126
OALEaseAction	128
OALMoveByAction	130
OALMoveToAction	132
OALPlaceAction	135
OALPropertyAction	136
OALSequentialActions	139
OALTargetedAction	158
OALActionManager	100
OALAudioFile	102
OALAudioSession	105
OALAudioTrack	109
OALAudioTracks	121
OALSimpleAudio	141
OALSuspendHandler	152

OALTools	159
OpenALManager	162
<NSObject>	
<ALSoundSource>	42
ALChannelSource	21
ALSource	51
OALEaseAction()	130
OALPropertyAction()	138
OALPropertyAction(Audio)	138
<OALSuspendListener>	155
<OALSuspendManager>	156
ALContext	29
ALDevice	35
ALListener	38
ALSource	51
OALAudioSession	105
OALAudioTrack	109
OALAudioTracks	121
OpenALManager	162

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

ALBuffer	A buffer for audio data that will be played via a SoundSource	15
ALCaptureDevice	<i>UNIMPLEMENTED FOR IOS</i> An OpenAL device for capturing sound data	18
ALChannelSource	A Sound source composed of other sources	21
ALContext	A context encompasses a single listener and a series of sources	29
ALContext()	35
ALDevice	A device is a logical mapping to an audio device through the OpenAL implementation	35
ALListener	The listener represents the user who is listening to sounds in 3D space	38
ALOrientation	Represents an orientation, consisting of an "at" vector (representing the "forward" direction), and the "up" vector (representing "up" for the subject)	41
ALPoint	Represents a 3-dimensional point for certain ObjectAL properties	42
<ALSoundSource>	Manages all properties relating to an OpenAL sound source	42
ALSoundSourcePool	A pool of sound sources, which can be fetched based on availability	49
ALSoundSourcePool(Private)	Private interface to SoundSourcePool	51
ALSource	A source represents an object that emits sound which can be heard by a listener	51
ALVector	Represents a 3-dimensional vector for certain ObjectAL properties	58
ALWrapper	A thin wrapper around the C OpenAL API, with a few convenience methods thrown in	58
ALWrapper(Private)	Private interface to ALWrapper	93
IOSVersion	Reports the version of iOS being run on the current device	94
NSMutableArray(WeakReferences)	Adds to NSMutableArray the ability to create an array that keeps weak references	95
NSMutableDictionary(WeakReferences)	96

OALAction	Represents an action that can be performed on an object	97
OALActionManager	Manages all ObjectAL actions	100
OALAudioFile	Maintains an open audio file and allows loading data from that file into new ALBuffer objects . .	102
OALAudioSession	Handles the audio session and interrupts	105
OALAudioTrack	Plays an audio track via AVAudioPlayer	109
OALAudioTracks	Keeps track of all AudioTrack objects	121
OALCallAction	Calls a selector on a target	123
OALConcurrentActions	A set of actions that get run concurrently	126
OALEaseAction	Applies an easing function to another action	128
OALEaseAction()	130
OALMoveByAction	Moves the target from its current position by the specified delta over time in 3D space	130
OALMoveToAction	Moves the target from its current position to the specified position over time in 3D space	132
OALPlaceAction	Places the target at the specified position	135
OALPropertyAction	136
OALPropertyAction()	138
OALPropertyAction(Audio)	138
OALSequentialActions	A set of actions that get run in sequence	139
OALSimpleAudio	A simpler interface to the ObjectAL sound library	141
OALSuspendHandler	Provides two controls (interrupted and manuallySuspended) for suspending a slave object, and also propagates such control messages to interested listeners	152
<OALSuspendListener>	Allows an object to participate in interrupt and suspend operations	155
<OALSuspendManager>	A suspend manager is a listener that also allows other objects to subscribe to receive events as the manager receives them	156
OALTargetedAction	Ignores whatever target it was invoked upon and applies the specified action on the target specified at creation time	158
OALTools	Miscellaneous tools used by ObjectAL	159
OpenALManager	Manager class for OpenAL objects (ObjectAL)	162

Chapter 4

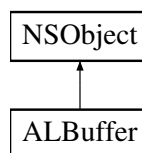
Class Documentation

4.1 ALBuffer Class Reference

A buffer for audio data that will be played via a SoundSource.

```
#import <ALBuffer.h>
```

Inheritance diagram for ALBuffer:



Instance Methods

- (id) - [initWithName:data:size:format:frequency:](#)
Initialize the buffer.
- (ALBuffer *) - [sliceWithName:offset:size:](#)
Returns a part of the buffer as a new buffer.

Class Methods

- (id) + [bufferWithName:data:size:format:frequency:](#)
Make a new buffer.

Protected Attributes

- void * [bufferData](#)
The uncompressed sound data to play.
- ALBuffer * [parentBuffer](#)
The parent buffer (which owns the uncompressed data)

Properties

- ALint [bits](#)
The size of a sample in bits.

- ALuint [bufferId](#)
The ID assigned to this buffer by OpenAL.
- ALint [channels](#)
The number of channels the buffer data plays in.
- ALDevice * [device](#)
The device this buffer was created for.
- ALenum [format](#)
The format of the audio data (see [al.h](#), [AL_FORMAT_XXX](#)).
- ALint [frequency](#)
The frequency this buffer runs at.
- NSString * [name](#)
The name given to this buffer upon creation.
- ALint [size](#)
The size, in bytes, of the currently loaded buffer data.
- float [duration](#)
The duration of the sample in this buffer, in seconds.
- bool [freeDataOnDestroy](#)
If true, calls [free\(\)](#) on the audio data when this object gets destroyed.

4.1.1 Detailed Description

A buffer for audio data that will be played via a SoundSource.

See Also

SoundSource

4.1.2 Method Documentation

4.1.2.1 + (id) [bufferWithName:](#) (NSString*) *name* data:(void*) *data* size:(ALsizei) *size* format:(ALenum) *format* frequency:(ALsizei) *frequency*

Make a new buffer.

Parameters

<i>name</i>	Optional name that you can use to identify this buffer in your code.
<i>data</i>	The sound data. Note: ALBuffer will call free() on this data when it is destroyed!
<i>size</i>	The size of the data in bytes.
<i>format</i>	The format of the data (see the Core Audio documentation).
<i>frequency</i>	The sampling frequency in Hz.

Returns

A new buffer.

4.1.2.2 - (id) [initWithName:](#) (NSString*) *name* data:(void*) *data* size:(ALsizei) *size* format:(ALenum) *format* frequency:(ALsizei) *frequency*

Initialize the buffer.

Parameters

<i>name</i>	Optional name that you can use to identify this buffer in your code.
<i>data</i>	The sound data. Note: ALBuffer will call <code>free()</code> on this data when it is destroyed!
<i>size</i>	The size of the data in bytes.
<i>format</i>	The format of the data (see the Core Audio documentation).
<i>frequency</i>	The sampling frequency in Hz.

Returns

The initialized buffer.

4.1.2.3 - (ALBuffer *) sliceWithName: (NSString *) sliceName offset:(ALsizei) offset size:(ALsizei) size

Returns a part of the buffer as a new buffer.

You can use this method to split a buffer into a sub-buffers. The sub-buffers retain a reference to their parent buffer, and share the same memory. Therefore, modifying the parent buffer contents will affect its slices and vice-versa.

Parameters

<i>sliceName</i>	Optional name that you can use to identify the created buffer in your code.
<i>offset</i>	The offset in sound frames where the slice starts.
<i>size</i>	The size of the slice in frames.

Returns

The requested buffer.

4.1.3 Member Data Documentation

4.1.3.1 - (void*) bufferData [protected]

The uncompressed sound data to play.

4.1.3.2 - (ALBuffer*) parentBuffer [protected]

The parent buffer (which owns the uncompressed data)

4.1.4 Property Documentation

4.1.4.1 - (ALint) bits [read], [nonatomic], [assign]

The size of a sample in bits.

4.1.4.2 - (ALuint) bufferId [read], [nonatomic], [assign]

The ID assigned to this buffer by OpenAL.

4.1.4.3 - (ALint) channels [read], [nonatomic], [assign]

The number of channels the buffer data plays in.

4.1.4.4 - (ALDevice *) device [read], [nonatomic], [retain]

The device this buffer was created for.

4.1.4.5 - (float) duration [read], [nonatomic], [assign]

The duration of the sample in this buffer, in seconds.

4.1.4.6 - (ALenum) format [read], [nonatomic], [assign]

The format of the audio data (see al.h, AL_FORMAT_XXX).

4.1.4.7 - (bool) freeDataOnDestroy [read], [write], [nonatomic], [assign]

If true, calls free() on the audio data when this object gets destroyed.

Default: YES

4.1.4.8 - (ALint) frequency [read], [nonatomic], [assign]

The frequency this buffer runs at.

4.1.4.9 - (NSString *) name [read], [write], [nonatomic], [retain]

The name given to this buffer upon creation.

You may change it at runtime if you wish.

4.1.4.10 - (ALint) size [read], [nonatomic], [assign]

The size, in bytes, of the currently loaded buffer data.

The documentation for this class was generated from the following files:

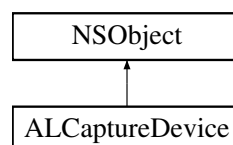
- ALBuffer.h
- ALBuffer.m

4.2 ALCaptureDevice Class Reference

UNIMPLEMENTED FOR IOS An OpenAL device for capturing sound data.

```
#import <ALCaptureDevice.h>
```

Inheritance diagram for ALCaptureDevice:



Instance Methods

- (id) - [initWithDeviceSpecifier:frequency:format:bufferSize:](#)
Open the specified device.
- (bool) - [startCapture](#)
Start capturing samples.
- (bool) - [stopCapture](#)
Stop capturing samples.
- (bool) - [moveSamples:toBuffer:](#)
Move captured samples to the specified buffer.
- (bool) - [isExtensionPresent:](#)
Check if the specified extension is present.
- (void *) - [getProcAddress:](#)
Get the address of the specified procedure (C function address).

Class Methods

- (id) + [deviceWithDeviceSpecifier:frequency:format:bufferSize:](#)
Open the specified device.

Properties

- int [captureSamples](#)
The number of capture samples available.
- ALCdevice * [device](#)
The OpenAL device pointer.
- NSArray * [extensions](#)
List of strings describing all extensions available on this device (NSString).*
- int [majorVersion](#)
The specification revision for this implementation (major version).
- int [minorVersion](#)
The specification revision for this implementation (minor version).

4.2.1 Detailed Description

UNIMPLEMENTED FOR IOS An OpenAL device for capturing sound data.

Note: This functionality is NOT implemented in iOS OpenAL!

This class is a placeholder in case such functionality is added in a future iOS SDK.

4.2.2 Method Documentation

4.2.2.1 + (id) [deviceWithDeviceSpecifier:](#) (NSString*) *deviceSpecifier* frequency:(ALCuint) *frequency* format:(ALCenum) *format* bufferSize:(ALCsizei) *bufferSize*

Open the specified device.

Parameters

<i>deviceSpecifier</i>	The name of the device to open (nil = default device).
<i>frequency</i>	The frequency to capture at.
<i>format</i>	The audio format to capture as.
<i>bufferSize</i>	The size of buffer that the device must allocate for audio capture.

Returns

A new capture device.

4.2.2.2 - (void *) getProcAddress: (NSString*) functionName

Get the address of the specified procedure (C function address).

Parameters

<i>functionName</i>	The name of the procedure to get.
---------------------	-----------------------------------

Returns

the procedure's address, or NULL if it wasn't found.

4.2.2.3 - (id) initWithDeviceSpecifier: (NSString*) deviceSpecifier frequency:(ALCuint) frequency format:(ALCenum) format bufferSize:(ALCsizei) bufferSize

Open the specified device.

Parameters

<i>deviceSpecifier</i>	The name of the device to open (nil = default device).
<i>frequency</i>	The frequency to capture at.
<i>format</i>	The audio format to capture as.
<i>bufferSize</i>	The size of buffer that the device must allocate for audio capture.

Returns

The initialized capture device.

4.2.2.4 - (bool) isExtensionPresent: (NSString*) name

Check if the specified extension is present.

Parameters

<i>name</i>	The name of the extension to check.
-------------	-------------------------------------

Returns

TRUE if the extension is present.

4.2.2.5 - (bool) moveSamples: (ALCsizei) numSamples toBuffer:(ALCvoid*) buffer

Move captured samples to the specified buffer.

This method will fail if less than the specified number of samples have been captured.

Parameters

<i>numSamples</i>	The number of samples to move.
<i>buffer</i>	the buffer to move the samples into.

Returns

TRUE if the operation was successful.

4.2.2.6 - (bool) startCapture

Start capturing samples.

Returns

TRUE if the operation was successful.

4.2.2.7 - (bool) stopCapture

Stop capturing samples.

Returns

TRUE if the operation was successful.

4.2.3 Property Documentation**4.2.3.1 - (int) captureSamples** [read], [nonatomic], [assign]

The number of capture samples available.

4.2.3.2 - (ALCdevice *) device [read], [nonatomic], [assign]

The OpenAL device pointer.

4.2.3.3 - (NSArray *) extensions [read], [nonatomic], [retain]

List of strings describing all extensions available on this device (NSString*).

4.2.3.4 - (int) majorVersion [read], [nonatomic], [assign]

The specification revision for this implementation (major version).

4.2.3.5 - (int) minorVersion [read], [nonatomic], [assign]

The specification revision for this implementation (minor version).

The documentation for this class was generated from the following files:

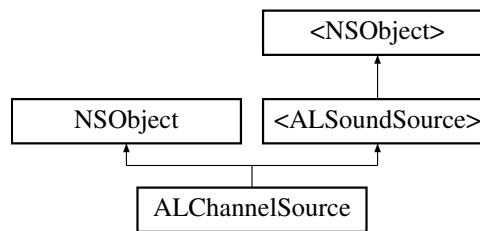
- ALCaptureDevice.h
- ALCaptureDevice.m

4.3 ALChannelSource Class Reference

A Sound source composed of other sources.

```
#import <ALChannelSource.h>
```

Inheritance diagram for ALChannelSource:



Instance Methods

- (id) - [initWithSources:](#)
Initialize a channel with a number of sources.
- (void) - [setDefaultFromSource:](#)
Set this channel's default values from those in the specified source.
- (void) - [resetToDefault](#)
Reset all sources in this channel to their default state.
- (void) - [addSource:](#)
Add a source to this channel.
- (id< [ALSoundSource](#) >) - [removeSource:](#)
Remove a source from the channel.
- ([ALChannelSource](#) *) - [splitChannelWithSources:](#)
Split the specified number of sources from this channel, creating a new channel.
- (void) - [addChannel:](#)
Absorb another channel's sources into this one.
- (NSArray *) - [clearUnusedBuffers](#)
Set all buffers in all non-playing sources to nil.
- (BOOL) - [removeBuffersNamed:](#)
Remove all instances of the specified buffer.

Class Methods

- (id) + [channelWithSources:](#)
Create a channel with a number of sources.

Protected Attributes

- bool [defaultsInitialized](#)
If YES, the defaults of this channel have been initialized.
- float **pitch**
- float **gain**
- float **maxDistance**
- float **rolloffFactor**
- float **referenceDistance**
- float **minGain**
- float **maxGain**
- float **coneOuterGain**
- float **coneInnerAngle**
- float **coneOuterAngle**
- float **reverbSendLevel**

- float **reverbOcclusion**
- float **reverbObstruction**
- [ALPoint](#) **position**
- [ALVector](#) **velocity**
- [ALVector](#) **direction**
- int **sourceRelative**
- int **sourceType**
- bool **looping**
- float [defaultPitch](#)
Default pitch.
- float [defaultGain](#)
Default gain.
- float [defaultMaxDistance](#)
Default max distance.
- float [defaultRolloffFactor](#)
Default rolloff factor.
- float [defaultReferenceDistance](#)
Default reference distance.
- float [defaultMinGain](#)
Default min gain.
- float [defaultMaxGain](#)
Default max gain.
- float [defaultConeOuterGain](#)
Default cone outer gain.
- float [defaultConeInnerAngle](#)
Default cone inner angle.
- float [defaultConeOuterAngle](#)
Default cone outer angle.
- [ALPoint](#) [defaultPosition](#)
Default position.
- [ALVector](#) [defaultVelocity](#)
Default velocity.
- [ALVector](#) [defaultDirection](#)
Default direction.
- int [defaultSourceRelative](#)
Default source relative.
- int [defaultSourceType](#)
Default source type.
- bool [defaultLooping](#)
Default looping.
- float [defaultReverbSendLevel](#)
Default reverb send level.
- float [defaultReverbOcclusion](#)
Default occlusion.
- float [defaultReverbObstruction](#)
Default obstruction.
- bool **interruptible**
- bool **muted**
- bool **paused**
- id [fadeCompleteTarget](#)
Target to inform when the current fade operation completes.

- SEL [fadeCompleteSelector](#)
Selector to call when the current fade operation completes.
- int [expectedFadeCallbackCount](#)
The expected number of sources that will callback when fading completes.
- int [currentFadeCallbackCount](#)
The actual number of sources that have called back.
- id [panCompleteTarget](#)
Target to inform when the current pan operation completes.
- SEL [panCompleteSelector](#)
Selector to call when the current pan operation completes.
- int [expectedPanCallbackCount](#)
The expected number of sources that will callback when panning completes.
- int [currentPanCallbackCount](#)
The actual number of sources that have called back.
- id [pitchCompleteTarget](#)
Target to inform when the current pitch operation completes.
- SEL [pitchCompleteSelector](#)
Selector to call when the current pitch operation completes.
- int [expectedPitchCallbackCount](#)
The expected number of sources that will callback when pitch op completes.
- int [currentPitchCallbackCount](#)
The actual number of sources that have called back.

Properties

- [ALContext](#) * [context](#)
This source's owning context.
- [ALSoundSourcePool](#) * [sourcePool](#)
Pool holding the actual sources.
- int [reservedSources](#)
The number of sources reserved by this channel.

4.3.1 Detailed Description

A Sound source composed of other sources.

Property values are applied to all sources within the channel.

Sounds will get played by any free sources within this channel.

If all sources are busy when playback is requested, it will attempt to interrupt a source to free it for playback.

4.3.2 Method Documentation

4.3.2.1 - (void) addChannel: (ALChannelSource*) channel

Absorb another channel's sources into this one.

All of the channel's sources will be moved into this channel.

Parameters

<i>channel</i>	The channel to absorb sources from.
----------------	-------------------------------------

4.3.2.2 - (void) addSource: (id<ALSoundSource>) source

Add a source to this channel.

Parameters

<i>source</i>	The source to add.
---------------	--------------------

4.3.2.3 + (id) channelWithSources: (int) reservedSources

Create a channel with a number of sources.

Parameters

<i>reservedSources</i>	the number of sources to reserve for this channel.
------------------------	--

Returns

A new channel.

4.3.2.4 - (NSArray *) clearUnusedBuffers

Set all buffers in all non-playing sources to nil.

Returns

A list of buffers that were cleared.

4.3.2.5 - (id) initWithSources: (int) reservedSources

Initialize a channel with a number of sources.

Parameters

<i>reservedSources</i>	the number of sources to reserve for this channel.
------------------------	--

Returns

The initialized channel.

4.3.2.6 - (BOOL) removeBuffersNamed: (NSString*) name

Remove all instances of the specified buffer.

Parameters

<i>name</i>	The name of the buffer.
-------------	-------------------------

Returns

NO if any of the matching buffers are currently being played.

4.3.2.7 - (id< **ALSoundSource** >) removeSource: (id< **ALSoundSource** >) *source*

Remove a source from the channel.

Parameters

<i>source</i>	The source to remove. If nil, remove any source.
---------------	--

Returns

The source that was removed.

4.3.2.8 - (void) resetToDefault

Reset all sources in this channel to their default state.

4.3.2.9 - (void) setDefaultsFromSource: (id< **ALSoundSource** >) *source*

Set this channel's default values from those in the specified source.

Parameters

<i>source</i>	the source to set default values from.
---------------	--

4.3.2.10 - (**ALChannelSource** *) splitChannelWithSources: (int) *numSources*

Split the specified number of sources from this channel, creating a new channel.

Parameters

<i>numSources</i>	The number of sources to split off
-------------------	------------------------------------

Returns

A new channel with the split-off sources.

4.3.3 Member Data Documentation

4.3.3.1 - (int) currentFadeCallbackCount [protected]

The actual number of sources that have called back.

4.3.3.2 - (int) currentPanCallbackCount [protected]

The actual number of sources that have called back.

4.3.3.3 - (int) currentPitchCallbackCount [protected]

The actual number of sources that have called back.

4.3.3.4 - (float) defaultConeInnerAngle [protected]

Default cone inner angle.

4.3.3.5 - (float) defaultConeOuterAngle [protected]

Default cone outer angle.

4.3.3.6 - (float) defaultConeOuterGain [protected]

Default cone outer gain.

4.3.3.7 - (ALVector) defaultDirection [protected]

Default direction.

4.3.3.8 - (float) defaultGain [protected]

Default gain.

4.3.3.9 - (bool) defaultLooping [protected]

Default looping.

4.3.3.10 - (float) defaultMaxDistance [protected]

Default max distance.

4.3.3.11 - (float) defaultMaxGain [protected]

Default max gain.

4.3.3.12 - (float) defaultMinGain [protected]

Default min gain.

4.3.3.13 - (float) defaultPitch [protected]

Default pitch.

4.3.3.14 - (ALPoint) defaultPosition [protected]

Default position.

4.3.3.15 - (float) defaultReferenceDistance [protected]

Default reference distance.

4.3.3.16 - (float) defaultReverbObstruction [protected]

Default obstruction.

4.3.3.17 - (float) defaultReverbOcclusion [protected]

Default occlusion.

4.3.3.18 - (float) defaultReverbSendLevel [protected]

Default reverb send level.

4.3.3.19 - (float) defaultRolloffFactor [protected]

Default rolloff factor.

4.3.3.20 - (bool) defaultsInitialized [protected]

If YES, the defaults of this channel have been initialized.

4.3.3.21 - (int) defaultSourceRelative [protected]

Default source relative.

4.3.3.22 - (int) defaultSourceType [protected]

Default source type.

4.3.3.23 - (ALVector) defaultVelocity [protected]

Default velocity.

4.3.3.24 - (int) expectedFadeCallbackCount [protected]

The expected number of sources that will callback when fading completes.

4.3.3.25 - (int) expectedPanCallbackCount [protected]

The expected number of sources that will callback when panning completes.

4.3.3.26 - (int) expectedPitchCallbackCount [protected]

The expected number of sources that will callback when pitch op completes.

4.3.3.27 - (SEL) fadeCompleteSelector [protected]

Selector to call when the current fade operation completes.

4.3.3.28 - (id) fadeCompleteTarget [protected]

Target to inform when the current fade operation completes.

4.3.3.29 - (SEL) panCompleteSelector [protected]

Selector to call when the current pan operation completes.

4.3.3.30 - (id) panCompleteTarget [protected]

Target to inform when the current pan operation completes.

4.3.3.31 - (SEL) pitchCompleteSelector [protected]

Selector to call when the current pitch operation completes.

4.3.3.32 - (id) pitchCompleteTarget [protected]

Target to inform when the current pitch operation completes.

4.3.4 Property Documentation

4.3.4.1 - (ALContext *) context [read], [nonatomic], [retain]

This source's owning context.

4.3.4.2 - (int) reservedSources [read], [write], [nonatomic], [assign]

The number of sources reserved by this channel.

4.3.4.3 - (ALSoundSourcePool *) sourcePool [read], [nonatomic], [retain]

Pool holding the actual sources.

All sources being used by this channel.

Do not modify!

The documentation for this class was generated from the following files:

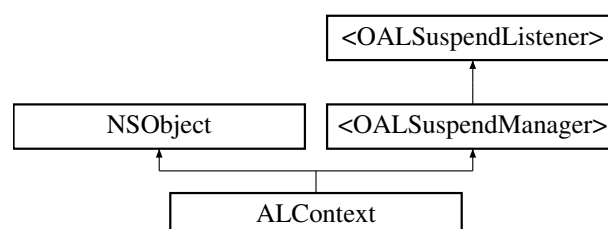
- ALChannelSource.h
- ALChannelSource.m

4.4 ALContext Class Reference

A context encompasses a single listener and a series of sources.

```
#import <ALContext.h>
```

Inheritance diagram for ALContext:



Instance Methods

- (id) - [initWithDevice:outputFrequency:refreshIntervals:synchronousContext:monoSources:stereoSources:](#)
Initialize this context on the specified device with attributes.
- (id) - [initWithDevice:attributes:](#)
Initialize this context for the specified device and attributes.
- (void) - [process](#)
Process this context.
- (void) - [stopAllSounds](#)
Stop all sound sources in this context.
- (void) - [clearBuffers](#)
Clear all buffers being used by sources in this context.
- (void) - [ensureContextIsCurrent](#)
Make sure this context is the current context.
- (bool) - [isExtensionPresent:](#)
Check if the specified extension is present in this context.
- (void *) - [getProcAddress:](#)
Get the address of the specified procedure (C function address).

Class Methods

- (id) + [contextOnDevice:attributes:](#)
Create a new context on the specified device.
- (id) + [contextOnDevice:outputFrequency:refreshIntervals:synchronousContext:monoSources:stereoSources:](#)
:
Create a new context on the specified device with attributes.

Protected Attributes

- NSMutableArray * [sources](#)
All sound sources associated with this context.
- bool **suspended**
- NSMutableArray * [attributes](#)
This context's attributes.
- OALSuspendHandler * [suspendHandler](#)
Handles suspending and interrupting for this object.

Properties

- NSString * [alVersion](#)
OpenAL version string in format "[spec major number]".
- NSArray * [attributes](#)
The current context's attribute list.
- ALCcontext * [context](#)
The OpenAL context pointer.
- ALDevice * [device](#)
The device this context was opened on.
- ALenum [distanceModel](#)
The current distance model.
- float [dopplerFactor](#)

- Exaggeration factor for Doppler effect.*
- NSArray * [extensions](#)
List of available extensions (NSString).*
- [ALListener](#) * [listener](#)
This context's listener.
- NSString * [renderer](#)
Information about the specific renderer.
- NSArray * [sources](#)
All sources associated with this context (ALSource).*
- float [speedOfSound](#)
Speed of sound in same units as velocities.
- NSString * [vendor](#)
Name of the vendor.

4.4.1 Detailed Description

A context encompasses a single listener and a series of sources.

A context is created from a device, and many contexts may be created (though multiple contexts would be unusual in an iOS app).

Note: Some property values are only valid if this context is the current context.

See Also

`ObjectAL.currentContext`

4.4.2 Method Documentation

4.4.2.1 - (void) clearBuffers

Clear all buffers being used by sources in this context.

4.4.2.2 + (id) contextOnDevice: (ALDevice *) device attributes:(NSArray*) attributes

Create a new context on the specified device.

Parameters

<i>device</i>	The device to open the context on.
<i>attributes</i>	An array of NSNumber in ordered pairs (attribute id followed by integer value). Possible attributes: ALC_FREQUENCY, ALC_REFRESH, ALC_SYNC, ALC_MONO_SOURCES, ALC_STEREO_SOURCES

Returns

A new context.

4.4.2.3 + (id) contextOnDevice: (ALDevice*) device outputFrequency:(int) outputFrequency refreshIntervals:(int) refreshIntervals synchronousContext:(bool) synchronousContext monoSources:(int) monoSources stereoSources:(int) stereoSources

Create a new context on the specified device with attributes.

Parameters

<i>device</i>	The device to open the context on.
<i>outputFrequency</i>	The frequency to mix all sources to before outputting (ignored by iOS).
<i>refreshIntervals</i>	The number of passes per second used to mix the audio sources. For games this can be 5-15. For audio intensive apps, it should be higher (ignored by iOS).
<i>synchronous-Context</i>	If true, this context runs on the main thread and depends on you calling alcUpdateContext (ignored by iOS).
<i>monoSources</i>	A hint indicating how many sources should support mono (default 28 on iOS).
<i>stereoSources</i>	A hint indicating how many sources should support stereo (default 4 on iOS).

Returns

A new context.

4.4.2.4 - (void) ensureContextIsCurrent

Make sure this context is the current context.

This method is used to work around iOS 4.0 and 4.2 bugs that could cause the context to be lost.

4.4.2.5 - (void *) getProcAddress: (NSString*) functionName

Get the address of the specified procedure (C function address).

Only valid when this is the current context.

Note: The OpenAL implementation is free to return a pointer even if it is not valid for this context. Always call isExtensionPresent first.

Parameters

<i>functionName</i>	the name of the procedure to get.
---------------------	-----------------------------------

Returns

the procedure's address, or NULL if it wasn't found.

4.4.2.6 - (id) initWithDevice: (ALDevice *) device attributes:(NSArray*) attributes

Initialize this context for the specified device and attributes.

Parameters

<i>device</i>	The device to open the context on.
<i>attributes</i>	An array of NSNumber in ordered pairs (attribute id followed by integer value). Possible attributes: ALC_FREQUENCY, ALC_REFRESH, ALC_SYNC, ALC_MONO_SOURCES, ALC_STEREO_SOURCES

Returns

The initialized context.

4.4.2.7 - (id) initOnDevice: (ALDevice*) *device* outputFrequency:(int) *outputFrequency* refreshIntervals:(int) *refreshIntervals* synchronousContext:(bool) *synchronousContext* monoSources:(int) *monoSources* stereoSources:(int) *stereoSources*

Initialize this context on the specified device with attributes.

Parameters

<i>device</i>	The device to open the context on.
<i>outputFrequency</i>	The frequency to mix all sources to before outputting (ignored by iOS).
<i>refreshIntervals</i>	The number of passes per second used to mix the audio sources. For games this can be 5-15. For audio intensive apps, it should be higher (ignored by iOS).
<i>synchronous-Context</i>	If true, this context runs on the main thread and depends on you calling alcUpdateContext (ignored by iOS).
<i>monoSources</i>	A hint indicating how many sources should support mono (default 28 on iOS).
<i>stereoSources</i>	A hint indicating how many sources should support stereo (default 4 on iOS).

Returns

The initialized context.

4.4.2.8 - (bool) isExtensionPresent: (NSString*) *name*

Check if the specified extension is present in this context.

Only valid when this is the current context.

Parameters

<i>name</i>	The name of the extension to check.
-------------	-------------------------------------

Returns

TRUE if the extension is present in this context.

4.4.2.9 - (void) process

Process this context.

4.4.2.10 - (void) stopAllSounds

Stop all sound sources in this context.

4.4.3 Member Data Documentation

4.4.3.1 - (NSMutableArray*) *attributes* [protected]

This context's attributes.

4.4.3.2 - (NSMutableArray*) sources [protected]

All sound sources associated with this context.

4.4.3.3 - (OALSuspendHandler*) suspendHandler [protected]

Handles suspending and interrupting for this object.

4.4.4 Property Documentation

4.4.4.1 - (NSString *) alVersion [read], [nonatomic], [retain]

OpenAL version string in format “[spec major number].

[spec minor number] [optional vendor version information]” Only valid when this is the current context.

4.4.4.2 - (NSArray*) attributes [read], [nonatomic], [retain]

The current context’s attribute list.

Only valid when this is the current context.

4.4.4.3 - (ALCcontext *) context [read], [nonatomic], [assign]

The OpenAL context pointer.

4.4.4.4 - (ALDevice*) device [read], [nonatomic], [retain]

The device this context was opened on.

4.4.4.5 - (ALenum) distanceModel [read], [write], [nonatomic], [assign]

The current distance model.

Legal values are AL_NONE, AL_INVERSE_DISTANCE, AL_INVERSE_DISTANCE_CLAMPED, AL_LINEAR_DISTANCE, AL_LINEAR_DISTANCE_CLAMPED, AL_EXPONENT_DISTANCE, and AL_EXPONENT_DISTANCE_CLAMPED. See the OpenAL spec for detailed information.

Only valid when this is the current context.

4.4.4.6 - (float) dopplerFactor [read], [write], [nonatomic], [assign]

Exaggeration factor for Doppler effect.

Only valid when this is the current context.

4.4.4.7 - (NSArray *) extensions [read], [nonatomic], [retain]

List of available extensions (NSString*).

Only valid when this is the current context.

4.4.4.8 `-(ALListener *) listener` `[read],[nonatomic],[retain]`

This context's listener.

4.4.4.9 `-(NSString *) renderer` `[read],[nonatomic],[retain]`

Information about the specific renderer.

Only valid when this is the current context.

4.4.4.10 `-(NSArray *) sources` `[read],[nonatomic],[retain]`

All sources associated with this context (ALSource*).

4.4.4.11 `-(float) speedOfSound` `[read],[write],[nonatomic],[assign]`

Speed of sound in same units as velocities.

Only valid when this is the current context.

4.4.4.12 `-(NSString *) vendor` `[read],[nonatomic],[retain]`

Name of the vendor.

Only valid when this is the current context.

The documentation for this class was generated from the following files:

- ALContext.h
- ALContext.m

4.5 ALContext() Category Reference

Properties

- [ALDevice](#) * **device**

The documentation for this category was generated from the following file:

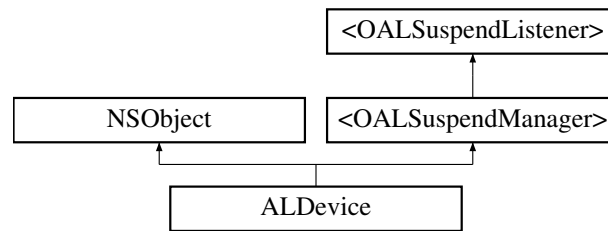
- ALContext.m

4.6 ALDevice Class Reference

A device is a logical mapping to an audio device through the OpenAL implementation.

```
#import <ALDevice.h>
```

Inheritance diagram for ALDevice:



Instance Methods

- (id) - [initWithDeviceSpecifier:](#)
Initialize with the specified device.
- (bool) - [isExtensionPresent:](#)
Check if the specified extension is present.
- (void *) - [getProcAddress:](#)
Get the address of the specified procedure (C function address).
- (void) - [clearBuffers](#)
Clear all buffers being used by sources of contexts opened on this device.

Class Methods

- (id) + [deviceWithDeviceSpecifier:](#)
Open the specified device.

Protected Attributes

- NSMutableArray * [contexts](#)
All contexts opened from this device.
- OALSuspendHandler * [suspendHandler](#)
Handles suspending and interrupting for this object.

Properties

- NSArray * [contexts](#)
All contexts created on this device (ALContext).*
- ALCdevice * [device](#)
The OpenAL device pointer.
- NSArray * [extensions](#)
List of strings describing all extensions available on this device (NSString).*
- int [majorVersion](#)
The specification revision for this implementation (major version).
- int [minorVersion](#)
The specification revision for this implementation (minor version).

4.6.1 Detailed Description

A device is a logical mapping to an audio device through the OpenAL implementation.

4.6.2 Method Documentation

4.6.2.1 - (void) clearBuffers

Clear all buffers being used by sources of contexts opened on this device.

4.6.2.2 + (id) deviceWithDeviceSpecifier: (NSString*) *deviceSpecifier*

Open the specified device.

Parameters

<i>deviceSpecifier</i>	The device to open (nil = default device).
------------------------	--

Returns

A new device.

4.6.2.3 - (void *) getProcAddress: (NSString*) *functionName*

Get the address of the specified procedure (C function address).

Parameters

<i>functionName</i>	the name of the procedure to get.
---------------------	-----------------------------------

Returns

the procedure's address, or NULL if it wasn't found.

4.6.2.4 - (id) initWithDeviceSpecifier: (NSString*) *deviceSpecifier*

Initialize with the specified device.

Parameters

<i>deviceSpecifier</i>	The device to open (nil = default device).
------------------------	--

Returns

the initialized device.

4.6.2.5 - (bool) isExtensionPresent: (NSString*) *name*

Check if the specified extension is present.

Parameters

<i>name</i>	The extension to check.
-------------	-------------------------

Returns

TRUE if the extension is present.

4.6.3 Member Data Documentation

4.6.3.1 `-(NSMutableArray*) contexts` `[protected]`

All contexts opened from this device.

4.6.3.2 `-(OALSuspendHandler*) suspendHandler` `[protected]`

Handles suspending and interrupting for this object.

4.6.4 Property Documentation

4.6.4.1 `-(NSArray*) contexts` `[read]`, `[nonatomic]`, `[retain]`

All contexts created on this device (`ALContext*`).

4.6.4.2 `-(ALCdevice *) device` `[read]`, `[nonatomic]`, `[assign]`

The OpenAL device pointer.

4.6.4.3 `-(NSArray *) extensions` `[read]`, `[nonatomic]`, `[retain]`

List of strings describing all extensions available on this device (`NSString*`).

4.6.4.4 `-(int) majorVersion` `[read]`, `[nonatomic]`, `[assign]`

The specification revision for this implementation (major version).

4.6.4.5 `-(int) minorVersion` `[read]`, `[nonatomic]`, `[assign]`

The specification revision for this implementation (minor version).

The documentation for this class was generated from the following files:

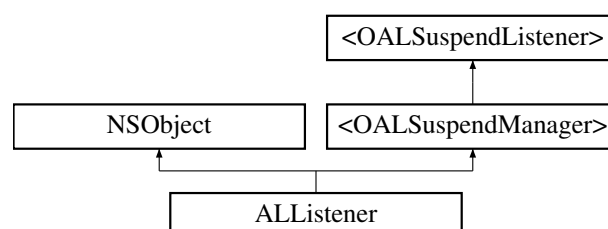
- `ALDevice.h`
- `ALDevice.m`

4.7 ALListener Class Reference

The listener represents the user who is listening to sounds in 3D space.

```
#import <ALListener.h>
```

Inheritance diagram for `ALListener`:



Protected Attributes

- [OALSuspendHandler](#) * [suspendHandler](#)
Handles suspending and interrupting for this object.

Properties

- [ALContext](#) * [context](#)
The context this listener belongs to (WEAK reference).
- bool [muted](#)
Causes this listener to stop hearing sound.
- float [gain](#)
Gain (volume), affecting every sound this listener hears (0.0 = no sound, 1.0 = max volume).
- [ALOrientation](#) [orientation](#)
Orientation (up: x, y, z, at: x, y, z).
- [ALPoint](#) [position](#)
Position (x, y, z).
- [ALVector](#) [velocity](#)
Velocity (x, y, z).
- bool [reverbOn](#)
Turns on reverb.
- float [globalReverbLevel](#)
The global reverb level (from -40.0db to 40.0db).
- int [reverbRoomType](#)
The room type to simulate for reverb.
- float [reverbEQGain](#)
The equalizer gain for reverb.
- float [reverbEQBandwidth](#)
The equalizer bandwidth for reverb.
- float [reverbEQFrequency](#)
The equalizer frequency for reverb.

Additional Inherited Members

4.7.1 Detailed Description

The listener represents the user who is listening to sounds in 3D space.

This object controls his position, orientation, and velocity, as well as providing a master gain.

A context contains one and only one listener.

4.7.2 Member Data Documentation

4.7.2.1 -(OALSuspendHandler*) suspendHandler [protected]

Handles suspending and interrupting for this object.

4.7.3 Property Documentation

4.7.3.1 -(ALContext*) context [read],[nonatomic],[assign]

The context this listener belongs to (WEAK reference).

4.7.3.2 `-(float) gain` `[read], [write], [nonatomic], [assign]`

Gain (volume), affecting every sound this listener hears (0.0 = no sound, 1.0 = max volume).

Only valid if this listener's context is the current context.

4.7.3.3 `-(float) globalReverbLevel` `[read], [write], [nonatomic], [assign]`

The global reverb level (from -40.0db to 40.0db).

(iOS 5.0+)

4.7.3.4 `-(bool) muted` `[read], [write], [nonatomic], [assign]`

Causes this listener to stop hearing sound.

It's called "muted" rather than "deaf" to give a consistent name with other mute functions.

4.7.3.5 `-(ALOrientation) orientation` `[read], [write], [nonatomic], [assign]`

Orientation (up: x, y, z, at: x, y, z).

Only valid if this listener's context is the current context.

4.7.3.6 `-(ALPoint) position` `[read], [write], [nonatomic], [assign]`

Position (x, y, z).

Only valid if this listener's context is the current context.

4.7.3.7 `-(float) reverbEQBandwidth` `[read], [write], [nonatomic], [assign]`

The equalizer bandwidth for reverb.

(iOS 5.0+)

4.7.3.8 `-(float) reverbEQFrequency` `[read], [write], [nonatomic], [assign]`

The equalizer frequency for reverb.

(iOS 5.0+)

4.7.3.9 `-(float) reverbEQGain` `[read], [write], [nonatomic], [assign]`

The equalizer gain for reverb.

(iOS 5.0+)

4.7.3.10 `-(bool) reverbOn` `[read], [write], [nonatomic], [assign]`

Turns on reverb.

(iOS 5.0+)

4.7.3.11 - (int) `reverbRoomType` [read],[write],[nonatomic],[assign]

The room type to simulate for reverb.

(iOS 5.0+)

Allowed room types:

ALC_ASA_REVERB_ROOM_TYPE_SmallRoom ALC_ASA_REVERB_ROOM_TYPE_MediumRoom ALC_ASA_REVERB_ROOM_TYPE_LargeRoom ALC_ASA_REVERB_ROOM_TYPE_MediumHall ALC_ASA_REVERB_ROOM_TYPE_LargeHall ALC_ASA_REVERB_ROOM_TYPE_Plate ALC_ASA_REVERB_ROOM_TYPE_MediumChamber ALC_ASA_REVERB_ROOM_TYPE_LargeChamber ALC_ASA_REVERB_ROOM_TYPE_Cathedral ALC_ASA_REVERB_ROOM_TYPE_LargeRoom2 ALC_ASA_REVERB_ROOM_TYPE_MediumHall2 ALC_ASA_REVERB_ROOM_TYPE_MediumHall3 ALC_ASA_REVERB_ROOM_TYPE_LargeHall2

4.7.3.12 - (ALVector) `velocity` [read],[write],[nonatomic],[assign]

Velocity (x, y, z).

Only valid if this listener's context is the current context.

The documentation for this class was generated from the following files:

- ALLlistener.h
- ALLlistener.m

4.8 ALOrientation Struct Reference

Represents an orientation, consisting of an "at" vector (representing the "forward" direction), and the "up" vector (representing "up" for the subject).

```
#include <ALTypes.h>
```

Public Attributes

- [ALVector at](#)
The "at" vector, representing "forward".
- [ALVector up](#)
The "up" vector, representing "up".

4.8.1 Detailed Description

Represents an orientation, consisting of an "at" vector (representing the "forward" direction), and the "up" vector (representing "up" for the subject).

4.8.2 Member Data Documentation

4.8.2.1 ALVector ALOrientation::at

The "at" vector, representing "forward".

4.8.2.2 ALVector ALOrientation::up

The "up" vector, representing "up".

The documentation for this struct was generated from the following file:

- ALTypes.h

4.9 ALPoint Struct Reference

Represents a 3-dimensional point for certain ObjectAL properties.

```
#include <ALTypes.h>
```

Public Attributes

- float [x](#)
The "X" coordinate.
- float [y](#)
The "Y" coordinate.
- float [z](#)
The "Z" coordinate.

4.9.1 Detailed Description

Represents a 3-dimensional point for certain ObjectAL properties.

4.9.2 Member Data Documentation

4.9.2.1 float ALPoint::x

The "X" coordinate.

4.9.2.2 float ALPoint::y

The "Y" coordinate.

4.9.2.3 float ALPoint::z

The "Z" coordinate.

The documentation for this struct was generated from the following file:

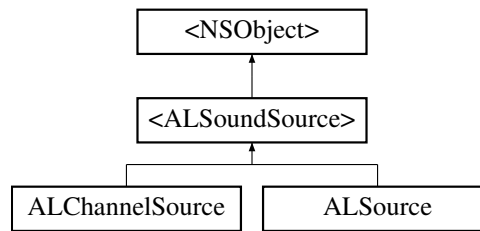
- ALTypes.h

4.10 <ALSoundSource> Protocol Reference

Manages all properties relating to an OpenAL sound source.

```
#import <ALSoundSource.h>
```

Inheritance diagram for <ALSoundSource>:



Instance Methods

- (id< [ALSoundSource](#) >) - [play:](#)
Play a sound.
- (id< [ALSoundSource](#) >) - [play:loop:](#)
Play a sound, optionally looping.
- (id< [ALSoundSource](#) >) - [play:gain:pitch:pan:loop:](#)
Play a sound, setting gain, pitch, pan, and looping.
- (void) - [stop](#)
Stop playing the current sound.
- (void) - [rewind](#)
Stop playing the current sound and set its state to `AL_INITIAL`.
- (void) - [fadeTo:duration:target:selector:](#)
Fade to the specified gain value.
- (void) - [stopFade](#)
Stop the currently running fade operation, if any.
- (void) - [panTo:duration:target:selector:](#)
pan to the specified value.
- (void) - [stopPan](#)
Stop the currently running pan operation, if any.
- (void) - [pitchTo:duration:target:selector:](#)
Gradually change pitch to the specified value.
- (void) - [stopPitch](#)
Stop the currently running pitch operation, if any.
- (void) - [stopActions](#)
Stop any currently running fade, pan, or pitch operations.
- (void) - [clear](#)
Clear any buffers this source is currently using.

Properties

- float [coneInnerAngle](#)
Cone inner angle (OpenAL property).
- float [coneOuterAngle](#)
Cone outer angle (OpenAL property).
- float [coneOuterGain](#)
Cone outer gain (OpenAL property).
- [ALVector](#) [direction](#)
Direction (OpenAL property).
- float [gain](#)
Gain (volume) (OpenAL property).
- float [volume](#)

- Volume (alias to gain).*
- bool `interruptible`
 - If true, this source may be interrupted when resources are low.*
- bool `looping`
 - Looping (OpenAL property).*
- float `maxDistance`
 - Max distance (OpenAL property).*
- float `maxGain`
 - Max gain (OpenAL property).*
- float `minGain`
 - Min gain (OpenAL property).*
- bool `muted`
 - If true, this source is muted.*
- bool `paused`
 - If true, this source is currently paused.*
- float `pitch`
 - Pitch (OpenAL property).*
- bool `playing`
 - If true, this source is currently playing audio.*
- `ALPoint` `position`
 - Position (OpenAL property).*
- float `referenceDistance`
 - Reference distance (OpenAL property).*
- float `rolloffFactor`
 - Rolloff factor (OpenAL property).*
- int `sourceRelative`
 - Source relative (OpenAL property).*
- int `sourceType`
 - Source type (OpenAL property).*
- `ALVector` `velocity`
 - Velocity (OpenAL property).*
- float `pan`
 - Pan value (-1.0 = far left, 1.0 = far right).*
- float `reverbSendLevel`
 - Reverb send level (how much reverb affects this source).*
- float `reverbOcclusion`
 - Reverb occlusion (wall/door between listener and source).*
- float `reverbObstruction`
 - Reverb obstruction (object between listener and source).*

4.10.1 Detailed Description

Manages all properties relating to an OpenAL sound source.

There are currently two classes that adhere to this protocol: `ALSource` and `ChannelSource` (which collectively manipulates a set of `ALSource` objects). A full description of the properties themselves is available in the OpenAL 1.1 Specification and Reference: <http://connect.creativelabs.com/openal/Documentation>

4.10.2 Method Documentation

4.10.2.1 - (void) clear

Clear any buffers this source is currently using.

4.10.2.2 - (void) fadeTo: (float) *gain* duration:(float) *duration* target:(id) *target* selector:(SEL) *selector*

Fade to the specified gain value.

Parameters

<i>gain</i>	The gain to fade to.
<i>duration</i>	The duration of the fade operation in seconds.
<i>target</i>	The target to notify when the fade completes (can be nil).
<i>selector</i>	The selector to call when the fade completes. The selector must accept a single parameter, which will be the object that performed the fade.

4.10.2.3 - (void) panTo: (float) *pan* duration:(float) *duration* target:(id) *target* selector:(SEL) *selector*

pan to the specified value.

Parameters

<i>pan</i>	The value to pan to.
<i>duration</i>	The duration of the pan operation in seconds.
<i>target</i>	The target to notify when the pan completes (can be nil).
<i>selector</i>	The selector to call when the pan completes. The selector must accept a single parameter, which will be the object that performed the pan.

4.10.2.4 - (void) pitchTo: (float) *pitch* duration:(float) *duration* target:(id) *target* selector:(SEL) *selector*

Gradually change pitch to the specified value.

Parameters

<i>pitch</i>	The value to change pitch to.
<i>duration</i>	The duration of the pitch operation in seconds.
<i>target</i>	The target to notify when the pitch change completes (can be nil).
<i>selector</i>	The selector to call when the pitch change completes. The selector must accept a single parameter, which will be the object that performed the pitch change.

4.10.2.5 - (id<ALSoundSource>) play: (ALBuffer *) *buffer*

Play a sound.

Parameters

<i>buffer</i>	the buffer to play.
---------------	---------------------

Returns

the source playing the sound, or nil if the sound could not be played.

4.10.2.6 - (id<ALSoundSource>) play: (ALBuffer *) *buffer* gain:(float) *gain* pitch:(float) *pitch* pan:(float) *pan* loop:(bool) *loop*

Play a sound, setting gain, pitch, pan, and looping.

Parameters

<i>buffer</i>	the buffer to play.
<i>gain</i>	The gain (volume) to play at (0.0 - 1.0).
<i>pitch</i>	The pitch to play at (1.0 = normal pitch).
<i>pan</i>	Left-right panning (-1.0 = far left, 1.0 = far right).
<i>loop</i>	If TRUE, the sound will loop until you call "stop" on the returned sound source.

Returns

the source playing the sound, or nil if the sound could not be played.

4.10.2.7 - (id<ALSoundSource>) play: (ALBuffer *) *buffer* loop:(bool) *loop*

Play a sound, optionally looping.

Parameters

<i>buffer</i>	the buffer to play.
<i>loop</i>	If TRUE, the sound will loop until you call "stop" on the returned sound source.

Returns

the source playing the sound, or nil if the sound could not be played.

4.10.2.8 - (void) rewind

Stop playing the current sound and set its state to AL_INITIAL.

4.10.2.9 - (void) stop

Stop playing the current sound.

4.10.2.10 - (void) stopActions

Stop any currently running fade, pan, or pitch operations.

4.10.2.11 - (void) stopFade

Stop the currently running fade operation, if any.

4.10.2.12 - (void) stopPan

Stop the currently running pan operation, if any.

4.10.2.13 - (void) stopPitch

Stop the currently running pitch operation, if any.

4.10.3 Property Documentation

4.10.3.1 - (float) **coneInnerAngle** [read], [write], [nonatomic], [assign]

Cone inner angle (OpenAL property).

4.10.3.2 - (float) **coneOuterAngle** [read], [write], [nonatomic], [assign]

Cone outer angle (OpenAL property).

4.10.3.3 - (float) **coneOuterGain** [read], [write], [nonatomic], [assign]

Cone outer gain (OpenAL property).

4.10.3.4 - (ALVector) **direction** [read], [write], [nonatomic], [assign]

Direction (OpenAL property).

4.10.3.5 - (float) **gain** [read], [write], [nonatomic], [assign]

Gain (volume) (OpenAL property).

4.10.3.6 - (bool) **interruptible** [read], [write], [nonatomic], [assign]

If true, this source may be interrupted when resources are low.

4.10.3.7 - (bool) **looping** [read], [write], [nonatomic], [assign]

Looping (OpenAL property).

4.10.3.8 - (float) **maxDistance** [read], [write], [nonatomic], [assign]

Max distance (OpenAL property).

4.10.3.9 - (float) **maxGain** [read], [write], [nonatomic], [assign]

Max gain (OpenAL property).

4.10.3.10 - (float) **minGain** [read], [write], [nonatomic], [assign]

Min gain (OpenAL property).

4.10.3.11 - (bool) **muted** [read], [write], [nonatomic], [assign]

If true, this source is muted.

4.10.3.12 - (float) pan [read], [write], [nonatomic], [assign]

Pan value (-1.0 = far left, 1.0 = far right).

Note: This effect is simulated by changing the source's X position. Do not use this property if you are modifying the position property as well.

4.10.3.13 - (bool) paused [read], [write], [nonatomic], [assign]

If true, this source is currently paused.

4.10.3.14 - (float) pitch [read], [write], [nonatomic], [assign]

Pitch (OpenAL property).

4.10.3.15 - (bool) playing [read], [nonatomic], [assign]

If true, this source is currently playing audio.

4.10.3.16 - (ALPoint) position [read], [write], [nonatomic], [assign]

Position (OpenAL property).

4.10.3.17 - (float) referenceDistance [read], [write], [nonatomic], [assign]

Reference distance (OpenAL property).

4.10.3.18 - (float) reverbObstruction [read], [write], [nonatomic], [assign]

Reverb obstruction (object between listener and source).

(iOS 5.0+) -100.0db (most obstruction) to 0.0 (no obstruction). Default 0.

4.10.3.19 - (float) reverbOcclusion [read], [write], [nonatomic], [assign]

Reverb occlusion (wall/door between listener and source).

(iOS 5.0+) -100.0db (most occlusion) to 0.0 (no occlusion). Default 0.

4.10.3.20 - (float) reverbSendLevel [read], [write], [nonatomic], [assign]

Reverb send level (how much reverb affects this source).

(iOS 5.0+) 0.0 = fully dry, 1.0 = fully wet. Default 0.

4.10.3.21 - (float) rolloffFactor [read], [write], [nonatomic], [assign]

Rolloff factor (OpenAL property).

4.10.3.22 - (int) sourceRelative [read], [write], [nonatomic], [assign]

Source relative (OpenAL property).

4.10.3.23 - (int) **sourceType** [read],[nonatomic],[assign]

Source type (OpenAL property).

4.10.3.24 - (ALVector) **velocity** [read],[write],[nonatomic],[assign]

Velocity (OpenAL property).

4.10.3.25 - (float) **volume** [read],[write],[nonatomic],[assign]

Volume (alias to gain).

The documentation for this protocol was generated from the following file:

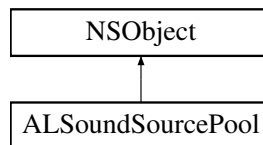
- `ALSoundSource.h`

4.11 ALSoundSourcePool Class Reference

A pool of sound sources, which can be fetched based on availability.

```
#import <ALSoundSourcePool.h>
```

Inheritance diagram for ALSoundSourcePool:



Instance Methods

- (void) - **addSource:**
Add a source to this pool.
- (void) - **removeSource:**
Remove a source from this pool.
- (id< **ALSoundSource** >) - **getFreeSource:**
Acquire a free or freeable source from this pool.

Class Methods

- (id) + **pool**
Make a new pool.

Protected Attributes

- NSMutableArray * **sources**
All sources managed by this pool (id<ALSoundSource>).

Properties

- NSArray * [sources](#)

All sources managed by this pool (id<ALSoundSource>).

4.11.1 Detailed Description

A pool of sound sources, which can be fetched based on availability.

4.11.2 Method Documentation

4.11.2.1 - (void) addSource: (id<ALSoundSource>) source

Add a source to this pool.

Parameters

<i>source</i>	The source to add.
---------------	--------------------

4.11.2.2 - (id<ALSoundSource>) getFreeSource: (bool) attemptToInterrupt

Acquire a free or freeable source from this pool.

It first attempts to find a completely free source. Failing this, it will attempt to interrupt a source and return that (if attemptToInterrupt is TRUE).

Parameters

<i>attemptToInterrupt</i>	If TRUE, attempt to interrupt sources to free them for use.
---------------------------	---

Returns

The freed sound source, or nil if no sources are freeable.

4.11.2.3 + (id) pool

Make a new pool.

Returns

A new pool.

4.11.2.4 - (void) removeSource: (id<ALSoundSource>) source

Remove a source from this pool.

Parameters

<i>source</i>	The source to remove.
---------------	-----------------------

4.11.3 Member Data Documentation

4.11.3.1 - (NSMutableArray*) sources [protected]

All sources managed by this pool (id<ALSoundSource>).

4.11.4 Property Documentation

4.11.4.1 - (NSArray*) sources [read], [nonatomic], [retain]

All sources managed by this pool (id<ALSoundSource>).

The documentation for this class was generated from the following files:

- ALSoundSourcePool.h
- ALSoundSourcePool.m

4.12 ALSoundSourcePool(Private) Category Reference

Private interface to SoundSourcePool.

Instance Methods

- (void) - [moveToHead:](#)
Move a source to the head of the list.

4.12.1 Detailed Description

Private interface to SoundSourcePool.

4.12.2 Method Documentation

4.12.2.1 - (void) moveToHead: (int) index

Move a source to the head of the list.

Parameters

<i>index</i>	the index of the source to move.
--------------	----------------------------------

The documentation for this category was generated from the following file:

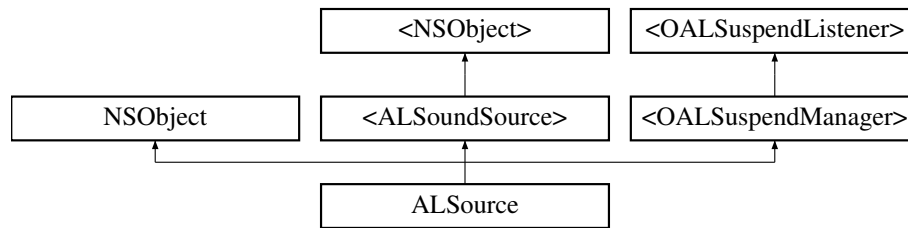
- ALSoundSourcePool.m

4.13 ALSource Class Reference

A source represents an object that emits sound which can be heard by a listener.

```
#import <ALSource.h>
```

Inheritance diagram for ALSource:



Instance Methods

- (id) - `initWithContext:`
Initialize a new source on the specified context.
- (id< `ALSoundSource` >) - `play`
Play the currently attached buffer.
- (bool) - `queueBuffer:`
Add a buffer to the buffer queue.
- (bool) - `queueBuffer:repeats:`
Add a buffer to the buffer queue, repeating it multiple times.
- (bool) - `queueBuffers:`
Add buffers to the buffer queue.
- (bool) - `queueBuffers:repeats:`
Add buffers to the buffer queue, repeating it multiple times.
- (bool) - `unqueueBuffer:`
Remove a buffer from the buffer queue.
- (bool) - `unqueueBuffers:`
Remove buffers from the buffer queue.
- (void) - `registerNotification:callback:userData:`
Register to receive notifications about an event on this source.
- (void) - `unregisterNotification:`
Unregister notifications for a notification type on this source.
- (void) - `unregisterAllNotifications`
Unregister all notifications for this source.

Class Methods

- (id) + `source`
Create a new source.
- (id) + `sourceOnContext:`
Create a new source on the specified context.

Protected Attributes

- bool `interruptible`
- float `gain`
- bool `muted`
- int `shadowState`
Shadow value which keeps the correct state value for `AL_PLAYING` and `AL_PAUSED`.
- bool `abortPlaybackResume`
Used to abort a pending playback resume if the user calls stop or pause.
- `OALAction` * `gainAction`

- Current action operating on the gain control.*
- [OALAction](#) * [panAction](#)
Current action operating on the pan control.
- [OALAction](#) * [pitchAction](#)
Current action operating on the pitch control.
- [OALSuspendHandler](#) * [suspendHandler](#)
Handles suspending and interrupting for this object.

Properties

- [ALBuffer](#) * [buffer](#)
The sound buffer this source is attached to (set to nil to detach the currently attached buffer).
- int [buffersQueued](#)
How many buffers this source has queued.
- int [buffersProcessed](#)
How many of these buffers have been processed during playback.
- [ALContext](#) * [context](#)
The context this source was opened on.
- float [offsetInBytes](#)
The offset into the current buffer (in bytes).
- float [offsetInSamples](#)
The offset into the current buffer (in samples).
- float [offsetInSeconds](#)
The offset into the current buffer (in seconds).
- ALuint [sourceId](#)
OpenAL's ID for this source.
- int [state](#)
The state of this source.

4.13.1 Detailed Description

A source represents an object that emits sound which can be heard by a listener. This source can have position, velocity, and direction.

4.13.2 Method Documentation

4.13.2.1 - (id) initOnContext: ([ALContext](#)*) *context*

Initialize a new source on the specified context.

Parameters

<i>context</i>	the context to create the source on.
----------------	--------------------------------------

Returns

A new source.

4.13.2.2 - (id< [ALSoundSource](#) >) play

Play the currently attached buffer.

Returns

the source playing the sound, or nil if the sound could not be played.

4.13.2.3 - (bool) queueBuffer: (ALBuffer*) buffer

Add a buffer to the buffer queue.

Parameters

<i>buffer</i>	the buffer to add to the queue.
---------------	---------------------------------

Returns

TRUE if the operation was successful.

4.13.2.4 - (bool) queueBuffer: (ALBuffer*) buffer repeats:(NSUInteger) repeats

Add a buffer to the buffer queue, repeating it multiple times.

Parameters

<i>buffer</i>	the buffer to add to the queue.
<i>repeats</i>	the number of times to repeat the buffer in the queue.

Returns

TRUE if the operation was successful.

4.13.2.5 - (bool) queueBuffers: (NSArray*) buffers

Add buffers to the buffer queue.

Parameters

<i>buffers</i>	the buffers to add to the queue.
----------------	----------------------------------

Returns

TRUE if the operation was successful.

4.13.2.6 - (bool) queueBuffers: (NSArray*) buffers repeats:(NSUInteger) repeats

Add buffers to the buffer queue, repeating it multiple times.

The buffers will be played in order, repeating the specified number of times.

Parameters

<i>buffers</i>	the buffers to add to the queue.
<i>repeats</i>	the number of times to repeat the buffer in the queue.

Returns

TRUE if the operation was successful.

4.13.2.7 - (void) registerNotification: (ALuint) *notificationID* callback:(OALSourceNotificationCallback) *callback* userData:(void*) *userData*

Register to receive notifications about an event on this source.

(iOS 5.0+)

The following notification types are recognized: AL_SOURCE_STATE - Sent when a source's state changes. AL_BUFFERS_PROCESSED - Sent when all buffers have been processed. AL_QUEUE_HAS_LOOPED - Sent when a looping source has looped to it's start point.

Parameters

<i>notificationID</i>	The kind of notification to be informed of (see above).
<i>callback</i>	The block to call for notification.
<i>userData</i>	a pointer that will be passed to the callback.

4.13.2.8 + (id) source

Create a new source.

Returns

A new source.

4.13.2.9 + (id) sourceOnContext: (ALContext*) *context*

Create a new source on the specified context.

Parameters

<i>context</i>	the context to create the source on.
----------------	--------------------------------------

Returns

A new source.

4.13.2.10 - (bool) unqueueBuffer: (ALBuffer*) *buffer*

Remove a buffer from the buffer queue.

Parameters

<i>buffer</i>	the buffer to remove from the queue.
---------------	--------------------------------------

Returns

TRUE if the operation was successful.

4.13.2.11 - (bool) `unqueueBuffers:` (NSArray*) *buffers*

Remove buffers from the buffer queue.

Parameters

<i>buffers</i>	the buffers to remove from the queue.
----------------	---------------------------------------

Returns

TRUE if the operation was successful.

4.13.2.12 - (void) `unregisterAllNotifications`

Unregister all notifications for this source.

(iOS 5.0+)

4.13.2.13 - (void) `unregisterNotification:` (ALuint) *notificationID*

Unregister notifications for a notification type on this source.

(iOS 5.0+)

Parameters

<i>notificationID</i>	The kind of notification to remove.
-----------------------	-------------------------------------

4.13.3 Member Data Documentation

4.13.3.1 - (bool) `abortPlaybackResume` [protected]

Used to abort a pending playback resume if the user calls stop or pause.

4.13.3.2 - (OALAction*) `gainAction` [protected]

Current action operating on the gain control.

4.13.3.3 - (OALAction*) `panAction` [protected]

Current action operating on the pan control.

4.13.3.4 - (OALAction*) `pitchAction` [protected]

Current action operating on the pitch control.

4.13.3.5 - (int) `shadowState` [protected]

Shadow value which keeps the correct state value for AL_PLAYING and AL_PAUSED.

We need this due to a buggy OpenAL implementation.

4.13.3.6 - (OALSuspendHandler*) suspendHandler [protected]

Handles suspending and interrupting for this object.

4.13.4 Property Documentation

4.13.4.1 - (ALBuffer *) buffer [read], [write], [nonatomic], [retain]

The sound buffer this source is attached to (set to nil to detach the currently attached buffer).

4.13.4.2 - (int) buffersProcessed [read], [nonatomic], [assign]

How many of these buffers have been processed during playback.

4.13.4.3 - (int) buffersQueued [read], [nonatomic], [assign]

How many buffers this source has queued.

4.13.4.4 - (ALContext *) context [read], [nonatomic], [retain]

The context this source was opened on.

4.13.4.5 - (float) offsetInBytes [read], [write], [nonatomic], [assign]

The offset into the current buffer (in bytes).

4.13.4.6 - (float) offsetInSamples [read], [write], [nonatomic], [assign]

The offset into the current buffer (in samples).

4.13.4.7 - (float) offsetInSeconds [read], [write], [nonatomic], [assign]

The offset into the current buffer (in seconds).

4.13.4.8 - (ALuint) sourceID [read], [nonatomic], [assign]

OpenAL's ID for this source.

4.13.4.9 - (int) state [read], [write], [nonatomic], [assign]

The state of this source.

The documentation for this class was generated from the following files:

- ALSource.h
- ALSource.m

4.14 ALVector Struct Reference

Represents a 3-dimensional vector for certain ObjectAL properties.

```
#include <ALTypes.h>
```

Public Attributes

- float [x](#)
The "X" coordinate.
- float [y](#)
The "Y" coordinate.
- float [z](#)
The "Z" coordinate.

4.14.1 Detailed Description

Represents a 3-dimensional vector for certain ObjectAL properties.

Properties are the same as for [ALPoint](#).

4.14.2 Member Data Documentation

4.14.2.1 float ALVector::x

The "X" coordinate.

4.14.2.2 float ALVector::y

The "Y" coordinate.

4.14.2.3 float ALVector::z

The "Z" coordinate.

The documentation for this struct was generated from the following file:

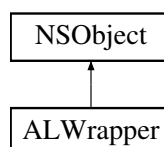
- [ALTypes.h](#)

4.15 ALWrapper Class Reference

A thin wrapper around the C OpenAL API, with a few convenience methods thrown in.

```
#import <ALWrapper.h>
```

Inheritance diagram for ALWrapper:



Class Methods

- (bool) + [genBuffers:numBuffers:](#)
Generate buffers.
- (ALuint) + [genBuffer](#)
Generate a buffer.
- (bool) + [deleteBuffers:numBuffers:](#)
Delete buffers.
- (bool) + [deleteBuffer:](#)
Delete a buffer.
- (bool) + [isBuffer:](#)
Check if the speified buffer exists.
- (bool) + [bufferData:format:data:size:frequency:](#)
Load data into a buffer.
- (bool) + [bufferf:parameter:value:](#)
Write a float paramter to a buffer.
- (bool) + [buffer3f:parameter:v1:v2:v3:](#)
Write a 3 float paramter to a buffer.
- (bool) + [bufferfv:parameter:values:](#)
Write a float array paramter to a buffer.
- (bool) + [bufferi:parameter:value:](#)
Write an integer paramter to a buffer.
- (bool) + [buffer3i:parameter:v1:v2:v3:](#)
Write a 3 integer paramter to a buffer.
- (bool) + [bufferiv:parameter:values:](#)
Write an integer array paramter to a buffer.
- (ALfloat) + [getBufferf:parameter:](#)
Read a float paramter from a buffer.
- (bool) + [getBuffer3f:parameter:v1:v2:v3:](#)
Read a 3 float paramter from a buffer.
- (bool) + [getBufferfv:parameter:values:](#)
Read a float array paramter from a buffer.
- (ALint) + [getBufferi:parameter:](#)
Read an integer paramter from a buffer.
- (bool) + [getBuffer3i:parameter:v1:v2:v3:](#)
Read a 3 integer paramter from a buffer.
- (bool) + [getBufferiv:parameter:values:](#)
Read an integer array paramter from a buffer.
- (bool) + [genSources:numSources:](#)
Generate sources.
- (ALuint) + [genSource](#)
Generate a source.
- (bool) + [deleteSources:numSources:](#)
Delete sources.
- (bool) + [deleteSource:](#)
Delete a source.
- (bool) + [isSource:](#)
Check if the speified source exists.
- (bool) + [sourcePlay:](#)
Play a source.
- (bool) + [sourcePlayv:numSources:](#)

- Play a bunch of sources.*

 - (bool) + [sourcePause:](#)

Pause a source.
- (bool) + [sourcePausev:numSources:](#)

Pause a bunch of sources.
- (bool) + [sourceStop:](#)

Stop a source.
- (bool) + [sourceStopv:numSources:](#)

Stop a bunch of sources.
- (bool) + [sourceRewind:](#)

Rewind a source.
- (bool) + [sourceRewindv:numSources:](#)

Rewind a bunch of sources.
- (bool) + [sourceQueueBuffers:numBuffers:bufferIds:](#)

Queue buffers into a source for sequential playback.
- (bool) + [sourceUnqueueBuffers:numBuffers:bufferIds:](#)

Unqueue previously queued buffers.
- (bool) + [sourcecf:parameter:value:](#)

Write a float paramter to a source.
- (bool) + [source3f:parameter:v1:v2:v3:](#)

Write a 3 float paramter to a source.
- (bool) + [sourcefv:parameter:values:](#)

Write a float array paramter to a source.
- (bool) + [sourceci:parameter:value:](#)

Write an integer paramter to a source.
- (bool) + [source3i:parameter:v1:v2:v3:](#)

Write a 3 integer paramter to a source.
- (bool) + [sourceiv:parameter:values:](#)

Write an integer array paramter to a source.
- (ALfloat) + [getSourcecf:parameter:](#)

Read a float paramter from a source.
- (bool) + [getSource3f:parameter:v1:v2:v3:](#)

Read a 3 float paramter from a source.
- (bool) + [getSourcefv:parameter:values:](#)

Read a float array paramter from a source.
- (ALint) + [getSourceci:parameter:](#)

Read an integer paramter from a source.
- (bool) + [getSource3i:parameter:v1:v2:v3:](#)

Read a 3 integer paramter from a source.
- (bool) + [getSourceiv:parameter:values:](#)

Read an integer array paramter from a source.
- (bool) + [listenerf:value:](#)

Write a float paramter to the current listener.
- (bool) + [listener3f:v1:v2:v3:](#)

Write a 3 float paramter to the current listener.
- (bool) + [listenerfv:values:](#)

Write a float array paramter to the current listener.
- (bool) + [listeneri:value:](#)

Write an integer paramter to the current listener.
- (bool) + [listener3i:v1:v2:v3:](#)

Write a 3 integer paramter to the current listener.

- (bool) + [listeneriv:values:](#)
Write an integer array paramter to the current listener.
- (ALfloat) + [getListenerf:](#)
Read a float paramter from the current listener.
- (bool) + [getListener3f:v1:v2:v3:](#)
Read a 3 float paramter from the current listener.
- (bool) + [getListenerfv:values:](#)
Read a float array paramter from the current listener.
- (ALint) + [getListeneri:](#)
Read an integer paramter from the current listener.
- (bool) + [getListener3i:v1:v2:v3:](#)
Read a 3 integer paramter from the current listener.
- (bool) + [getListeneriv:values:](#)
Read an integer array paramter from the current listener.
- (bool) + [enable:](#)
Enable a capability.
- (bool) + [disable:](#)
Disable a capability.
- (bool) + [isEnabled:](#)
Check if a capability is enabled.
- (bool) + [getBoolean:](#)
Get a boolean parameter.
- (ALdouble) + [getDouble:](#)
Get a double parameter.
- (ALfloat) + [getFloat:](#)
Get a float parameter.
- (ALint) + [getInteger:](#)
Get an integer parameter.
- (NSString *) + [getString:](#)
Get a string parameter.
- (NSArray *) + [getNullSeparatedStringList:](#)
Get a string list parameter.
- (NSArray *) + [getSpaceSeparatedStringList:](#)
Get a string list parameter.
- (bool) + [getBooleanv:values:](#)
Get a boolean array parameter.
- (bool) + [getDoublev:values:](#)
Get a double array parameter.
- (bool) + [getFloatv:values:](#)
Get a float array parameter.
- (bool) + [getIntegerv:values:](#)
Get an integer array parameter.
- (bool) + [distanceModel:](#)
Set the distance model.
- (bool) + [dopplerFactor:](#)
Set the doppler factor.
- (bool) + [speedOfSound:](#)
Set the speed of sound.
- (bool) + [isExtensionPresent:](#)
Check if an extension is present.
- (void *) + [getProcAddress:](#)

- Get the address of a procedure.*

 - (ALenum) + [getEnumValue:](#)

Get the enum value from its name.
- (ALCdevice *) + [openDevice:](#)

Open a device.
- (bool) + [closeDevice:](#)

Close a device.
- (ALCcontext *) + [createContext:attributes:](#)

Create an OpenAL context.
- (bool) + [makeContextCurrent:](#)

Make the specified context the current context.
- (bool) + [makeContextCurrent:deviceReference:](#)

Make the specified context the current context, passing in a device reference for more informative logging info.
- (void) + [processContext:](#)

Process a context.
- (void) + [suspendContext:](#)

Suspend a context.
- (void) + [destroyContext:](#)

Destroy a context.
- (ALCcontext *) + [getCurrentContext](#)

Get the current context.
- (ALCdevice *) + [getContextsDevice:](#)

Get the device a context was created from.
- (ALCdevice *) + [getContextsDevice:deviceReference:](#)

Get the device a context was created from, passing in a device reference for more informative logging info.
- (bool) + [isExtensionPresent:name:](#)

Check if an extension is present on a device.
- (void *) + [getProcAddress:name:](#)

Get the address of a procedure for a device.
- (ALenum) + [getEnumValue:name:](#)

Get the enum value from its name.
- (NSString *) + [getString:attribute:](#)

Get a string attribute.
- (NSArray *) + [getNullSeparatedStringList:attribute:](#)

Get a string list attribute.
- (NSArray *) + [getSpaceSeparatedStringList:attribute:](#)

Get a string list attribute.
- (ALint) + [getInteger:attribute:](#)

Get an integer attribute.
- (bool) + [getInterv:attribute:size:data:](#)

Get an integer array attribute.
- (ALCdevice *) + [openCaptureDevice:frequency:format:bufferSize:](#)

UNSUPPORTED ON IOS Open an audio capture device.
- (bool) + [closeCaptureDevice:](#)

Close a capture device.
- (bool) + [startCapture:](#)

Start capturing audio data.
- (bool) + [stopCapture:](#)

Stop capturing audio data.
- (bool) + [captureSamples:buffer:numSamples:](#)

Get captured samples from a device.

- (ALdouble) + [getMixerOutputDataRate](#)
Get the iOS device's mixer output data rate.
- (bool) + [setMixerOutputDataRate:](#)
Set the iOS device's mixer output data rate.
- (bool) + [bufferDataStatic:format:data:size:frequency:](#)
Load data into a buffer.
- (bool) + [asaGetListenerb:](#)
Read a boolean ASA property from a listener.
- (ALint) + [asaGetListeneri:](#)
Read an integer ASA property from a listener.
- (ALfloat) + [asaGetListenerf:](#)
Read a floating point ASA property from a listener.
- (bool) + [asaListenerb:value:](#)
Write a boolean ASA value to a listener.
- (bool) + [asaListeneri:value:](#)
Write an integer ASA value to a listener.
- (bool) + [asaListenerf:value:](#)
Write a floating point ASA value to a listener.
- (bool) + [asaGetSourceb:property:](#)
Read a boolean ASA property from a source.
- (ALint) + [asaGetSourcei:property:](#)
Read an integer ASA property from a source.
- (ALfloat) + [asaGetSourcef:property:](#)
Read a floating point ASA property from a source.
- (bool) + [asaSourceb:property:value:](#)
Write a boolean ASA value to a source.
- (bool) + [asaSourcei:property:value:](#)
Write an integer ASA value to a source.
- (bool) + [asaSourcef:property:value:](#)
Write a floating point ASA value to a source.
- (bool) + [setRenderingQuality:](#)
Set the rendering quality.
- (ALint) + [getRenderingQuality](#)
Get the rendering quality.
- (bool) + [addNotification:onSource:callback:userData:](#)
Add a notification callback to a source.
- (bool) + [removeNotification:onSource:callback:userData:](#)
Remove a notification callback from a source.

4.15.1 Detailed Description

A thin wrapper around the C OpenAL API, with a few convenience methods thrown in.

Wherever possible, methods return the requested data rather than requiring a pointer to be passed in. Besides collecting the API calls into a single global object, all calls are combined with an error check. Any OpenAL errors that occur will be logged if error logging is enabled.

4.15.2 Method Documentation

4.15.2.1 **+ (bool) addNotification: (ALuint) notificationID onSource:(ALuint) source callback:(alSourceNotificationProc) callback userData:(void*) userData**

Add a notification callback to a source.

The following notification types are recognized: AL_SOURCE_STATE - Sent when a source's state changes. AL_BUFFERS_PROCESSED - Sent when all buffers have been processed. AL_QUEUE_HAS_LOOPED - Sent when a looping source has looped to it's start point.

Parameters

<i>notificationID</i>	The kind of notification to be informed of (see above).
<i>source</i>	The source ID.
<i>callback</i>	The function to call for notification.
<i>userData</i>	a pointer that will be passed to the callback.

Returns

TRUE if the operation was successful.

4.15.2.2 **+ (bool) asaGetListenerb: (ALuint) property**

Read a boolean ASA property from a listener.

Parameters

<i>property</i>	The property to read.
-----------------	-----------------------

Returns

The property's value.

4.15.2.3 **+ (ALfloat) asaGetListenerf: (ALuint) property**

Read a floating point ASA property from a listener.

Parameters

<i>property</i>	The property to read.
-----------------	-----------------------

Returns

The property's value.

4.15.2.4 **+ (ALint) asaGetListeneri: (ALuint) property**

Read an integer ASA property from a listener.

Parameters

<i>property</i>	The property to read.
-----------------	-----------------------

Returns

The property's value.

4.15.2.5 + (bool) asaGetSourceb: (ALuint) *sourceId* property:(ALuint) *property*

Read a boolean ASA property from a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>property</i>	The property to read.

Returns

The property's value.

4.15.2.6 + (ALfloat) asaGetSourcef: (ALuint) *sourceId* property:(ALuint) *property*

Read a floating point ASA property from a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>property</i>	The property to read.

Returns

The property's value.

4.15.2.7 + (ALint) asaGetSourcei: (ALuint) *sourceId* property:(ALuint) *property*

Read an integer ASA property from a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>property</i>	The property to read.

Returns

The property's value.

4.15.2.8 + (bool) asaListenerb: (ALuint) *property* value:(bool) *value*

Write a boolean ASA value to a listener.

Parameters

<i>property</i>	The property to write to.
<i>value</i>	The value to write.

Returns

TRUE if the operation was successful.

4.15.2.9 + (bool) asaListenerf: (ALuint) *property* value:(ALfloat) *value*

Write a floating point ASA value to a listener.

Parameters

<i>property</i>	The property to write to.
<i>value</i>	The value to write.

Returns

TRUE if the operation was successful.

4.15.2.10 + (bool) asaListeneri: (ALuint) *property* value:(ALint) *value*

Write an integer ASA value to a listener.

Parameters

<i>property</i>	The property to write to.
<i>value</i>	The value to write.

Returns

TRUE if the operation was successful.

4.15.2.11 + (bool) asaSourceb: (ALuint) *sourceId* property:(ALuint) *property* value:(bool) *value*

Write a boolean ASA value to a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>property</i>	The property to write to.
<i>value</i>	The value to write.

Returns

TRUE if the operation was successful.

4.15.2.12 + (bool) asaSourcef: (ALuint) *sourceId* property:(ALuint) *property* value:(ALfloat) *value*

Write a floating point ASA value to a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>property</i>	The property to write to.
<i>value</i>	The value to write.

Returns

TRUE if the operation was successful.

4.15.2.13 + (bool) asaSourcei: (ALuint) *sourceId* property:(ALuint) *property* value:(ALint) *value*

Write an integer ASA value to a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>property</i>	The property to write to.
<i>value</i>	The value to write.

Returns

TRUE if the operation was successful.

4.15.2.14 + (bool) buffer3f: (ALuint) *bufferId* parameter:(ALenum) *parameter* v1:(ALfloat) *v1* v2:(ALfloat) *v2* v3:(ALfloat) *v3*

Write a 3 float paramter to a buffer.

Parameters

<i>bufferId</i>	The buffer's ID.
<i>parameter</i>	the parameter to write to.
<i>v1</i>	The first value to write.
<i>v2</i>	The second value to write.
<i>v3</i>	The third value to write.

Returns

TRUE if the operation was successful.

4.15.2.15 + (bool) buffer3i: (ALuint) *bufferId* parameter:(ALenum) *parameter* v1:(ALint) *v1* v2:(ALint) *v2* v3:(ALint) *v3*

Write a 3 integer paramter to a buffer.

Parameters

<i>bufferId</i>	The buffer's ID.
<i>parameter</i>	The parameter to write to.
<i>v1</i>	The first value to write.
<i>v2</i>	The second value to write.
<i>v3</i>	The third value to write.

Returns

TRUE if the operation was successful.

4.15.2.16 + (bool) bufferData: (ALuint) *bufferId* format:(ALenum) *format* data:(const ALvoid*) *data* size:(ALsizei) *size* frequency:(ALsizei) *frequency*

Load data into a buffer.

Parameters

<i>bufferId</i>	The ID of the buffer to load data into.
<i>format</i>	The format of the data being loaded (typically AL_FORMAT_MONO16 or AL_FORMAT_STEREO16).
<i>data</i>	The audio data.
<i>size</i>	The size of the data in bytes.
<i>frequency</i>	The sample frequency of the data.

4.15.2.17 + (bool) *bufferDataStatic*: (ALuint) *bufferId* *format*:(ALenum) *format* *data*:(const ALvoid*) *data* *size*:(ALsizei) *size* *frequency*:(ALsizei) *frequency*

Load data into a buffer.

Unlike "bufferData", with this method the buffer will use the passed in data buffer directly rather than allocating its own memory and copying from the data buffer.

Parameters

<i>bufferId</i>	The ID of the buffer to load data into.
<i>format</i>	The format of the data being loaded (typically AL_FORMAT_MONO16 or AL_FORMAT_STEREO16).
<i>data</i>	The audio data.
<i>size</i>	The size of the data in bytes.
<i>frequency</i>	The sample frequency of the data.

4.15.2.18 + (bool) *bufferf*: (ALuint) *bufferId* *parameter*:(ALenum) *parameter* *value*:(ALfloat) *value*

Write a float paramter to a buffer.

Parameters

<i>bufferId</i>	The buffer's ID.
<i>parameter</i>	The parameter to write to.
<i>value</i>	The value to write.

Returns

TRUE if the operation was successful.

4.15.2.19 + (bool) *bufferfv*: (ALuint) *bufferId* *parameter*:(ALenum) *parameter* *values*:(ALfloat*) *values*

Write a float array paramter to a buffer.

Parameters

<i>bufferId</i>	The buffer's ID.
<i>parameter</i>	The parameter to write to.
<i>values</i>	The values to write.

Returns

TRUE if the operation was successful.

4.15.2.20 + (bool) *bufferi*: (ALuint) *bufferId* parameter:(ALenum) *parameter* value:(ALint) *value*

Write an integer paramter to a buffer.

Parameters

<i>bufferId</i>	The buffer's ID.
<i>parameter</i>	The parameter to write to.
<i>value</i>	The value to write.

Returns

TRUE if the operation was successful.

4.15.2.21 + (bool) *bufferiv*: (ALuint) *bufferId* parameter:(ALenum) *parameter* values:(ALint*) *values*

Write an integer array paramter to a buffer.

Parameters

<i>bufferId</i>	The buffer's ID.
<i>parameter</i>	The parameter to write to.
<i>values</i>	The values to write.

Returns

TRUE if the operation was successful.

4.15.2.22 + (bool) *captureSamples*: (ALCdevice*) *device* *buffer*:(ALCvoid*) *buffer* numSamples:(ALCsizei) *numSamples*

Get captured samples from a device.

Parameters

<i>device</i>	the device to fetch samples from.
<i>buffer</i>	the buffer to copy the samples into.
<i>numSamples</i>	the number of samples to fetch.

4.15.2.23 + (bool) *closeCaptureDevice*: (ALCdevice*) *device*

Close a capture device.

Parameters

<i>device</i>	The device to close.
---------------	----------------------

Returns

TRUE if the operation was successful.

4.15.2.24 + (bool) *closeDevice*: (ALCdevice*) *device*

Close a device.

Parameters

<i>device</i>	The device to close.
---------------	----------------------

Returns

TRUE if the operation was successful.

4.15.2.25 + (ALContext *) createContext: (ALCdevice*) device attributes:(ALCint*) attributes

Create an OpenAL context.

Parameters

<i>device</i>	The device to open the context on.
<i>attributes</i>	The attributes to use when creating the context.

Returns

The new context.

4.15.2.26 + (bool) deleteBuffer: (ALuint) bufferId

Delete a buffer.

Parameters

<i>bufferId</i>	The ID of the buffer to delete.
-----------------	---------------------------------

Returns

TRUE if the operation was successful.

4.15.2.27 + (bool) deleteBuffers: (ALuint*) bufferIds numBuffers:(ALsizei) numBuffers

Delete buffers.

Parameters

<i>bufferIds</i>	Pointer to an array containing the buffer IDs.
<i>numBuffers</i>	the number of buffers to delete.

Returns

TRUE if the operation was successful.

4.15.2.28 + (bool) deleteSource: (ALuint) sourceId

Delete a source.

Parameters

<i>sourceId</i>	The ID of the source to delete.
-----------------	---------------------------------

Returns

TRUE if the operation was successful.

4.15.2.29 + (bool) deleteSources: (ALuint*) *sourceIds* numSources:(ALsizei) *numSources*

Delete sources.

Parameters

<i>sourceIds</i>	Pointer to an array containing the source IDs.
<i>numSources</i>	the number of sources to delete.

Returns

TRUE if the operation was successful.

4.15.2.30 + (void) destroyContext: (ALCcontext*) *context*

Destroy a context.

Parameters

<i>context</i>	The context to destroy.
----------------	-------------------------

Returns

TRUE if the operation was successful.

4.15.2.31 + (bool) disable: (ALenum) *capability*

Disable a capability.

Parameters

<i>capability</i>	The capability to disable.
-------------------	----------------------------

Returns

TRUE if the operation was successful.

4.15.2.32 + (bool) distanceModel: (ALenum) *value*

Set the distance model.

Parameters

<i>value</i>	The value to set.
--------------	-------------------

Returns

TRUE if the operation was successful.

4.15.2.33 + (bool) dopplerFactor: (ALfloat) *value*

Set the doppler factor.

Parameters

<i>value</i>	The value to set.
--------------	-------------------

Returns

TRUE if the operation was successful.

4.15.2.34 + (bool) enable: (ALenum) *capability*

Enable a capability.

Parameters

<i>capability</i>	The capability to enable.
-------------------	---------------------------

Returns

TRUE if the operation was successful.

4.15.2.35 + (ALuint) genBuffer

Generate a buffer.

Returns

the buffer's ID.

4.15.2.36 + (bool) genBuffers: (ALuint*) *bufferIds* numBuffers:(ALsizei) *numBuffers*

Generate buffers.

Parameters

<i>bufferIds</i>	Pointer to an array that will receive the buffer IDs.
<i>numBuffers</i>	the number of buffers to generate.

Returns

TRUE if the operation was successful.

4.15.2.37 + (ALuint) genSource

Generate a source.

Returns

the source's ID.

4.15.2.38 + (bool) genSources: (ALuint*) *sourceIds* numSources:(ALsizei) *numSources*

Generate sources.

Parameters

<i>sourceIds</i>	Pointer to an array that will receive the source IDs.
<i>numSources</i>	the number of sources to generate.

Returns

TRUE if the operation was successful.

4.15.2.39 + (bool) getBoolean: (ALenum) *parameter*

Get a boolean parameter.

Parameters

<i>parameter</i>	The parameter to fetch.
------------------	-------------------------

Returns

The parameter's current value.

4.15.2.40 + (bool) getBooleanv: (ALenum) *parameter* values:(ALboolean*) *values*

Get a boolean array parameter.

Parameters

<i>parameter</i>	The parameter to fetch.
<i>values</i>	An array to hold the result.

Returns

TRUE if the operation was successful.

4.15.2.41 + (bool) getBuffer3f: (ALuint) *bufferId* parameter:(ALenum) *parameter* v1:(ALfloat*) *v1* v2:(ALfloat*) *v2* v3:(ALfloat*) *v3*

Read a 3 float paramter from a buffer.

Parameters

<i>bufferId</i>	The buffer's ID.
<i>parameter</i>	The parameter to read.
<i>v1</i>	The first value to read.
<i>v2</i>	The second value to read.
<i>v3</i>	The third value to read.

Returns

TRUE if the operation was successful.

4.15.2.42 + (bool) getBuffer3i: (ALuint) *bufferId* parameter:(ALenum) *parameter* v1:(ALint*) v1 v2:(ALint*) v2 v3:(ALint*) v3

Read a 3 integer paramter from a buffer.

Parameters

<i>bufferId</i>	The buffer's ID.
<i>parameter</i>	The parameter to read.
<i>v1</i>	The first value to read.
<i>v2</i>	The second value to read.
<i>v3</i>	The third value to read.

Returns

TRUE if the operation was successful.

4.15.2.43 + (ALfloat) getBufferf: (ALuint) *bufferId* parameter:(ALenum) *parameter*

Read a float paramter from a buffer.

Parameters

<i>bufferId</i>	The buffer's ID.
<i>parameter</i>	The parameter to read.

Returns

The parameter's value.

4.15.2.44 + (bool) getBufferfv: (ALuint) *bufferId* parameter:(ALenum) *parameter* values:(ALfloat*) *values*

Read a float array paramter from a buffer.

Parameters

<i>bufferId</i>	The buffer's ID.
<i>parameter</i>	The parameter to read.
<i>values</i>	An array to store the values.

Returns

TRUE if the operation was successful.

4.15.2.45 + (ALint) getBufferi: (ALuint) *bufferId* parameter:(ALenum) *parameter*

Read an integer paramter from a buffer.

Parameters

<i>bufferId</i>	The buffer's ID.
<i>parameter</i>	The parameter to read.

Returns

The parameter's value.

4.15.2.46 + (bool) getBufferiv: (ALuint) *bufferId* parameter:(ALenum) *parameter* values:(ALint*) *values*

Read an integer array paramter from a buffer.

Parameters

<i>bufferId</i>	The buffer's ID.
<i>parameter</i>	The parameter to read.
<i>values</i>	An array to store the values.

Returns

TRUE if the operation was successful.

4.15.2.47 + (ALCdevice *) getContextsDevice: (ALCcontext*) *context*

Get the device a context was created from.

Parameters

<i>context</i>	The context.
----------------	--------------

Returns

The context's device.

4.15.2.48 + (ALCdevice *) getContextsDevice: (ALCcontext*) *context* deviceReference:(ALCdevice*) *deviceReference*

Get the device a context was created from, passing in a device reference for more informative logging info.

Parameters

<i>context</i>	The context.
<i>deviceReference</i>	The device reference to use when logging an error.

Returns

The context's device.

4.15.2.49 + (ALCcontext *) getCurrentContext

Get the current context.

Returns

the current context.

4.15.2.50 + (ALdouble) getDouble: (ALenum) *parameter*

Get a double parameter.

Parameters

<i>parameter</i>	The parameter to fetch.
------------------	-------------------------

Returns

The parameter's current value.

4.15.2.51 + (bool) getDoublev: (ALenum) *parameter* values:(ALdouble*) *values*

Get a double array parameter.

Parameters

<i>parameter</i>	The parameter to fetch.
<i>values</i>	An array to hold the result.

Returns

TRUE if the operation was successful.

4.15.2.52 + (ALenum) getEnumValue: (NSString*) *enumName*

Get the enum value from its name.

Parameters

<i>enumName</i>	the name of the enum value.
-----------------	-----------------------------

Returns

The enum value.

4.15.2.53 + (ALenum) getEnumValue: (ALCdevice*) *device* name:(NSString*) *enumName*

Get the enum value from its name.

Parameters

<i>device</i>	The device to check on.
<i>enumName</i>	the name of the enum value.

Returns

The enum value.

4.15.2.54 + (ALfloat) getFloat: (ALenum) *parameter*

Get a float parameter.

Parameters

<i>parameter</i>	The parameter to fetch.
------------------	-------------------------

Returns

The parameter's current value.

4.15.2.55 + (bool) getFloatv: (ALenum) *parameter* values:(ALfloat*) *values*

Get a float array parameter.

Parameters

<i>parameter</i>	The parameter to fetch.
<i>values</i>	An array to hold the result.

Returns

TRUE if the operation was successful.

4.15.2.56 + (ALint) getInteger: (ALenum) *parameter*

Get an integer parameter.

Parameters

<i>parameter</i>	The parameter to fetch.
------------------	-------------------------

Returns

The parameter's current value.

4.15.2.57 + (ALint) getInteger: (ALCdevice*) *device* attribute:(ALenum) *attribute*

Get an integer attribute.

Parameters

<i>device</i>	The device to read the attribute from.
<i>attribute</i>	The attribute to fetch.

Returns

The parameter's current value.

4.15.2.58 + (bool) getInterv: (ALCdevice*) *device* attribute:(ALenum) *attribute* size:(ALsizei) *size* data:(ALCint*) *data*

Get an integer array attribute.

Parameters

<i>device</i>	The device to read the attribute from.
<i>attribute</i>	The attribute to read.
<i>size</i>	the size of the receiving array.
<i>data</i>	An array to store the values.

Returns

TRUE if the operation was successful.

4.15.2.59 + (bool) getIntegerv: (ALenum) *parameter* values:(ALint*) *values*

Get an integer array parameter.

Parameters

<i>parameter</i>	The parameter to fetch.
<i>values</i>	An array to hold the result.

Returns

TRUE if the operation was successful.

4.15.2.60 + (bool) getListener3f: (ALenum) *parameter* v1:(ALfloat*) v1 v2:(ALfloat*) v2 v3:(ALfloat*) v3

Read a 3 float paramter from the current listener.

Parameters

<i>parameter</i>	The parameter to read.
<i>v1</i>	The first value to read.
<i>v2</i>	The second value to read.
<i>v3</i>	The third value to read.

Returns

TRUE if the operation was successful.

4.15.2.61 + (bool) getListener3i: (ALenum) *parameter* v1:(ALint*) v1 v2:(ALint*) v2 v3:(ALint*) v3

Read a 3 integer paramter from the current listener.

Parameters

<i>parameter</i>	The parameter to read.
<i>v1</i>	The first value to read.
<i>v2</i>	The second value to read.
<i>v3</i>	The third value to read.

Returns

TRUE if the operation was successful.

4.15.2.62 + (ALfloat) getListenerf: (ALenum) *parameter*

Read a float paramter from the current listener.

Parameters

<i>parameter</i>	The parameter to read.
------------------	------------------------

Returns

The parameter's value.

4.15.2.63 + (bool) getListenerfv: (ALenum) parameter values:(ALfloat*) values

Read a float array paramter from the current listener.

Parameters

<i>parameter</i>	The parameter to read.
<i>values</i>	An array to store the values.

Returns

TRUE if the operation was successful.

4.15.2.64 + (ALint) getListeneri: (ALenum) parameter

Read an integer paramter from the current listener.

Parameters

<i>parameter</i>	The parameter to read.
------------------	------------------------

Returns

The parameter's value.

4.15.2.65 + (bool) getListeneriv: (ALenum) parameter values:(ALint*) values

Read an integer array paramter from the current listener.

Parameters

<i>parameter</i>	The parameter to read.
<i>values</i>	An array to store the values.

Returns

TRUE if the operation was successful.

4.15.2.66 + (ALdouble) getMixerOutputDataRate

Get the iOS device's mixer outut data rate.

Returns

The mixer output data rate.

4.15.2.67 + (NSArray *) getNullSeparatedStringList: (ALenum) parameter

Get a string list parameter.

Use this method for OpenAL parameters that return a null separated list.

Parameters

<i>parameter</i>	The parameter to fetch.
------------------	-------------------------

Returns

The parameter's current value (as an array of NSString*).

4.15.2.68 + (NSArray *) getNullSeparatedStringList: (ALCdevice*) device attribute:(ALenum) attribute

Get a string list attribute.

Use this method for OpenAL attributes that return a null separated list.

Parameters

<i>device</i>	The device to read the attribute from.
<i>attribute</i>	The attribute to fetch.

Returns

The parameter's current value (as an array of NSString*).

4.15.2.69 + (void *) getProcAddress: (NSString*) functionName

Get the address of a procedure.

Parameters

<i>functionName</i>	The name of the procedure to fetch.
---------------------	-------------------------------------

Returns

A pointer to the procedure, or NULL if it wasn't found.

4.15.2.70 + (void *) getProcAddress: (ALCdevice*) device name:(NSString*) functionName

Get the address of a procedure for a device.

Parameters

<i>device</i>	The device to check on.
<i>functionName</i>	The name of the procedure to check for.

Returns

The procedure's address, or NULL if not found.

4.15.2.71 + (ALint) getRenderingQuality

Get the rendering quality.

Returns

The current rendering quality.

4.15.2.72 + (bool) getSource3f: (ALuint) *sourceId* parameter:(ALenum) *parameter* v1:(ALfloat*) v1 v2:(ALfloat*) v2 v3:(ALfloat*) v3

Read a 3 float paramter from a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>parameter</i>	The parameter to read.
<i>v1</i>	The first value to read.
<i>v2</i>	The second value to read.
<i>v3</i>	The third value to read.

Returns

TRUE if the operation was successful.

4.15.2.73 + (bool) getSource3i: (ALuint) *sourceId* parameter:(ALenum) *parameter* v1:(ALint*) v1 v2:(ALint*) v2 v3:(ALint*) v3

Read a 3 integer paramter from a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>parameter</i>	The parameter to read.
<i>v1</i>	The first value to read.
<i>v2</i>	The second value to read.
<i>v3</i>	The third value to read.

Returns

TRUE if the operation was successful.

4.15.2.74 + (ALfloat) getSourcef: (ALuint) *sourceId* parameter:(ALenum) *parameter*

Read a float paramter from a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>parameter</i>	The parameter to read.

Returns

The parameter's value.

4.15.2.75 + (bool) getSourcefv: (ALuint) *sourceId* parameter:(ALenum) *parameter* values:(ALfloat*) *values*

Read a float array paramter from a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>parameter</i>	The parameter to read.
<i>values</i>	An array to store the values.

Returns

TRUE if the operation was successful.

4.15.2.76 + (ALint) getSourceId: (ALuint) sourceId parameter:(ALenum) parameter

Read an integer parameter from a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>parameter</i>	The parameter to read.

Returns

The parameter's value.

4.15.2.77 + (bool) getSourceIv: (ALuint) sourceId parameter:(ALenum) parameter values:(ALint*) values

Read an integer array parameter from a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>parameter</i>	The parameter to read.
<i>values</i>	An array to store the values.

Returns

TRUE if the operation was successful.

4.15.2.78 + (NSArray *) getSpaceSeparatedStringList: (ALenum) parameter

Get a string list parameter.

Use this method for OpenAL parameters that return a space separated list.

Parameters

<i>parameter</i>	The parameter to fetch.
------------------	-------------------------

Returns

The parameter's current value (as an array of NSString*).

4.15.2.79 + (NSArray *) getSpaceSeparatedStringList: (ALCdevice*) device attribute:(ALenum) attribute

Get a string list attribute.

Use this method for OpenAL attributes that return a space separated list.

Parameters

<i>device</i>	The device to read the attribute from.
<i>attribute</i>	The attribute to fetch.

Returns

The parameter's current value (as an array of NSString*).

4.15.2.80 + (NSString *) getString: (ALenum) parameter

Get a string parameter.

Parameters

<i>parameter</i>	The parameter to fetch.
------------------	-------------------------

Returns

The parameter's current value.

4.15.2.81 + (NSString *) getString: (ALCdevice*) device attribute:(ALenum) attribute

Get a string attribute.

Parameters

<i>device</i>	The device to read the attribute from.
<i>attribute</i>	The attribute to fetch.

Returns

The parameter's current value.

4.15.2.82 + (bool) isBuffer: (ALuint) bufferId

Check if the speified buffer exists.

Parameters

<i>bufferId</i>	The ID of the buffer to query.
-----------------	--------------------------------

Returns

TRUE if the buffer exists.

4.15.2.83 + (bool) isEnabled: (ALenum) capability

Check if a capability is enabled.

Parameters

<i>capability</i>	The capability to check.
-------------------	--------------------------

Returns

TRUE if the capability is enabled.

4.15.2.84 + (bool) isExtensionPresent: (NSString*) extensionName

Check if an extension is present.

Parameters

<i>extensionName</i>	The name of the extension to check.
----------------------	-------------------------------------

Returns

TRUE if the extension is present.

4.15.2.85 + (bool) isExtensionPresent: (ALCdevice*) device name:(NSString*) extensionName

Check if an extension is present on a device.

Parameters

<i>device</i>	The device to check for an extension on.
<i>extensionName</i>	The name of the extension to check for.

Returns

TRUE if the extension is present.

4.15.2.86 + (bool) isSource: (ALuint) sourceId

Check if the speified source exists.

Parameters

<i>sourceId</i>	The ID of the source to query.
-----------------	--------------------------------

Returns

TRUE if the buffer exists.

4.15.2.87 + (bool) listener3f: (ALenum) parameter v1:(ALfloat) v1 v2:(ALfloat) v2 v3:(ALfloat) v3

Write a 3 float paramter to the current listener.

Parameters

<i>parameter</i>	the parameter to write to.
<i>v1</i>	The first value to write.
<i>v2</i>	The second value to write.
<i>v3</i>	The third value to write.

Returns

TRUE if the operation was successful.

4.15.2.88 + (bool) listener3i: (ALenum) *parameter* v1:(ALint) v1 v2:(ALint) v2 v3:(ALint) v3

Write a 3 integer paramter to the current listener.

Parameters

<i>parameter</i>	The parameter to write to.
<i>v1</i>	The first value to write.
<i>v2</i>	The second value to write.
<i>v3</i>	The third value to write.

Returns

TRUE if the operation was successful.

4.15.2.89 + (bool) listenerf: (ALenum) *parameter* value:(ALfloat) *value*

Write a float paramter to the current listener.

Parameters

<i>parameter</i>	The parameter to write to.
<i>value</i>	The value to write.

Returns

TRUE if the operation was successful.

4.15.2.90 + (bool) listenerfv: (ALenum) *parameter* values:(ALfloat*) *values*

Write a float array paramter to the current listener.

Parameters

<i>parameter</i>	The parameter to write to.
<i>values</i>	The values to write.

Returns

TRUE if the operation was successful.

4.15.2.91 + (bool) listeneri: (ALenum) *parameter* value:(ALint) *value*

Write an integer paramter to the current listener.

Parameters

<i>parameter</i>	The parameter to write to.
<i>value</i>	The value to write.

Returns

TRUE if the operation was successful.

4.15.2.92 + (bool) listeneriv: (ALenum) parameter values:(ALint*) values

Write an integer array paramter to the current listener.

Parameters

<i>parameter</i>	The parameter to write to.
<i>values</i>	The values to write.

Returns

TRUE if the operation was successful.

4.15.2.93 + (bool) makeContextCurrent: (ALCcontext*) context

Make the specified context the current context.

Parameters

<i>context</i>	the context to make current.
----------------	------------------------------

Returns

TRUE if the operation was successful.

4.15.2.94 + (bool) makeContextCurrent: (ALCcontext*) context deviceReference:(ALCdevice*) deviceReference

Make the specified context the current context, passing in a device reference for more informative logging info.

Parameters

<i>context</i>	The context to make current.
<i>deviceReference</i>	The device reference to use when logging an error.

Returns

TRUE if the operation was successful.

4.15.2.95 + (ALCdevice *) openCaptureDevice: (NSString*) deviceName frequency:(ALCuint) frequency format:(ALCenum) format bufferSize:(ALCsizei) bufferSize

UNSUPPORTED ON IOS Open an audio capture device.

Parameters

<i>deviceName</i>	The name of the device to open (nil = open the default device).
<i>frequency</i>	The sampling frequency to use.
<i>format</i>	The format to capture the data as.
<i>bufferSize</i>	The size of capture buffer to use.

Returns

The opened device, or nil if an error occurred.

4.15.2.96 + (ALCdevice *) openDevice: (NSString*) *deviceName*

Open a device.

Parameters

<i>deviceName</i>	The name of the device to open (nil = open the default device).
-------------------	---

Returns

The opened device, or nil on failure.

4.15.2.97 + (void) processContext: (ALCcontext*) *context*

Process a context.

Parameters

<i>context</i>	The context to process.
----------------	-------------------------

Returns

TRUE if the operation was successful.

4.15.2.98 + (bool) removeNotification: (ALuint) *notificationID* onSource:(ALuint) *source* callback:(alSourceNotificationProc) *callback* userData:(void*) *userData*

Remove a notification callback from a source.

Parameters

<i>notificationID</i>	The kind of notification (see addNotification).
<i>source</i>	The source ID.
<i>callback</i>	The function to be unregistered.
<i>userData</i>	not actually needed but part of the API.

Returns

TRUE if the operation was successful.

4.15.2.99 + (bool) setMixerOutputDataRate: (ALdouble) *frequency*

Set the iOS device's mixer output data rate.

Parameters

<i>frequency</i>	The output data rate (frequency).
------------------	-----------------------------------

4.15.2.100 + (bool) setRenderingQuality: (ALint) *quality*

Set the rendering quality.

The value may be one of:

ALC_MAC_OSX_SPATIAL_RENDERING_QUALITY_HIGH ALC_MAC_OSX_SPATIAL_RENDERING_QUALITY_LOW ALC_IPHONE_SPATIAL_RENDERING_QUALITY_HEADPHONES (iOS only)

Parameters

<i>quality</i>	The quality.
----------------	--------------

4.15.2.101 + (bool) source3f: (ALuint) *sourceId* parameter:(ALenum) *parameter* v1:(ALfloat) *v1* v2:(ALfloat) *v2* v3:(ALfloat) *v3*

Write a 3 float paramter to a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>parameter</i>	the parameter to write to.
<i>v1</i>	The first value to write.
<i>v2</i>	The second value to write.
<i>v3</i>	The third value to write.

Returns

TRUE if the operation was successful.

4.15.2.102 + (bool) source3i: (ALuint) *sourceId* parameter:(ALenum) *parameter* v1:(ALint) *v1* v2:(ALint) *v2* v3:(ALint) *v3*

Write a 3 integer paramter to a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>parameter</i>	The parameter to write to.
<i>v1</i>	The first value to write.
<i>v2</i>	The second value to write.
<i>v3</i>	The third value to write.

Returns

TRUE if the operation was successful.

4.15.2.103 + (bool) sourcef: (ALuint) *sourceId* parameter:(ALenum) *parameter* value:(ALfloat) *value*

Write a float paramter to a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>parameter</i>	The parameter to write to.
<i>value</i>	The value to write.

Returns

TRUE if the operation was successful.

4.15.2.104 + (bool) sourcefv: (ALuint) *sourceId* parameter:(ALenum) *parameter* values:(ALfloat*) *values*

Write a float array paramter to a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>parameter</i>	The parameter to write to.
<i>values</i>	The values to write.

Returns

TRUE if the operation was successful.

4.15.2.105 + (bool) sourcei: (ALuint) *sourceId* parameter:(ALenum) *parameter* value:(ALint) *value*

Write an integer paramter to a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>parameter</i>	The parameter to write to.
<i>value</i>	The value to write.

Returns

TRUE if the operation was successful.

4.15.2.106 + (bool) sourceiv: (ALuint) *sourceId* parameter:(ALenum) *parameter* values:(ALint*) *values*

Write an integer array paramter to a source.

Parameters

<i>sourceId</i>	The source's ID.
<i>parameter</i>	The parameter to write to.
<i>values</i>	The values to write.

Returns

TRUE if the operation was successful.

4.15.2.107 + (bool) sourcePause: (ALuint) *sourceId*

Pause a source.

Parameters

<i>sourceId</i>	The ID of the source to pause.
-----------------	--------------------------------

Returns

TRUE if the operation is successful.

4.15.2.108 + (bool) sourcePausev: (ALuint*) *sourceIds* numSources:(ALsizei) *numSources*

Pause a bunch of sources.

Parameters

<i>sourceIds</i>	The sources to pause.
<i>numSources</i>	The number of sources in <i>sourceIds</i> .

Returns

TRUE if the operation is successful.

4.15.2.109 + (bool) sourcePlay: (ALuint) *sourceId*

Play a source.

Parameters

<i>sourceId</i>	The ID of the source to play.
-----------------	-------------------------------

Returns

TRUE if the buffer exists.

4.15.2.110 + (bool) sourcePlayv: (ALuint*) *sourceIds* numSources:(ALsizei) *numSources*

Play a bunch of sources.

Parameters

<i>sourceIds</i>	The sources to play.
<i>numSources</i>	The number of sources in <i>sourceIds</i> .

Returns

TRUE if the operation is successful.

4.15.2.111 + (bool) sourceQueueBuffers: (ALuint) *sourceId* numBuffers:(ALsizei) *numBuffers* bufferIds:(ALuint*) *bufferIds*

Queue buffers into a source for sequential playback.

Parameters

<i>sourceId</i>	The source to use for playback.
<i>numBuffers</i>	The number of buffers to queue.
<i>bufferIds</i>	The IDs of the buffers to queue.

Returns

TRUE if the operation is successful.

4.15.2.112 + (bool) sourceRewind: (ALuint) *sourceId*

Rewind a source.

Parameters

<i>sourceId</i>	The ID of the source to rewind.
-----------------	---------------------------------

Returns

TRUE if the operation is successful.

4.15.2.113 + (bool) sourceRewindv: (ALuint*) *sourceIds* numSources:(ALsizei) *numSources*

Rewind a bunch of sources.

Parameters

<i>sourceIds</i>	The sources to rewind.
<i>numSources</i>	The number of sources in <i>sourceIds</i> .

Returns

TRUE if the operation is successful.

4.15.2.114 + (bool) sourceStop: (ALuint) *sourceId*

Stop a source.

Parameters

<i>sourceId</i>	The ID of the source to stop.
-----------------	-------------------------------

Returns

TRUE if the operation is successful.

4.15.2.115 + (bool) sourceStopv: (ALuint*) *sourceIds* numSources:(ALsizei) *numSources*

Stop a bunch of sources.

Parameters

<i>sourceIds</i>	The sources to stop.
<i>numSources</i>	The number of sources in <i>sourceIds</i> .

Returns

TRUE if the operation is successful.

4.15.2.116 + (bool) sourceUnqueueBuffers: (ALuint) *sourceId* numBuffers:(ALsizei) *numBuffers* bufferIds:(ALuint*) *bufferIds*

Unqueue previously queued buffers.

Parameters

<i>sourceId</i>	The source the buffers were previously queued in.
<i>numBuffers</i>	The number of buffers to unqueue.
<i>bufferIds</i>	The IDs of the buffers to unqueue.

Returns

TRUE if the operation is successful.

4.15.2.117 + (bool) speedOfSound: (ALfloat) *value*

Set the speed of sound.

Parameters

<i>value</i>	The value to set.
--------------	-------------------

Returns

TRUE if the operation was successful.

4.15.2.118 + (bool) startCapture: (ALCdevice*) *device*

Start capturing audio data.

Parameters

<i>device</i>	The device to capture on.
---------------	---------------------------

Returns

TRUE if the operation was successful.

4.15.2.119 + (bool) stopCapture: (ALCdevice*) *device*

Stop capturing audio data.

Parameters

<i>device</i>	The device capturing audio data.
---------------	----------------------------------

Returns

TRUE if the operation was successful.

4.15.2.120 + (void) suspendContext: (ALCcontext*) *context*

Suspend a context.

Parameters

<i>context</i>	The context to suspend.
----------------	-------------------------

Returns

TRUE if the operation was successful.

The documentation for this class was generated from the following files:

- ALWrapper.h
- ALWrapper.m

4.16 ALWrapper(Private) Category Reference

Private interface to [ALWrapper](#).

Instance Methods

- (BOOL) - [checkIfSuccessful](#)
Check the OpenAL error status and log an error message if necessary.
- (BOOL) - [checkIfSuccessfulWithDevice](#)
Check the OpenAL error status and log an error message if necessary.

Class Methods

- (NSArray *) + [decodeNullSeparatedStringList](#):
Decode an OpenAL supplied NULL-separated string list into an NSArray.
- (NSArray *) + [decodeSpaceSeparatedStringList](#):
Decode an OpenAL supplied space-separated string list into an NSArray.

4.16.1 Detailed Description

Private interface to [ALWrapper](#).

4.16.2 Method Documentation

4.16.2.1 - (BOOL) checkIfSuccessful (const char *) contextInfo

Check the OpenAL error status and log an error message if necessary.

Parameters

<i>contextInfo</i>	Contextual information to add when logging an error.
--------------------	--

Returns

TRUE if the operation was successful (no error).

4.16.2.2 - (BOOL) checkIfSuccessfulWithDevice (const char *) *contextInfo* (ALCdevice *) *device*

Check the OpenAL error status and log an error message if necessary.

Parameters

<i>contextInfo</i>	Contextual information to add when logging an error.
<i>device</i>	The device to check for errors on.

Returns

TRUE if the operation was successful (no error).

4.16.2.3 + (NSArray*) decodeNullSeparatedStringList: (const ALCchar *) *source*

Decode an OpenAL supplied NULL-separated string list into an NSArray.

Parameters

<i>source</i>	the string list as supplied by OpenAL.
---------------	--

Returns

the string list in an NSArray of NSString.

4.16.2.4 + (NSArray*) decodeSpaceSeparatedStringList: (const ALCchar *) *source*

Decode an OpenAL supplied space-separated string list into an NSArray.

Parameters

<i>source</i>	the string list as supplied by OpenAL.
---------------	--

Returns

the string list in an NSArray of NSString.

The documentation for this category was generated from the following file:

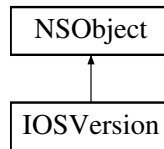
- ALWrapper.m

4.17 IOSVersion Class Reference

Reports the version of iOS being run on the current device.

```
#import <IOSVersion.h>
```

Inheritance diagram for IOSVersion:



Protected Member Functions

- () - [SYNTHESIZE_SINGLETON_FOR_CLASS_HEADER](#)
Singleton implementation providing "sharedInstance" and "purgeSharedInstance" methods.

Properties

- float [version](#)
Holds the current iOS version.

4.17.1 Detailed Description

Reports the version of iOS being run on the current device.

4.17.2 Method Documentation

4.17.2.1 - IOSVersion: (IOSVersion)

Singleton implementation providing "sharedInstance" and "purgeSharedInstance" methods.

- (**IOSVersion***) **sharedInstance**: Get the shared singleton instance.

- (**void**) **purgeSharedInstance**: Purge (deallocate) the shared instance.

4.17.3 Property Documentation

4.17.3.1 - (float) version [read],[nonatomic],[assign]

Holds the current iOS version.

The version of iOS being run on the current device as a float in the format x.yy.

The documentation for this class was generated from the following file:

- IOSVersion.h

4.18 NSMutableArray(WeakReferences) Category Reference

Adds to NSMutableArray the ability to create an array that keeps weak references.

```
#import <NSMutableArray+WeakReferences.h>
```

Class Methods

- (id) + [mutableArrayUsingWeakReferences](#)
Create an NSMutableArray that uses weak references.

- (id) + [mutableArrayUsingWeakReferencesWithCapacity:](#)
Create an NSMutableArray that uses weak references.
- (id) + [newMutableArrayUsingWeakReferences](#)
Create an NSMutableArray that uses weak references (no pending autorelease).
- (id) + [newMutableArrayUsingWeakReferencesWithCapacity:](#)
Create an NSMutableArray that uses weak references (no pending autorelease).

4.18.1 Detailed Description

Adds to NSMutableArray the ability to create an array that keeps weak references.

4.18.2 Method Documentation

4.18.2.1 + (id) mutableArrayUsingWeakReferences

Create an NSMutableArray that uses weak references.

4.18.2.2 + (id) mutableArrayUsingWeakReferencesWithCapacity: (NSUInteger) capacity

Create an NSMutableArray that uses weak references.

Parameters

<i>capacity</i>	The initial capacity of the array.
-----------------	------------------------------------

4.18.2.3 + (id) newMutableArrayUsingWeakReferences

Create an NSMutableArray that uses weak references (no pending autorelease).

4.18.2.4 + (id) newMutableArrayUsingWeakReferencesWithCapacity: (NSUInteger) capacity

Create an NSMutableArray that uses weak references (no pending autorelease).

Parameters

<i>capacity</i>	The initial capacity of the array.
-----------------	------------------------------------

The documentation for this category was generated from the following files:

- NSMutableArray+WeakReferences.h
- NSMutableArray+WeakReferences.m

4.19 NSMutableArrayDictionary(WeakReferences) Category Reference

Class Methods

- (NSMutableDictionary *) + [mutableDictionaryUsingWeakReferences](#)
Create an NSMutableDictionary that uses weak references.
- (NSMutableDictionary *) + [mutableDictionaryUsingWeakReferencesWithCapacity:](#)
Create an NSMutableDictionary that uses weak references.

- (NSMutableDictionary *) + [newMutableDictionaryUsingWeakReferences](#)
Create an NSMutableDictionary that uses weak references (no pending autorelease).
- (NSMutableDictionary *) + [newMutableDictionaryUsingWeakReferencesWithCapacity:](#)
Create an NSMutableDictionary that uses weak references (no pending autorelease).

4.19.1 Method Documentation

4.19.1.1 + (NSMutableDictionary *) mutableDictionaryUsingWeakReferences

Create an NSMutableDictionary that uses weak references.

4.19.1.2 + (NSMutableDictionary *) mutableDictionaryUsingWeakReferencesWithCapacity: (NSUInteger) capacity

Create an NSMutableDictionary that uses weak references.

Parameters

<i>capacity</i>	The initial capacity of the dictionary.
-----------------	---

4.19.1.3 + (NSMutableDictionary *) newMutableDictionaryUsingWeakReferences

Create an NSMutableDictionary that uses weak references (no pending autorelease).

4.19.1.4 + (NSMutableDictionary *) newMutableDictionaryUsingWeakReferencesWithCapacity: (NSUInteger) capacity

Create an NSMutableDictionary that uses weak references (no pending autorelease).

Parameters

<i>capacity</i>	The initial capacity of the dictionary.
-----------------	---

The documentation for this category was generated from the following files:

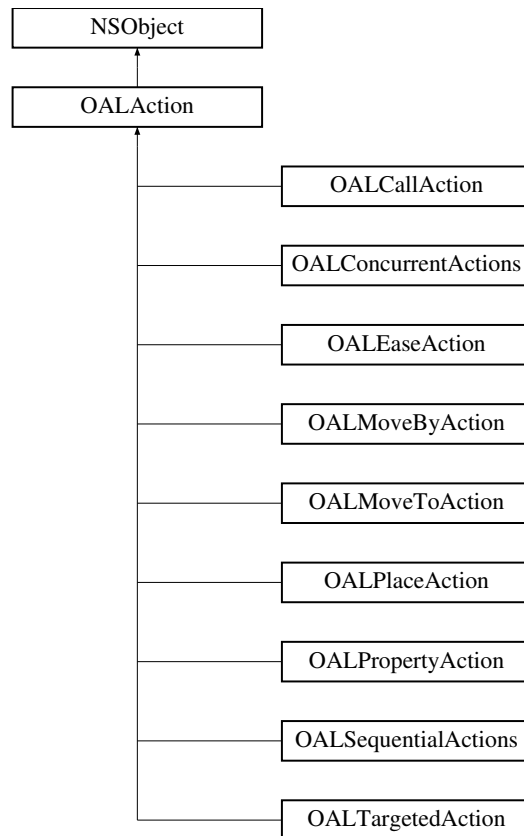
- NSMutableDictionary+WeakReferences.h
- NSMutableDictionary+WeakReferences.m

4.20 OALAction Class Reference

Represents an action that can be performed on an object.

```
#import <OALAction.h>
```

Inheritance diagram for OALAction:



Instance Methods

- (id) - [initWithDuration:](#)
Initialize an action.
- (void) - [runWithTarget:](#)
Run this action on a target.
- (void) - [prepareWithTarget:](#)
Called by runWithTarget to do any final preparations before running.
- (void) - [startAction](#)
Called by runWithTarget to start the action running.
- (void) - [updateCompletion:](#)
Called by [OALActionManager](#) to update this action's progress.
- (void) - [stopAction](#)
Stop this action.

Protected Attributes

- bool [runningInManager_](#)
If TRUE, this action is running via [OALActionManager](#).

Properties

- id [target](#)
The target to perform the action on.
- float [duration](#)

The duration of the action, in seconds.

- float `elapsed`

The amount of time that has elapsed for this action, in seconds.

- bool `running`

If true, the action is currently running.

4.20.1 Detailed Description

Represents an action that can be performed on an object.

4.20.2 Method Documentation

4.20.2.1 - (id) initWithDuration: (float) *duration*

Initialize an action.

Parameters

<i>duration</i>	The duration of this action in seconds.
-----------------	---

Returns

The initialized action.

4.20.2.2 - (void) prepareWithTarget: (id) *target*

Called by runWithTraget to do any final preparations before running.

Subclasses must ensure that duration is valid when this method returns.

Parameters

<i>target</i>	The target to run the action on.
---------------	----------------------------------

4.20.2.3 - (void) runWithTarget: (id) *target*

Run this action on a target.

Parameters

<i>target</i>	The target to run the action on.
---------------	----------------------------------

4.20.2.4 - (void) startAction

Called by runWithTarget to start the action running.

4.20.2.5 - (void) stopAction

Stop this action.

4.20.2.6 - (void) updateCompletion: (float) *proportionComplete*

Called by [OALActionManager](#) to update this action's progress.

Parameters

<i>proportion-Complete</i>	The proportion of this action's duration that has elapsed.
----------------------------	--

4.20.3 Member Data Documentation

4.20.3.1 - (bool) runningInManager_ [protected]

If TRUE, this action is running via [OALActionManager](#).

4.20.4 Property Documentation

4.20.4.1 - (float) duration [read], [nonatomic], [assign]

The duration of the action, in seconds.

4.20.4.2 - (float) elapsed [read], [write], [nonatomic], [assign]

The amount of time that has elapsed for this action, in seconds.

4.20.4.3 - (bool) running [read], [nonatomic], [assign]

If true, the action is currently running.

4.20.4.4 - (id) target [read], [nonatomic], [assign]

The target to perform the action on.

WEAK REFERENCE.

The documentation for this class was generated from the following files:

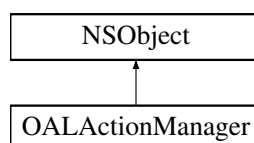
- OALAction.h
- OALAction.m

4.21 OALActionManager Class Reference

Manages all ObjectAL actions.

```
#import <OALActionManager.h>
```

Inheritance diagram for OALActionManager:



Instance Methods

- (void) - [stopAllActions](#)
Stops ALL running actions on ALL targets.

Protected Member Functions

- () - [SYNTHESIZE_SINGLETON_FOR_CLASS_HEADER](#)
Singleton implementation providing "sharedInstance" and "purgeSharedInstance" methods.

Protected Attributes

- NSMutableArray * [targets](#)
All targets that have actions running on them (id).
- NSMutableArray * [targetActions](#)
Parallel array to "targets", maintaining a list of all actions per target (NSMutableArray)*
- NSMutableArray * [actionsToAdd](#)
All actions that are to be added on the next pass (OALAction)*
- NSMutableArray * [actionsToRemove](#)
All actions that are to be removed on the next pass (OALAction)*
- NSTimer * [stepTimer](#)
The timer which we use to update the actions.
- uint64_t [lastTimestamp](#)
The last time that was recorded.

4.21.1 Detailed Description

Manages all ObjectAL actions.

4.21.2 Method Documentation

4.21.2.1 - (void) stopAllActions

Stops ALL running actions on ALL targets.

4.21.2.2 - OALActionManager: (OALActionManager)

Singleton implementation providing "sharedInstance" and "purgeSharedInstance" methods.

- (OALAudioSupport*) **sharedInstance**: Get the shared singleton instance.

- (void) **purgeSharedInstance**: Purge (deallocate) the shared instance.

4.21.3 Member Data Documentation

4.21.3.1 - (NSMutableArray*) actionsToAdd [protected]

All actions that are to be added on the next pass (OALAction*)

4.21.3.2 - (NSMutableArray*) actionsToRemove [protected]

All actions that are to be removed on the next pass (OALAction*)

4.21.3.3 - (uint64_t) lastTimestamp [protected]

The last time that was recorded.

4.21.3.4 - (NSTimer*) stepTimer [protected]

The timer which we use to update the actions.

4.21.3.5 - (NSMutableArray*) targetActions [protected]

Parallel array to "targets", maintaining a list of all actions per target (NSMutableArray*)

4.21.3.6 - (NSMutableArray*) targets [protected]

All targets that have actions running on them (id).

The documentation for this class was generated from the following files:

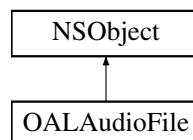
- OALActionManager.h
- OALActionManager.m

4.22 OALAudioFile Class Reference

Maintains an open audio file and allows loading data from that file into new [ALBuffer](#) objects.

```
#import <OALAudioFile.h>
```

Inheritance diagram for OALAudioFile:



Instance Methods

- (id) - [initWithUrl:reduceToMono:](#)
Initialize this object with the audio file at the specified URL.
- (void *) - [audioDataWithStartFrame:numFrames:bufferSize:](#)
Read audio data from this file into a new buffer.
- ([ALBuffer](#) *) - [bufferNamed:startFrame:numFrames:](#)
Create a new [ALBuffer](#) with the contents of this file.

Class Methods

- ([OALAudioFile](#) *) + [fileWithUrl:reduceToMono:](#)
Open the audio file at the specified URL.
- ([ALBuffer](#) *) + [bufferFromUrl:reduceToMono:](#)
Convenience method to load the entire contents of a URL into a new [ALBuffer](#).

Protected Attributes

- `AudioStreamBasicDescription` [streamDescription](#)
A description of the audio data in this file.
- `ExtAudioFileRef` [fileHandle](#)
The OS specific file handle.
- `UInt32` [originalChannelsPerFrame](#)
The actual number of channels in the audio data if not reducing to mono.

Properties

- `NSURL *` [url](#)
The URL of the audio file.
- `AudioStreamBasicDescription *` [streamDescription](#)
A description of the audio data in this file.
- `SInt64` [totalFrames](#)
The total number of audio frames in this file.
- `bool` [reduceToMono](#)
If YES, reduce any stereo data to mono (stereo samples don't support panning or positional audio).

4.22.1 Detailed Description

Maintains an open audio file and allows loading data from that file into new [ALBuffer](#) objects.

4.22.2 Method Documentation

4.22.2.1 - (void *) `audioDataWithStartFrame: (SInt64) startFrame numFrames:(SInt64) numFrames bufferSize:(UInt32*) bufferSize`

Read audio data from this file into a new buffer.

Parameters

<i>startFrame</i>	The starting audio frame to read data from.
<i>numFrames</i>	The number of frames to read.
<i>bufferSize</i>	On successful return, contains the size of the returned buffer, in bytes.

Returns

The audio data or nil on error. You are responsible for calling `free()` on the data.

4.22.2.2 + (ALBuffer *) `bufferFromUrl: (NSURL*) url reduceToMono:(bool) reduceToMono`

Convenience method to load the entire contents of a URL into a new [ALBuffer](#).

Parameters

<i>url</i>	The URL to open the audio file from.
<i>reduceToMono</i>	If YES, reduce any stereo track to mono (stereo samples don't support panning or positional audio).

Returns

an [ALBuffer](#) object.

4.22.2.3 - (ALBuffer *) bufferNamed: (NSString*) *name* startFrame:(SInt64) *startFrame* numFrames:(SInt64) *numFrames*

Create a new [ALBuffer](#) with the contents of this file.

Parameters

<i>name</i>	The name to be given to this ALBuffer .
<i>startFrame</i>	The starting audio frame to read data from.
<i>numFrames</i>	The number of frames to read.

Returns

a new [ALBuffer](#) containing the audio data.

4.22.2.4 + (OALAudioFile *) fileWithURL: (NSURL*) *url* reduceToMono:(bool) *reduceToMono*

Open the audio file at the specified URL.

Parameters

<i>url</i>	The URL to open the audio file from.
<i>reduceToMono</i>	If YES, reduce any stereo track to mono (stereo samples don't support panning or positional audio).

Returns

a new audio file object.

4.22.2.5 - (id) initWithUrl: (NSURL*) *url* reduceToMono:(bool) *reduceToMono*

Initialize this object with the audio file at the specified URL.

Parameters

<i>url</i>	The URL to open the audio file from.
<i>reduceToMono</i>	If YES, reduce any stereo track to mono (stereo samples don't support panning or positional audio).

Returns

the initialized audio file object.

4.22.3 Member Data Documentation

4.22.3.1 - (ExtAudioFileRef) fileHandle [protected]

The OS specific file handle.

4.22.3.2 - (UInt32) **originalChannelsPerFrame** [protected]

The actual number of channels in the audio data if not reducing to mono.

4.22.3.3 - (AudioStreamBasicDescription) **streamDescription** [protected]

A description of the audio data in this file.

4.22.4 Property Documentation

4.22.4.1 - (bool) **reduceToMono** [read],[write],[nonatomic],[assign]

If YES, reduce any stereo data to mono (stereo samples don't support panning or positional audio).

4.22.4.2 - (AudioStreamBasicDescription *) **streamDescription** [read],[nonatomic],[assign]

A description of the audio data in this file.

4.22.4.3 - (SInt64) **totalFrames** [read],[nonatomic],[assign]

The total number of audio frames in this file.

4.22.4.4 - (NSURL *) **url** [read],[nonatomic],[retain]

The URL of the audio file.

The documentation for this class was generated from the following files:

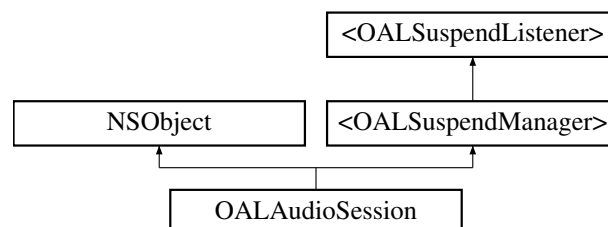
- OALAudioFile.h
- OALAudioFile.m

4.23 OALAudioSession Class Reference

Handles the audio session and interrupts.

```
#import <OALAudioSession.h>
```

Inheritance diagram for OALAudioSession:



Instance Methods

- (void) - [forceEndInterruption](#)
Force an interrupt end.

Protected Member Functions

- `()` - [SYNTHESIZE_SINGLETON_FOR_CLASS_HEADER](#)
Singleton implementation providing "sharedInstance" and "purgeSharedInstance" methods.

Protected Attributes

- `bool` [handlingErrorNotification](#)
Flag signifying that we are currently handling an error notification.
- `bool` [audioSessionWasActive](#)
If true, the audio session was active when the interrupt occurred.
- `OALSuspendHandler *` [suspendHandler](#)
Handles suspending and interrupting for this object.
- `NSDate *` [lastResetTime](#)
Marks the last time the audio session was reset due to error.

Properties

- `NSString *` [audioSessionCategory](#)
The current audio session category.
- `bool` [allowIpod](#)
If YES, allow ipod music to continue playing (NOT SUPPORTED ON THE SIMULATOR).
- `bool` [ipodDucking](#)
If YES, ipod music will duck (lower in volume) when the audio session activates.
- `bool` [useHardwareIfAvailable](#)
Determines what to do if no other application is playing audio and allowIpod = YES (NOT SUPPORTED ON THE SIMULATOR).
- `bool` [honorSilentSwitch](#)
If true, mute when backgrounded, screen locked, or the ringer switch is turned off (NOT SUPPORTED ON THE SIMULATOR).
- `bool` [handleInterruptions](#)
If true, automatically handle interruptions.
- `float` [preferredIOBufferDuration](#)
The preferred I/O buffer duration, in seconds.
- `bool` [ipodPlaying](#)
If true, another application (usually iPod) is playing music.
- `bool` [audioSessionActive](#)
If true, the audio session is active.
- `float` [hardwareVolume](#)
Get the device's final hardware output volume, as controlled by the volume button on the side of the device.
- `bool` [hardwareMuted](#)
Check if the hardware mute switch is on (not supported on the simulator or iOS 5+).
- `NSString *` [audioRoute](#)
Check what hardware route the audio is taking, such as "Speaker" or "Headphone" (not supported on the simulator).

4.23.1 Detailed Description

Handles the audio session and interrupts.

4.23.2 Method Documentation

4.23.2.1 - (void) forceEndInterruption

Force an interrupt end.

This can be useful in cases where a buggy OS fails to end an interrupt.

Be VERY CAREFUL when using this!

4.23.2.2 - OALAudioSession: (OALAudioSession)

Singleton implementation providing "sharedInstance" and "purgeSharedInstance" methods.

- (OALAudioSupport*) **sharedInstance**: Get the shared singleton instance.

- (void) **purgeSharedInstance**: Purge (deallocate) the shared instance.

4.23.3 Member Data Documentation

4.23.3.1 - (bool) audioSessionWasActive [protected]

If true, the audio session was active when the interrupt occurred.

4.23.3.2 - (bool) handlingErrorNotification [protected]

Flag signifying that we are currently handling an error notification.

This prevents onAudioError: from becoming reentrant due to self.manuallySuspended setting off a chain of calls that result in another error notification broadcast.

4.23.3.3 - (NSDate*) lastResetTime [protected]

Marks the last time the audio session was reset due to error.

This is used to avoid getting stuck in a rapid-fire reset-error loop.

4.23.3.4 - (OALSuspendHandler*) suspendHandler [protected]

Handles suspending and interrupting for this object.

4.23.4 Property Documentation

4.23.4.1 - (bool) allowIpod [read], [write], [nonatomic], [assign]

If YES, allow ipod music to continue playing (NOT SUPPORTED ON THE SIMULATOR).

Note: If this is enabled, and another app is playing music, background audio playback will use the SOFTWARE codecs, NOT hardware.

If allowIpod = NO, the application will ALWAYS use hardware decoding.

See Also

[useHardwareIfAvailable](#)

Default value: YES

4.23.4.2 - (NSString*) audioRoute [read], [nonatomic], [retain]

Check what hardware route the audio is taking, such as "Speaker" or "Headphone" (not supported on the simulator).

4.23.4.3 - (bool) audioSessionActive [read], [write], [nonatomic], [assign]

If true, the audio session is active.

4.23.4.4 - (NSString *) audioSessionCategory [read], [write], [nonatomic], [retain]

The current audio session category.

If this value is explicitly set, the other session properties "allowIpad", "useHardwareIfAvailable", "honorSilentSwitch", and "ipodDucking" may be modified to remain compatible with the category.

See Also

AVAudioSessionCategory

Default value: nil

4.23.4.5 - (bool) handleInterruptions [read], [write], [nonatomic], [assign]

If true, automatically handle interruptions.

Default value: YES

4.23.4.6 - (bool) hardwareMuted [read], [nonatomic], [assign]

Check if the hardware mute switch is on (not supported on the simulator or iOS 5+).

Note: If headphones are plugged in, hardwareMuted will always return FALSE regardless of the switch state.

Note: Please file a bug report with Apple to get this functionality restored in iOS 5!

4.23.4.7 - (float) hardwareVolume [read], [nonatomic], [assign]

Get the device's final hardware output volume, as controlled by the volume button on the side of the device.

4.23.4.8 - (bool) honorSilentSwitch [read], [write], [nonatomic], [assign]

If true, mute when backgrounded, screen locked, or the ringer switch is turned off (NOT SUPPORTED ON THE SIMULATOR).

Default value: YES

4.23.4.9 - (bool) ipodDucking [read], [write], [nonatomic], [assign]

If YES, ipod music will duck (lower in volume) when the audio session activates.

Default value: NO

4.23.4.10 - (bool) ipodPlaying [read], [nonatomic], [assign]

If true, another application (usually iPod) is playing music.

4.23.4.11 - (float) `preferredIOBufferDuration` [read],[write],[nonatomic],[assign]

The preferred I/O buffer duration, in seconds.

Lower values give less playback latency, but use more CPU.

4.23.4.12 - (bool) `useHardwareIfAvailable` [read],[write],[nonatomic],[assign]

Determines what to do if no other application is playing audio and `allowIpod` = YES (NOT SUPPORTED ON THE SIMULATOR).

If NO, the application will ALWAYS use software decoding. The advantage to this is that the user can background your application and then start audio playing from another application. If `useHardwareIfAvailable` = YES, the user won't be able to do this.

If this is set to YES, the application will use hardware decoding if no other application is currently playing audio. However, no other application will be able to start playing audio if it wasn't playing already.

Note: This switch has no effect if `allowIpod` = NO.

See Also

[allowIpod](#)

Default value: YES

The documentation for this class was generated from the following files:

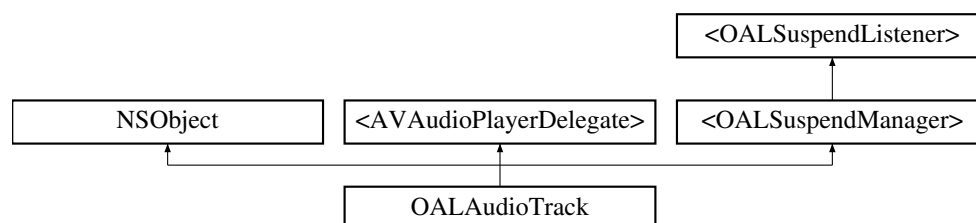
- OALAudioSession.h
- OALAudioSession.m

4.24 OALAudioTrack Class Reference

Plays an audio track via AVAudioPlayer.

```
#import <OALAudioTrack.h>
```

Inheritance diagram for OALAudioTrack:



Instance Methods

- (bool) - [preloadUrl:](#)
Preload the contents of a URL for playback.
- (bool) - [preloadUrl:seekTime:](#)
Preload the contents of a URL for playback.
- (bool) - [preloadFile:](#)
Preload the contents of a file for playback.
- (bool) - [preloadFile:seekTime:](#)

- Preload the contents of a file for playback.*

 - (bool) - [preloadUrlAsync:target:selector:](#)
Asynchronously preload the contents of a URL for playback.
 - (bool) - [preloadUrlAsync:seekTime:target:selector:](#)
Asynchronously preload the contents of a URL for playback.
 - (bool) - [preloadFileAsync:target:selector:](#)
Asynchronously preload the contents of a file for playback.
 - (bool) - [preloadFileAsync:seekTime:target:selector:](#)
Asynchronously preload the contents of a file for playback.
 - (bool) - [playUrl:](#)
Play the contents of a URL once.
 - (bool) - [playUrl:loops:](#)
Play the contents of a URL and loop the specified number of times.
 - (bool) - [playFile:](#)
Play the contents of a file once.
 - (bool) - [playFile:loops:](#)
Play the contents of a file and loop the specified number of times.
 - (void) - [playUrlAsync:target:selector:](#)
Play the contents of a URL asynchronously once.
 - (void) - [playUrlAsync:loops:target:selector:](#)
Play the contents of a URL asynchronously and loop the specified number of times.
 - (void) - [playFileAsync:target:selector:](#)
Play the contents of a file asynchronously once.
 - (void) - [playFileAsync:loops:target:selector:](#)
Play the contents of a file asynchronously and loop the specified number of times.
 - (bool) - [play](#)
Play the currently loaded audio track.
 - (bool) - [playAtTime:](#)
Plays a sound asynchronously, starting at a specified point in the audio output device's timeline.
 - (bool) - [playAfterTrack:](#)
Plays the currently preloaded track asynchronously when the specified track completes.
 - (bool) - [playAfterTrack:timeAdjust:](#)
Plays the currently preloaded track asynchronously when the specified track completes.
 - (void) - [stop](#)
Stop playing and stop all operations.
 - (void) - [fadeTo:duration:target:selector:](#)
Fade to the specified gain value.
 - (void) - [stopFade](#)
Stop the currently running fade operation, if any.
 - (void) - [panTo:duration:target:selector:](#)
Pan to the specified pan value.
 - (void) - [stopPan](#)
Stop the currently running pan operation, if any.
 - (void) - [stopActions](#)
Stop any internal fade or pan actions.
 - (void) - [clear](#)
Unload and clear all audio data, stop playing, and stop all operations.
 - (void) - [updateMeters](#)
Updates the metering system to give current values.
 - (float) - [averagePowerForChannel:](#)
Gives the average power for a given channel, in decibels, for the sound being played.
 - (float) - [peakPowerForChannel:](#)
Gives the peak power for a given channel, in decibels, for the sound being played.

Class Methods

- (id) + [track](#)
Create a new audio track.

Protected Attributes

- bool **interrupted**
- AVAudioPlayer * [simulatorPlayerRef](#)
When the simulator is running (and the playback fix is in use), player will be copied to here, and then player set to nil.
- NSOperationQueue * [operationQueue](#)
Operation queue for running asynchronous operations.
- [OALAction](#) * [gainAction](#)
The current action being applied to gain.
- [OALAction](#) * [panAction](#)
The current action being applied to pan.
- [OALSuspendHandler](#) * [suspendHandler](#)
Handles suspending and interrupting for this object.

Properties

- NSURL * [currentlyLoadedUrl](#)
The URL of the currently loaded audio data.
- id< AVAudioPlayerDelegate > [delegate](#)
Optional object that will receive notifications for decoding errors, audio interruptions (such as an incoming phone call), and playback completion.
- float [gain](#)
The gain (volume) for playback (0.0 - 1.0, where 1.0 = no attenuation).
- float [volume](#)
The volume (alias to gain) for playback (0.0 - 1.0, where 1.0 = no attenuation).
- float [pan](#)
Pan value (-1.0 = far left, 1.0 = far right).
- bool [muted](#)
If true, audio track is muted.
- bool [autoPreload](#)
If true, automatically preload again when playback stops.
- bool [preloaded](#)
If true, audio track is in preloaded state.
- NSInteger [numberOfLoops](#)
The number of times to loop playback (-1 = forever).
- bool [paused](#)
If true, pause playback.
- AVAudioPlayer * [player](#)
Access to the underlying AVAudioPlayer object.
- bool [playing](#)
If true, the audio player is currently playing.
- NSTimeInterval [currentTime](#)
The current playback position in seconds from the start of the sound.
- NSTimeInterval [deviceCurrentTime](#)
The value of this property increases monotonically while an audio player is playing or paused.
- NSTimeInterval [duration](#)

The duration, in seconds, of the currently loaded sound.

- NSInteger [numberOfChannels](#)

The number of channels in the currently loaded sound.

- bool [meteringEnabled](#)

If true, this track is recording metering data.

4.24.1 Detailed Description

Plays an audio track via AVAudioPlayer.

Unlike AVAudioPlayer, however, it can be re-used to play another file. Interruptions can be handled by OALAudioSupport (enabled by default).

4.24.2 Method Documentation

4.24.2.1 - (float) averagePowerForChannel: (NSInteger) channelNumber

Gives the average power for a given channel, in decibels, for the sound being played.

0 dB indicates maximum power (full scale).

-160 dB indicates minimum power (near silence).

If the signal provided to the audio player exceeds full scale, then the value may be > 0.

Note: The value returned is in reference to when updateMeters was last called. You must call updateMeters again before calling this method to get a current value.

Parameters

<i>channelNumber</i>	The channel to get the value from. For mono or left, use 0. For right, use 1.
----------------------	---

Returns

the average power for the channel.

4.24.2.2 - (void) clear

Unload and clear all audio data, stop playing, and stop all operations.

4.24.2.3 - (void) fadeTo: (float) gain duration:(float) duration target:(id) target selector:(SEL) selector

Fade to the specified gain value.

Parameters

<i>gain</i>	The gain to fade to.
<i>duration</i>	The duration of the fade operation in seconds.
<i>target</i>	The target to notify when the fade completes (can be nil).
<i>selector</i>	The selector to call when the fade completes. The selector must accept a single parameter, which will be the object that performed the fade.

4.24.2.4 - (void) panTo: (float) pan duration:(float) duration target:(id) target selector:(SEL) selector

Pan to the specified pan value.

Note: This will have no effect on iOS versions prior to 4.0.

Parameters

<i>pan</i>	The value to pan to.
<i>duration</i>	The duration of the pan operation in seconds.
<i>target</i>	The target to notify when the pan completes (can be nil).
<i>selector</i>	The selector to call when the pan completes. The selector must accept a single parameter, which will be the object that performed the pan.

4.24.2.5 - (float) *peakPowerForChannel:* (NSUInteger) *channelNumber*

Gives the peak power for a given channel, in decibels, for the sound being played.

0 dB indicates maximum power (full scale).

-160 dB indicates minimum power (near silence).

If the signal provided to the audio player exceeds full scale, then the value may be > 0.

Note: The value returned is in reference to when `updateMeters` was last called. You must call `updateMeters` again before calling this method to get a current value.

Parameters

<i>channelNumber</i>	The channel to get the value from. For mono or left, use 0. For right, use 1.
----------------------	---

Returns

the average power for the channel.

4.24.2.6 - (bool) *play*

Play the currently loaded audio track.

Returns

TRUE if the operation was successful.

4.24.2.7 - (bool) *playAfterTrack:* (OALAudioTrack*) *track*

Plays the currently preloaded track asynchronously when the specified track completes.

Note: This will have no effect on iOS versions prior to 4.0.

Parameters

<i>track</i>	The track to play after
--------------	-------------------------

Returns

YES if the playback was successfully scheduled.

4.24.2.8 - (bool) *playAfterTrack:* (OALAudioTrack*) *track* *timeAdjust:(NSTimeInterval)* *timeAdjust*

Plays the currently preloaded track asynchronously when the specified track completes.

Note: This will have no effect on iOS versions prior to 4.0.

Parameters

<i>track</i>	The track to play after
<i>timeAdjust</i>	fine-tune value added to the time start offset.

Returns

YES if the playback was successfully scheduled.

4.24.2.9 - (bool) playAtTime: (NSTimeInterval) time

Plays a sound asynchronously, starting at a specified point in the audio output device's timeline.

Note: This will have no effect on iOS versions prior to 4.0.

Parameters

<i>time</i>	The time (device time) to start playing at.
-------------	---

Returns

YES if the playback was successfully scheduled.

4.24.2.10 - (bool) playFile: (NSString*) path

Play the contents of a file once.

Parameters

<i>path</i>	The file containing the sound data.
-------------	-------------------------------------

Returns

TRUE if the operation was successful.

4.24.2.11 - (bool) playFile: (NSString*) path loops:(NSInteger) loops

Play the contents of a file and loop the specified number of times.

Parameters

<i>path</i>	The file containing the sound data.
<i>loops</i>	The number of times to loop playback (-1 = forever)

Returns

TRUE if the operation was successful.

4.24.2.12 - (void) playFileAsync: (NSString*) path loops:(NSInteger) loops target:(id) target selector:(SEL) selector

Play the contents of a file asynchronously and loop the specified number of times.

Parameters

<i>path</i>	The file containing the sound data.
<i>loops</i>	The number of times to loop playback (-1 = forever)
<i>target</i>	the target to inform when playing has started.
<i>selector</i>	the selector to call when playing has started.

4.24.2.13 - (void) playFileAsync: (NSString*) *path* target:(id) *target* selector:(SEL) *selector*

Play the contents of a file asynchronously once.

Parameters

<i>path</i>	The file containing the sound data.
<i>target</i>	the target to inform when playing has started.
<i>selector</i>	the selector to call when playing has started.

4.24.2.14 - (bool) playUrl: (NSURL*) *url*

Play the contents of a URL once.

Parameters

<i>url</i>	The URL containing the sound data.
------------	------------------------------------

Returns

TRUE if the operation was successful.

4.24.2.15 - (bool) playUrl: (NSURL*) *url* loops:(NSInteger) *loops*

Play the contents of a URL and loop the specified number of times.

Parameters

<i>url</i>	The URL containing the sound data.
<i>loops</i>	The number of times to loop playback (-1 = forever)

Returns

TRUE if the operation was successful.

4.24.2.16 - (void) playUrlAsync: (NSURL*) *url* loops:(NSInteger) *loops* target:(id) *target* selector:(SEL) *selector*

Play the contents of a URL asynchronously and loop the specified number of times.

Parameters

<i>url</i>	The URL containing the sound data.
<i>loops</i>	The number of times to loop playback (-1 = forever)
<i>target</i>	the target to inform when playing has started.
<i>selector</i>	the selector to call when playing has started.

4.24.2.17 - (void) playUrlAsync: (NSURL*) url target:(id) target selector:(SEL) selector

Play the contents of a URL asynchronously once.

Parameters

<i>url</i>	The URL containing the sound data.
<i>target</i>	the target to inform when playing has started.
<i>selector</i>	the selector to call when playing has started.

4.24.2.18 - (bool) preloadFile: (NSString*) path

Preload the contents of a file for playback.

Once the audio data is preloaded, you can call "play" to play it.

Parameters

<i>path</i>	The file containing the sound data.
-------------	-------------------------------------

Returns

TRUE if the operation was successful.

4.24.2.19 - (bool) preloadFile: (NSString*) path seekTime:(NSTimeInterval) seekTime

Preload the contents of a file for playback.

Once the audio data is preloaded, you can call "play" to play it.

Parameters

<i>path</i>	The file containing the sound data.
<i>seekTime</i>	The position in the file to start playing at.

Returns

TRUE if the operation was successful.

4.24.2.20 - (bool) preloadFileAsync: (NSString*) path seekTime:(NSTimeInterval) seekTime target:(id) target selector:(SEL) selector

Asynchronously preload the contents of a file for playback.

Once the audio data is preloaded, you can call "play" to play it.

Parameters

<i>path</i>	The file containing the sound data.
<i>seekTime</i>	The position in the file to start playing at.
<i>target</i>	the target to inform when preparation is complete.
<i>selector</i>	the selector to call when preparation is complete.

Returns

TRUE if the operation was successfully queued.

4.24.2.21 - (bool) preloadFileAsync: (NSString*) *path* target:(id) *target* selector:(SEL) *selector*

Asynchronously preload the contents of a file for playback.

Once the audio data is preloaded, you can call "play" to play it.

Parameters

<i>path</i>	The file containing the sound data.
<i>target</i>	the target to inform when preparation is complete.
<i>selector</i>	the selector to call when preparation is complete.

Returns

TRUE if the operation was successfully queued.

4.24.2.22 - (bool) preloadUrl: (NSURL*) *url*

Preload the contents of a URL for playback.

Once the audio data is preloaded, you can call "play" to play it.

Parameters

<i>url</i>	The URL containing the sound data.
------------	------------------------------------

Returns

TRUE if the operation was successful.

4.24.2.23 - (bool) preloadUrl: (NSURL*) *url* seekTime:(NSTimeInterval) *seekTime*

Preload the contents of a URL for playback.

Once the audio data is preloaded, you can call "play" to play it.

Parameters

<i>url</i>	The URL containing the sound data.
<i>seekTime</i>	The position in the file to start playing at.

Returns

TRUE if the operation was successful.

4.24.2.24 - (bool) preloadUrlAsync: (NSURL*) *url* seekTime:(NSTimeInterval) *seekTime* target:(id) *target* selector:(SEL) *selector*

Asynchronously preload the contents of a URL for playback.

Once the audio data is preloaded, you can call "play" to play it.

Parameters

<i>url</i>	The URL containing the sound data.
<i>seekTime</i>	The position in the file to start playing at.
<i>target</i>	the target to inform when preparation is complete.
<i>selector</i>	the selector to call when preparation is complete.

Returns

TRUE if the operation was successfully queued.

4.24.2.25 - (bool) preloadUrlAsync: (NSURL*) url target:(id) target selector:(SEL) selector

Asynchronously preload the contents of a URL for playback.

Once the audio data is preloaded, you can call "play" to play it.

Parameters

<i>url</i>	The URL containing the sound data.
<i>target</i>	the target to inform when preparation is complete.
<i>selector</i>	the selector to call when preparation is complete.

Returns

TRUE if the operation was successfully queued.

4.24.2.26 - (void) stop

Stop playing and stop all operations.

4.24.2.27 - (void) stopActions

Stop any internal fade or pan actions.

4.24.2.28 - (void) stopFade

Stop the currently running fade operation, if any.

4.24.2.29 - (void) stopPan

Stop the currently running pan operation, if any.

Note: This will have no effect on iOS versions prior to 4.0.

4.24.2.30 + (id) track

Create a new audio track.

Returns

A new audio track.

4.24.2.31 - (void) updateMeters

Updates the metering system to give current values.

You must call this method before calling `averagePowerForChannel` or `peakPowerForChannel` in order to get current values.

4.24.3 Member Data Documentation

4.24.3.1 - (OALAction*) gainAction [protected]

The current action being applied to gain.

4.24.3.2 - (NSOperationQueue*) operationQueue [protected]

Operation queue for running asynchronous operations.

Note: Only one asynchronous operation is allowed at a time.

4.24.3.3 - (OALAction*) panAction [protected]

The current action being applied to pan.

4.24.3.4 - (AVAudioPlayer*) simulatorPlayerRef [protected]

When the simulator is running (and the playback fix is in use), player will be copied to here, and then player set to nil.

This prevents other code from inadvertently raising the volume and starting playback.

4.24.3.5 - (OALSuspendHandler*) suspendHandler [protected]

Handles suspending and interrupting for this object.

4.24.4 Property Documentation

4.24.4.1 - (bool) autoPreload [read], [write], [nonatomic], [assign]

If true, automatically preload again when playback stops.

4.24.4.2 - (NSURL *) currentlyLoadedUrl [read], [nonatomic], [retain]

The URL of the currently loaded audio data.

4.24.4.3 - (NSTimeInterval) currentTime [read], [write], [nonatomic], [assign]

The current playback position in seconds from the start of the sound.

You can set this to change the playback position, whether it is currently playing or not.

4.24.4.4 - (id< AVAudioPlayerDelegate >) **delegate** [read],[write],[nonatomic],[assign]

Optional object that will receive notifications for decoding errors, audio interruptions (such as an incoming phone call), and playback completion.

Note: [OALAudioTrack](#) keeps a WEAK reference to delegate, so make sure you clear it when your object is going to be deallocated.

4.24.4.5 - (NSTimeInterval) **deviceCurrentTime** [read],[nonatomic],[assign]

The value of this property increases monotonically while an audio player is playing or paused.

If more than one audio player is connected to the audio output device, device time continues incrementing as long as at least one of the players is playing or paused.

If the audio output device has no connected audio players that are either playing or paused, device time reverts to 0.

Use this property to indicate “now” when calling the `playAtTime:` instance method. By configuring multiple audio players to play at a specified offset from `deviceCurrentTime`, you can perform precise synchronization—as described in the discussion for that method.

Note: This will have no effect on iOS versions prior to 4.0.

4.24.4.6 - (NSTimeInterval) **duration** [read],[nonatomic],[assign]

The duration, in seconds, of the currently loaded sound.

4.24.4.7 - (float) **gain** [read],[write],[nonatomic],[assign]

The gain (volume) for playback (0.0 - 1.0, where 1.0 = no attenuation).

4.24.4.8 - (bool) **meteringEnabled** [read],[write],[nonatomic],[assign]

If true, this track is recording metering data.

If true, metering is enabled.

4.24.4.9 - (bool) **muted** [read],[write],[nonatomic],[assign]

If true, audio track is muted.

4.24.4.10 - (NSInteger) **numberOfChannels** [read],[nonatomic],[assign]

The number of channels in the currently loaded sound.

4.24.4.11 - (NSInteger) **numberOfLoops** [read],[write],[nonatomic],[assign]

The number of times to loop playback (-1 = forever).

Note: This value will be ignored, and get changed when you call the various `playXX` methods. Only “play” will use the current value of “numberOfLoops”.

4.24.4.12 - (float) pan [read], [write], [nonatomic], [assign]

Pan value (-1.0 = far left, 1.0 = far right).

Note: This will have no effect on iOS versions prior to 4.0.

4.24.4.13 - (bool) paused [read], [write], [nonatomic], [assign]

If true, pause playback.

4.24.4.14 - (AVAudioPlayer *) player [read], [nonatomic], [retain]

Access to the underlying AVAudioPlayer object.

WARNING: Be VERY careful when accessing this, as some methods could cause it to fall out of sync with [OALAudioTrack](#) (particularly play/pause/stop methods).

4.24.4.15 - (bool) playing [read], [nonatomic], [assign]

If true, the audio player is currently playing.

If true, background music is currently playing.

We need to maintain our own value because AVAudioPlayer will sometimes say it's not playing when it actually is.

4.24.4.16 - (bool) preloaded [read], [nonatomic], [assign]

If true, audio track is in preloaded state.

4.24.4.17 - (float) volume [read], [write], [nonatomic], [assign]

The volume (alias to gain) for playback (0.0 - 1.0, where 1.0 = no attenuation).

The documentation for this class was generated from the following files:

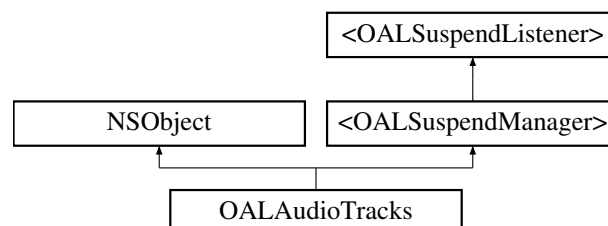
- OALAudioTrack.h
- OALAudioTrack.m

4.25 OALAudioTracks Class Reference

Keeps track of all AudioTrack objects.

```
#import <OALAudioTracks.h>
```

Inheritance diagram for OALAudioTracks:



Instance Methods

- (void) - [stopAllTracks](#)
Stop playback on all audio tracks.
- () - [SYNTHESIZE_SINGLETON_FOR_CLASS_HEADER](#)
Singleton implementation providing "sharedInstance" and "purgeSharedInstance" methods.

Protected Attributes

- NSMutableArray * [tracks](#)
All instantiated audio tracks.
- NSTimer * [deviceTimePoller](#)
Timer to poll deviceCurrentTime so that it doesn't get reset on a device.
- [OALSuspendHandler](#) * [suspendHandler](#)
Handles suspending and interrupting for this object.

Properties

- bool [paused](#)
Pauses/unpauses all audio tracks.
- bool [muted](#)
Mutes/unmutes all audio tracks.
- NSArray * [tracks](#)
All instantiated audio tracks.

4.25.1 Detailed Description

Keeps track of all AudioTrack objects.

4.25.2 Method Documentation

4.25.2.1 - (void) stopAllTracks

Stop playback on all audio tracks.

4.25.2.2 - OALAudioTracks: (OALAudioTracks)

Singleton implementation providing "sharedInstance" and "purgeSharedInstance" methods.

- (OALAudioTracks*) **sharedInstance**: Get the shared singleton instance.

- (void) **purgeSharedInstance**: Purge (deallocate) the shared instance.

4.25.3 Member Data Documentation

4.25.3.1 - (NSTimer*) deviceTimePoller [protected]

Timer to poll deviceCurrentTime so that it doesn't get reset on a device.

4.25.3.2 - (OALSuspendHandler*) suspendHandler [protected]

Handles suspending and interrupting for this object.

4.25.3.3 - (NSMutableArray*) tracks [protected]

All instantiated audio tracks.

4.25.4 Property Documentation

4.25.4.1 - (bool) muted [read], [write], [nonatomic], [assign]

Mutes/unmutes all audio tracks.

4.25.4.2 - (bool) paused [read], [write], [nonatomic], [assign]

Pauses/unpauses all audio tracks.

4.25.4.3 - (NSArray*) tracks [read], [nonatomic], [retain]

All instantiated audio tracks.

The documentation for this class was generated from the following files:

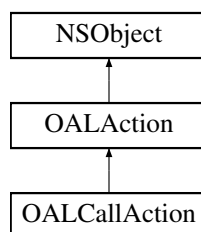
- OALAudioTracks.h
- OALAudioTracks.m

4.26 OALCallAction Class Reference

Calls a selector on a target.

```
#import <OALUtilityActions.h>
```

Inheritance diagram for OALCallAction:



Instance Methods

- (id) - initWithCallTarget:selector:
Initialize an action.
- (id) - initWithCallTarget:selector:withObject:
Initialize an action.
- (id) - initWithCallTarget:selector:withObject:withObject:
Initialize an action.

Class Methods

- (id) + actionWithCallTarget:selector:
Create an action.

- (id) + [actionWithCallTarget:selector:withObject:](#)
Create an action.
- (id) + [actionWithCallTarget:selector:withObject:withObject:](#)
Create an action.

Protected Attributes

- id [callTarget_](#)
The target to call the selector on.
- SEL [selector_](#)
The selector to invoke.
- int [numObjects_](#)
The number of parameters which will be passed to the selector.
- id [object1_](#)
The first object to pass to the selector, if any.
- id [object2_](#)
The second object to pass to the selector, if any.

Additional Inherited Members

4.26.1 Detailed Description

Calls a selector on a target.

This action will ignore whatever target it is run against, and will invoke the selector on the target specified at creation time.

4.26.2 Method Documentation

4.26.2.1 + (id) actionWithCallTarget: (id) *callTarget* selector:(SEL) *selector*

Create an action.

Parameters

<i>callTarget</i>	The target to call.
<i>selector</i>	The selector to invoke.

Returns

A new action.

4.26.2.2 + (id) actionWithCallTarget: (id) *callTarget* selector:(SEL) *selector* withObject:(id) *object*

Create an action.

Parameters

<i>callTarget</i>	The target to call.
<i>selector</i>	The selector to invoke.
<i>object</i>	The object to pass to the selector.

Returns

A new action.

4.26.2.3 + (id) *actionWithCallTarget:* (id) *callTarget* selector:(SEL) *selector* withObject:(id) *firstObject* withObject:(id) *secondObject*

Create an action.

Parameters

<i>callTarget</i>	The target to call.
<i>selector</i>	The selector to invoke.
<i>firstObject</i>	The first object to pass to the selector.
<i>secondObject</i>	The second object to pass to the selector.

Returns

A new action.

4.26.2.4 - (id) *initWithCallTarget:* (id) *callTarget* selector:(SEL) *selector*

Initialize an action.

Parameters

<i>callTarget</i>	The target to call.
<i>selector</i>	The selector to invoke.

Returns

The initialized action.

4.26.2.5 - (id) *initWithCallTarget:* (id) *callTarget* selector:(SEL) *selector* withObject:(id) *object*

Initialize an action.

Parameters

<i>callTarget</i>	The target to call.
<i>selector</i>	The selector to invoke.
<i>object</i>	The object to pass to the selector.

Returns

Initialize an action.

4.26.2.6 - (id) *initWithCallTarget:* (id) *callTarget* selector:(SEL) *selector* withObject:(id) *firstObject* withObject:(id) *secondObject*

Initialize an action.

Parameters

<i>callTarget</i>	The target to call.
<i>selector</i>	The selector to invoke.
<i>firstObject</i>	The first object to pass to the selector.
<i>secondObject</i>	The second object to pass to the selector.

Returns

The initialized action.

4.26.3 Member Data Documentation

4.26.3.1 - (id) callTarget_ [protected]

The target to call the selector on.

4.26.3.2 - (int) numObjects_ [protected]

The number of parameters which will be passed to the selector.

4.26.3.3 - (id) object1_ [protected]

The first object to pass to the selector, if any.

4.26.3.4 - (id) object2_ [protected]

The second object to pass to the selector, if any.

4.26.3.5 - (SEL) selector_ [protected]

The selector to invoke.

The documentation for this class was generated from the following files:

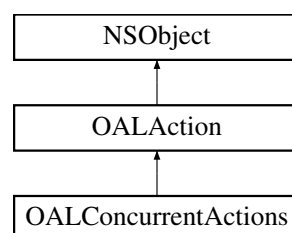
- OALUtilityActions.h
- OALUtilityActions.m

4.27 OALConcurrentActions Class Reference

A set of actions that get run concurrently.

```
#import <OALUtilityActions.h>
```

Inheritance diagram for OALConcurrentActions:



Instance Methods

- (id) - [initWithActions:](#)
Initialize an action.

Class Methods

- (id) + [actions:](#)
Create an action.
- (id) + [actionsFromArray:](#)
Create an action.

Properties

- NSMutableArray * [actions](#)
The actions which will be run.

Additional Inherited Members

4.27.1 Detailed Description

A set of actions that get run concurrently.

4.27.2 Method Documentation

4.27.2.1 + (id) actions: (OALAction*) actions , NS_REQUIRES_NIL_TERMINATION

Create an action.

Parameters

<i>actions</i>	The comma separated list of actions.
<i>NS_REQUIRES_NIL_TERMINATION</i>	List of actions must be terminated by a nil.

Returns

A new set of concurrent actions.

4.27.2.2 + (id) actionsFromArray: (NSArray*) actions

Create an action.

Parameters

<i>actions</i>	The actions to run.
----------------	---------------------

Returns

A new set of concurrent actions.

4.27.2.3 - (id) initWithActions: (NSArray*) actions

Initialize an action.

Parameters

<i>actions</i>	The actions to run.
----------------	---------------------

Returns

The initialized set of concurrent actions.

4.27.3 Property Documentation

4.27.3.1 - (NSMutableArray*) actions [read], [write], [nonatomic], [retain]

The actions which will be run.

The documentation for this class was generated from the following files:

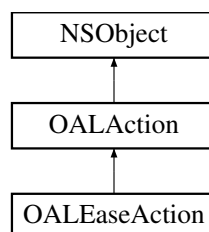
- OALUtilityActions.h
- OALUtilityActions.m

4.28 OALEaseAction Class Reference

Applies an easing function to another action.

```
#import <OALAction.h>
```

Inheritance diagram for OALEaseAction:



Instance Methods

- (id) - [initWithShape:phase:action:](#)
Initialize an ease action.

Class Methods

- (OALEaseAction *) + [actionWithShape:phase:action:](#)
Create a new ease action.
- (EaseFunctionPtr) + [easeFunctionForShape:phase:](#)
Get a pointer to an ease function of the specified shape and phase.

Protected Attributes

- [OALAction](#) * **action_**

Additional Inherited Members

4.28.1 Detailed Description

Applies an easing function to another action.

Normally, an action progresses at a linear rate. An ease changes that to a curve.

4.28.2 Method Documentation

4.28.2.1 + (OALEaseAction *) actionWithShape: (OALEaseShape) *shape* phase:(OALEasePhase) *phase* action:(OALAction*) *action*

Create a new ease action.

Parameters

<i>shape</i>	The shape of the curve to apply.
<i>phase</i>	What phase of the action to apply the curve to. The action to apply the curve to.

Returns

A new action.

4.28.2.2 + (EaseFunctionPtr) easeFunctionForShape: (OALEaseShape) *shape* phase:(OALEasePhase) *phase*

Get a pointer to an ease function of the specified shape and phase.

Parameters

<i>shape</i>	The shape of the curve to apply.
<i>phase</i>	What phase of the action to apply the curve to.

Returns

a pointer to the appropriate function.

4.28.2.3 - (id) initWithShape: (OALEaseShape) *shape* phase:(OALEasePhase) *phase* action:(OALAction*) *action*

Initialize an ease action.

Parameters

<i>shape</i>	The shape of the curve to apply.
<i>phase</i>	What phase of the action to apply the curve to. The action to apply the curve to.

Returns

The initialized action.

The documentation for this class was generated from the following files:

- OALAction.h
- OALAction.m

4.29 OALEaseAction() Category Reference

Properties

- [OALAction](#) * **action**
- EaseFunctionPtr **easeFunction**

The documentation for this category was generated from the following file:

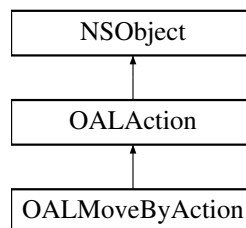
- OALAction.m

4.30 OALMoveByAction Class Reference

Moves the target from its current position by the specified delta over time in 3D space.

```
#import <OALAudioActions.h>
```

Inheritance diagram for OALMoveByAction:



Instance Methods

- (id) - [initWithDuration:delta:](#)
Initialize an action.
- (id) - [initWithUnitsPerSecond:delta:](#)
Initialize an action.

Class Methods

- (id) + [actionWithDuration:delta:](#)
Create a new action.
- (id) + [actionWithUnitsPerSecond:delta:](#)
Create a new action.

Protected Attributes

- [ALPoint](#) **startPoint**
The point this move is starting at.

Properties

- [ALPoint](#) **delta**
The amount to move the target by.
- float [unitsPerSecond](#)
The speed at which to move the target.

4.30.1 Detailed Description

Moves the target from its current position by the specified delta over time in 3D space.

4.30.2 Method Documentation

4.30.2.1 + (id) actionWithDuration: (float) *duration* delta:(ALPoint) *delta*

Create a new action.

Parameters

<i>duration</i>	The duration of the move.
<i>delta</i>	The amount to move by.

Returns

A new action.

4.30.2.2 + (id) actionWithUnitsPerSecond: (float) *unitsPerSecond* delta:(ALPoint) *delta*

Create a new action.

Parameters

<i>unitsPerSecond</i>	The rate of movement.
<i>delta</i>	The amount to move by.

Returns

A new action.

4.30.2.3 - (id) initWithDuration: (float) *duration* delta:(ALPoint) *delta*

Initialize an action.

Parameters

<i>duration</i>	The duration of the move.
<i>delta</i>	The amount to move by.

Returns

The initialized action.

4.30.2.4 - (id) initWithUnitsPerSecond: (float) *unitsPerSecond* delta:(ALPoint) *delta*

Initialize an action.

Parameters

<i>unitsPerSecond</i>	The rate of movement.
<i>delta</i>	The amount to move by.

Returns

The initialized action.

4.30.3 Member Data Documentation**4.30.3.1 - (ALPoint) startPoint** [protected]

The point this move is starting at.

4.30.4 Property Documentation**4.30.4.1 - (ALPoint) delta** [read],[write],[nonatomic],[assign]

The amount to move the target by.

4.30.4.2 - (float) unitsPerSecond [read],[write],[nonatomic],[assign]

The speed at which to move the target.

If this is 0, the target will be moved at the speed determined by duration.

The documentation for this class was generated from the following files:

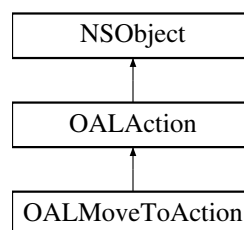
- OALAudioActions.h
- OALAudioActions.m

4.31 OALMoveToAction Class Reference

Moves the target from its current position to the specified position over time in 3D space.

```
#import <OALAudioActions.h>
```

Inheritance diagram for OALMoveToAction:

**Instance Methods**

- (id) - [initWithDuration:position:](#)
Initialize an action.
- (id) - [initWithUnitsPerSecond:position:](#)
Initialize an action.

Class Methods

- (id) + [actionWithDuration:position:](#)

Create a new action.

- (id) + [actionWithDuration:position:](#)

Create a new action.

Protected Attributes

- [ALPoint startPoint](#)

The point this move is starting at.

- [ALPoint delta](#)

The distance being moved.

Properties

- [ALPoint position](#)

The position to move the target to.

- float [unitsPerSecond](#)

The speed at which to move the target.

4.31.1 Detailed Description

Moves the target from its current position to the specified position over time in 3D space.

4.31.2 Method Documentation

4.31.2.1 + (id) [actionWithDuration: \(float\) duration position:\(ALPoint\) position](#)

Create a new action.

Parameters

<i>duration</i>	The duration of the move.
<i>position</i>	The position to move to.

Returns

A new action.

4.31.2.2 + (id) [actionWithUnitsPerSecond: \(float\) unitsPerSecond position:\(ALPoint\) position](#)

Create a new action.

Parameters

<i>unitsPerSecond</i>	The rate of movement.
<i>position</i>	The position to move to.

Returns

A new action.

4.31.2.3 - (id) initWithDuration: (float) *duration* position:(ALPoint) *position*

Initialize an action.

Parameters

<i>duration</i>	The duration of the move.
<i>position</i>	The position to move to.

Returns

The initialized action.

4.31.2.4 - (id) initWithUnitsPerSecond: (float) *unitsPerSecond* position:(ALPoint) *position*

Initialize an action.

Parameters

<i>unitsPerSecond</i>	The rate of movement.
<i>position</i>	The position to move to.

Returns

The initialized action.

4.31.3 Member Data Documentation

4.31.3.1 - (ALPoint) *delta* [protected]

The distance being moved.

4.31.3.2 - (ALPoint) *startPoint* [protected]

The point this move is starting at.

4.31.4 Property Documentation

4.31.4.1 - (ALPoint) *position* [read],[write],[nonatomic],[assign]

The position to move the target to.

4.31.4.2 - (float) *unitsPerSecond* [read],[write],[nonatomic],[assign]

The speed at which to move the target.

If this is 0, the target will be moved at the speed determined by duration.

The documentation for this class was generated from the following files:

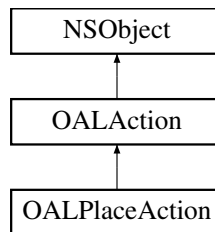
- OALAudioActions.h
- OALAudioActions.m

4.32 OALPlaceAction Class Reference

Places the target at the specified position.

```
#import <OALAudioActions.h>
```

Inheritance diagram for OALPlaceAction:



Instance Methods

- (id) - [initWithPosition:](#)
Initialize an action with the specified position.

Class Methods

- (id) + [actionWithPosition:](#)
Create an action with the specified position.

Properties

- [ALPoint position](#)
The position where the target will be placed.

Additional Inherited Members

4.32.1 Detailed Description

Places the target at the specified position.

4.32.2 Method Documentation

4.32.2.1 + (id) actionWithPosition: (ALPoint) position

Create an action with the specified position.

Parameters

<i>position</i>	The position to place the target at.
-----------------	--------------------------------------

Returns

A new action.

4.32.2.2 - (id) initWithPosition: (ALPoint) position

Initialize an action with the specified position.

Parameters

<i>position</i>	The position to place the target at.
-----------------	--------------------------------------

Returns

The initialized action.

4.32.3 Property Documentation

4.32.3.1 - (ALPoint) position [read],[write],[nonatomic],[assign]

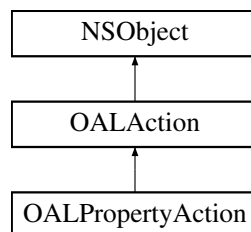
The position where the target will be placed.

The documentation for this class was generated from the following files:

- OALAudioActions.h
- OALAudioActions.m

4.33 OALPropertyAction Class Reference

Inheritance diagram for OALPropertyAction:



Instance Methods

- (id) - initWithDuration:propertyKey:endValue:
Initialize an action using the default function.
- (id) - initWithDuration:propertyKey:startValue:endValue:
Initialize an action.

Class Methods

- (id) + actionWithDuration:propertyKey:endValue:
Create a new action using the default function.
- (id) + actionWithDuration:propertyKey:startValue:endValue:
Create a new action.
- (OALPropertyAction *) + pitchActionWithDuration:endValue:
- (OALPropertyAction *) + pitchActionWithDuration:startValue:endValue:
- (OALPropertyAction *) + panActionWithDuration:endValue:
- (OALPropertyAction *) + panActionWithDuration:startValue:endValue:
- (OALPropertyAction *) + gainActionWithDuration:endValue:
- (OALPropertyAction *) + gainActionWithDuration:startValue:endValue:

Properties

- float [startValue](#)
The value that the property in the target will hold at the start of the action.
- float [endValue](#)
The value that the property in the target will hold at the end of the action.

Additional Inherited Members

4.33.1 Method Documentation

4.33.1.1 + (id) initWithDuration: (float) duration propertyKey:(NSString*) propertyKey endValue:(float) endValue

Create a new action using the default function.

The start value will be the current value of the target this action is applied to.

Parameters

<i>duration</i>	The duration of this action in seconds.
<i>propertyKey</i>	The property to modify.
<i>endValue</i>	The "ending" value that this action will converge upon when setting the target's property.

Returns

A new action.

4.33.1.2 + (id) initWithDuration: (float) duration propertyKey:(NSString*) propertyKey startValue:(float) startValue endValue:(float) endValue

Create a new action.

Parameters

<i>duration</i>	The duration of this action in seconds.
<i>propertyKey</i>	The property to modify.
<i>startValue</i>	The "starting" value that this action will diverge from when setting the target's property. If NAN, use the current value from the target.
<i>endValue</i>	The "ending" value that this action will converge upon when setting the target's property.

Returns

A new action.

4.33.1.3 - (id) initWithDuration: (float) duration propertyKey:(NSString*) propertyKey endValue:(float) endValue

Initialize an action using the default function.

The start value will be the current value of the target this action is applied to.

Parameters

<i>duration</i>	The duration of this action in seconds.
<i>propertyKey</i>	The property to modify.
<i>endValue</i>	The "ending" value that this action will converge upon when setting the target's property.

Returns

The initialized action.

4.33.1.4 - (id) initWithDuration: (float) *duration* propertyKey:(NSString*) *propertyKey* startValue:(float) *startValue* endValue:(float) *endValue*

Initialize an action.

Parameters

<i>duration</i>	The duration of this action in seconds.
<i>propertyKey</i>	The property to modify.
<i>startValue</i>	The "starting" value that this action will diverge from when setting the target's property. If NAN, use the current value from the target.
<i>endValue</i>	The "ending" value that this action will converge upon when setting the target's property.

Returns

The initialized action.

4.33.2 Property Documentation

4.33.2.1 - (float) *endValue* [read], [write], [nonatomic], [assign]

The value that the property in the target will hold at the end of the action.

4.33.2.2 - (float) *startValue* [read], [write], [nonatomic], [assign]

The value that the property in the target will hold at the start of the action.

The documentation for this class was generated from the following files:

- OALAction.h
- OALAction.m

4.34 OALPropertyAction() Category Reference**Properties**

- float **delta**
- NSString * **propertyKey**

The documentation for this category was generated from the following file:

- OALAction.m

4.35 OALPropertyAction(Audio) Category Reference**Class Methods**

- (OALPropertyAction *) + **pitchActionWithDuration:endValue:**

- (OALPropertyAction *) + **pitchActionWithDuration:startValue:endValue:**
- (OALPropertyAction *) + **panActionWithDuration:endValue:**
- (OALPropertyAction *) + **panActionWithDuration:startValue:endValue:**
- (OALPropertyAction *) + **gainActionWithDuration:endValue:**
- (OALPropertyAction *) + **gainActionWithDuration:startValue:endValue:**

The documentation for this category was generated from the following files:

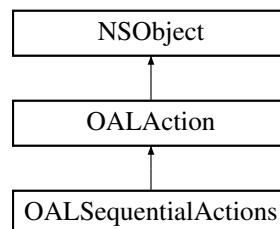
- OALAudioActions.h
- OALAudioActions.m

4.36 OALSequentialActions Class Reference

A set of actions that get run in sequence.

```
#import <OALUtilityActions.h>
```

Inheritance diagram for OALSequentialActions:



Instance Methods

- (id) - **initWithActions:**
Initialize an action.

Class Methods

- (id) + **actions:**
Create an action.
- (id) + **actionsFromArray:**
Create an action.

Protected Attributes

- NSInteger **actionIndex_**
The index of the action currently being processed.
- float **pLastComplete_**
The last completeness proportion value acted upon.
- float **pCurrentActionDuration_**
The proportional duration of the current action.
- float **pCurrentActionComplete_**
The proportional completeness of the current action.

Properties

- NSMutableArray * [actions](#)

The actions which will be run.

4.36.1 Detailed Description

A set of actions that get run in sequence.

4.36.2 Method Documentation

4.36.2.1 + (id) actions: (OALAction*) actions, NS_REQUIRES_NIL_TERMINATION

Create an action.

Parameters

<i>actions</i>	The comma separated list of actions.
<i>NS_REQUIRES_NIL_TERMINATION</i>	List of actions must be terminated by a nil.

Returns

A new set of sequential actions.

4.36.2.2 + (id) actionsFromArray: (NSArray*) actions

Create an action.

Parameters

<i>actions</i>	The actions to run.
----------------	---------------------

Returns

A new set of sequential actions.

4.36.2.3 - (id) initWithActions: (NSArray*) actions

Initialize an action.

Parameters

<i>actions</i>	The actions to run.
----------------	---------------------

Returns

The initialized set of sequential actions.

4.36.3 Member Data Documentation

4.36.3.1 - (NSInteger) `actionIndex_` [protected]

The index of the action currently being processed.

4.36.3.2 - (float) `pCurrentActionComplete_` [protected]

The proportional completeness of the current action.

4.36.3.3 - (float) `pCurrentActionDuration_` [protected]

The proportional duration of the current action.

4.36.3.4 - (float) `pLastComplete_` [protected]

The last completeness proportion value acted upon.

4.36.4 Property Documentation

4.36.4.1 - (NSMutableArray*) `actions` [read],[write],[nonatomic],[retain]

The actions which will be run.

The documentation for this class was generated from the following files:

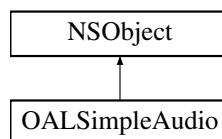
- OALUtilityActions.h
- OALUtilityActions.m

4.37 OALSimpleAudio Class Reference

A simpler interface to the ObjectAL sound library.

```
#import <OALSimpleAudio.h>
```

Inheritance diagram for OALSimpleAudio:



Instance Methods

- (bool) - [preloadBg:](#)
Preload background music.
- (bool) - [preloadBg:seekTime:](#)
Preload background music.
- (bool) - [playBg](#)
Play whatever background music is preloaded.
- (bool) - [playBgWithLoop:](#)
Play whatever background music is preloaded.
- (bool) - [playBg:](#)

- Play the background music at the specified path.*

 - (bool) - [playBg:loop:](#)

Play the background music at the specified path.
- (bool) - [playBg:volume:pan:loop:](#)

Play the background music at the specified path.
- (void) - [stopBg](#)

Stop the background music playback and rewind.
- (ALBuffer *) - [preloadEffect:](#)

Preload and cache a sound effect for later playback.
- (ALBuffer *) - [preloadEffect:reduceToMono:](#)

Preload and cache a sound effect for later playback.
- (bool) - [unloadEffect:](#)

Unload a preloaded effect.
- (void) - [unloadAllEffects](#)

Unload all preloaded effects that are not currently being played (paused or not).
- (id< [ALSoundSource](#) >) - [playEffect:](#)

Play a sound effect with volume 1.0, pitch 1.0, pan 0.0, loop NO.
- (id< [ALSoundSource](#) >) - [playEffect:loop:](#)

Play a sound effect with volume 1.0, pitch 1.0, pan 0.0.
- (id< [ALSoundSource](#) >) - [playEffect:volume:pitch:pan:loop:](#)

Play a sound effect.
- (id< [ALSoundSource](#) >) - [playBuffer:volume:pitch:pan:loop:](#)

Play a sound effect from a user-supplied buffer.
- (void) - [stopAllEffects](#)

Stop ALL sound effect playback.
- (void) - [stopEverything](#)

Stop all effects and bg music.
- (void) - [resetToDefault](#)

Reset everything in this object to its default state.

Class Methods

- (OALSimpleAudio *) + [sharedInstanceWithSources:](#)
- Start [OALSimpleAudio](#) with the specified number of reserved sources.*
- (OALSimpleAudio *) + [sharedInstanceWithReservedSources:monoSources:stereoSources:](#)
- Start [OALSimpleAudio](#) with the specified parameters.*

Protected Member Functions

- () - [SYNTHESIZE_SINGLETON_FOR_CLASS_HEADER](#)
- Singleton implementation providing "sharedInstance" and "purgeSharedInstance" methods.*

Protected Attributes

- NSMutableDictionary * [preloadCache](#)
- Cache for preloaded sound samples.*
- uint [pendingLoadCount](#)
- keeping track of how many effects remain to be loaded*

Properties

- bool [allowIpod](#)
If YES, allow ipod music to continue playing (NOT SUPPORTED ON THE SIMULATOR).
- bool [useHardwareIfAvailable](#)
Determines what to do if no other application is playing audio and allowIpod = YES (NOT SUPPORTED ON THE SIMULATOR).
- bool [honorSilentSwitch](#)
If true, mute when backgrounded, screen locked, or the ringer switch is turned off (NOT SUPPORTED ON THE SIMULATOR).
- int [reservedSources](#)
The number of sources [OALSimpleAudio](#) is using (max 32 on current iOS devices).
- [ALDevice](#) * [device](#)
The device we are using.
- [ALContext](#) * [context](#)
The context we are using.
- [ALChannelSource](#) * [channel](#)
The sound channel used by this object.
- NSURL * [backgroundTrackURL](#)
Background audio URL.
- [OALAudioTrack](#) * [backgroundTrack](#)
Audio track to play background music.
- bool [bgPaused](#)
Pauses BG music playback.
- bool [bgMuted](#)
Mutes BG music playback.
- bool [bgPlaying](#)
If true, BG music is currently playing.
- float [bgVolume](#)
Background music playback gain/volume (0.0 - 1.0)
- bool [effectsPaused](#)
Pauses effects playback.
- bool [effectsMuted](#)
Mutes effects playback.
- float [effectsVolume](#)
Master effects gain/volume (0.0 - 1.0)
- bool [paused](#)
Pauses everything.
- bool [muted](#)
Mutes all audio.
- bool [preloadCacheEnabled](#)
Enables/disables the preload cache.
- NSInteger [preloadCacheCount](#)
The number of items currently in the preload cache.
- bool [manuallySuspended](#)
Set to YES to manually suspend the sound system.
- bool [interrupted](#)
If YES, the sound system is interrupted.
- bool [suspended](#)
If YES, the sound system is suspended.

4.37.1 Detailed Description

A simpler interface to the ObjectAL sound library.

This singleton can be used alone for simpler audio needs, or in conjunction with user-created audio objects for more advanced needs (as is done in many of the demos).

For sound effects, it initializes OpenAL with the default [ALDevice](#), an [ALContext](#), and an [ALChannelSource](#) consisting of all 32 interruptible [ALSource](#) objects (the maximum currently allowed for iOS). If you want to create your own sources as well, change the reservedSources property.

For background audio, it creates a single [OALAudioTrack](#), which will not reserve resources unless used. (you can create more [OALAudioTrack](#) objects for your own use if you want).

This singleton also provides access to the more common configuration options available in OALAudioSupport.

All audio playback commands are delegated either to the [ALChannelSource](#) (for sound effects), or to the [OALAudioTrack](#) (for BG music).

4.37.2 Method Documentation

4.37.2.1 - (bool) playBg

Play whatever background music is preloaded.

Returns

TRUE if the operation was successful.

4.37.2.2 - (bool) playBg: (NSString*) path

Play the background music at the specified path.

If the music has not been preloaded, this method will load the music and then play, incurring a slight delay.

Note: only **ONE** background music file may be played or preloaded at a time via [OALSimpleAudio](#). If you play or preload another file, the one currently playing will stop.

Parameters

<i>path</i>	The path containing the background music.
-------------	---

Returns

TRUE if the operation was successful.

4.37.2.3 - (bool) playBg: (NSString*) path loop:(bool) loop

Play the background music at the specified path.

If the music has not been preloaded, this method will load the music and then play, incurring a slight delay.

Note: only **ONE** background music file may be played or preloaded at a time via [OALSimpleAudio](#). If you play or preload another file, the one currently playing will stop.

Parameters

<i>path</i>	The path containing the background music.
<i>loop</i>	If true, loop the bg track.

Returns

TRUE if the operation was successful.

4.37.2.4 - (bool) playBg: (NSString*) filePath volume:(float) volume pan:(float) pan loop:(bool) loop

Play the background music at the specified path.

If the music has not been preloaded, this method will load the music and then play, incurring a slight delay.

Note: only **ONE** background music file may be played or preloaded at a time via [OALSimpleAudio](#). If you play or preload another file, the one currently playing will stop. To play multiple audio tracks, create an [OALAudioTrack](#).

Note: pan will have no effect when running on iOS versions prior to 4.0.

Parameters

<i>filePath</i>	The path containing the sound data.
<i>volume</i>	The volume (gain) to play at (0.0 - 1.0).
<i>pan</i>	Left-right panning (-1.0 = far left, 1.0 = far right) (Only on iOS 4.0+).
<i>loop</i>	If TRUE, the sound will loop until you call "stopBg".

Returns

TRUE if the operation was successful.

4.37.2.5 - (bool) playBgWithLoop: (bool) loop

Play whatever background music is preloaded.

Parameters

<i>loop</i>	If true, loop the bg track.
-------------	-----------------------------

Returns

TRUE if the operation was successful.

4.37.2.6 - (id< ALSoundSource >) playBuffer: (ALBuffer*) buffer volume:(float) volume pitch:(float) pitch pan:(float) pan loop:(bool) loop

Play a sound effect from a user-supplied buffer.

Parameters

<i>buffer</i>	The buffer containing the sound data.
<i>volume</i>	The volume (gain) to play at (0.0 - 1.0).
<i>pitch</i>	The pitch to play at (1.0 = normal pitch).
<i>pan</i>	Left-right panning (-1.0 = far left, 1.0 = far right).
<i>loop</i>	If TRUE, the sound will loop until you call "stop" on the returned sound source.

Returns

The sound source being used for playback, or nil if an error occurred (You'll need to keep this if you want to be able to stop a looped playback).

4.37.2.7 - (id< **ALSoundSource** >) playEffect: (NSString*) *filePath*

Play a sound effect with volume 1.0, pitch 1.0, pan 0.0, loop NO.

The sound will be loaded and cached if it wasn't already.

Parameters

<i>filePath</i>	The path containing the sound data.
-----------------	-------------------------------------

Returns

The sound source being used for playback, or nil if an error occurred.

4.37.2.8 - (id< **ALSoundSource** >) playEffect: (NSString*) *filePath* loop:(bool) *loop*

Play a sound effect with volume 1.0, pitch 1.0, pan 0.0.

The sound will be loaded and cached if it wasn't already.

Parameters

<i>filePath</i>	The path containing the sound data.
<i>loop</i>	If TRUE, the sound will loop until you call "stop" on the returned sound source.

Returns

The sound source being used for playback, or nil if an error occurred.

4.37.2.9 - (id< **ALSoundSource** >) playEffect: (NSString*) *filePath* volume:(float) *volume* pitch:(float) *pitch* pan:(float) *pan* loop:(bool) *loop*

Play a sound effect.

The sound will be loaded and cached if it wasn't already.

Parameters

<i>filePath</i>	The path containing the sound data.
<i>volume</i>	The volume (gain) to play at (0.0 - 1.0).
<i>pitch</i>	The pitch to play at (1.0 = normal pitch).
<i>pan</i>	Left-right panning (-1.0 = far left, 1.0 = far right).
<i>loop</i>	If TRUE, the sound will loop until you call "stop" on the returned sound source.

Returns

The sound source being used for playback, or nil if an error occurred (You'll need to keep this if you want to be able to stop a looped playback).

4.37.2.10 - (bool) preloadBg: (NSString*) *path*

Preload background music.

Note: only **ONE** background music file may be played or preloaded at a time via [OALSimpleAudio](#). If you play or preload another file, the one currently playing will stop.

Parameters

<i>path</i>	The path containing the background music.
-------------	---

Returns

TRUE if the operation was successful.

4.37.2.11 - (bool) preloadBg: (NSString*) path seekTime:(NSTimeInterval) seekTime

Preload background music.

Note: only **ONE** background music file may be played or preloaded at a time via [OALSimpleAudio](#). If you play or preload another file, the one currently playing will stop.

Parameters

<i>path</i>	The path containing the background music.
<i>seekTime</i>	the position in the file to start playing at.

Returns

TRUE if the operation was successful.

4.37.2.12 - (ALBuffer *) preloadEffect: (NSString*) filePath

Preload and cache a sound effect for later playback.

Parameters

<i>filePath</i>	The path containing the sound data.
-----------------	-------------------------------------

4.37.2.13 - (ALBuffer *) preloadEffect: (NSString*) filePath reduceToMono:(bool) reduceToMono

Preload and cache a sound effect for later playback.

Parameters

<i>filePath</i>	The path containing the sound data.
<i>reduceToMono</i>	If true, reduce the sample to mono (stereo samples don't support panning or positional audio).

4.37.2.14 - (void) resetToDefault

Reset everything in this object to its default state.

4.37.2.15 + (OALSimpleAudio *) sharedInstanceWithReservedSources: (int) reservedSources monoSources:(int) monoSources stereoSources:(int) stereoSources

Start [OALSimpleAudio](#) with the specified parameters.

With this initializer, you can set the total number of mono and stereo sources available, as well as how many sources are to be reserved by [OALSimpleAudio](#).

The number of mono and stereo sources represents the GLOBAL number of sources available for EVERYONE, not

just [OALSimpleAudio](#). Their combined values must not exceed 32 (the max allowed sources in iOS).

reservedSources is independent of this; it represents how many of the above mentioned sources to reserve for [OALSimpleAudio](#)'s use.

Note: This method must be called ONLY ONCE, *BEFORE* any attempt is made to access the shared instance.

Parameters

<i>reservedSources</i>	The number of sources to reserve for OALSimpleAudio 's use when initializing. iOS currently supports up to 32 sources total.
<i>monoSources</i>	The GLOBAL number of sources supporting mono (default 28).
<i>stereoSources</i>	The GLOBAL number of sources supporting stereo (default 4).

Returns

The shared instance.

4.37.2.16 + (OALSimpleAudio *) sharedInstanceWithSources: (int) sources

Start [OALSimpleAudio](#) with the specified number of reserved sources.

Call this initializer if you want to use [OALSimpleAudio](#), but keep some of the device's audio sources (there are 32 in total) for your own use.

Note: This method must be called ONLY ONCE, *BEFORE* any attempt is made to access the shared instance. To change the reserved sources after instantiation, modify reservedSources.

Parameters

<i>sources</i>	the number of sources OALSimpleAudio will reserve for itself.
----------------	---

Returns

The shared instance.

4.37.2.17 - (void) stopAllEffects

Stop ALL sound effect playback.

4.37.2.18 - (void) stopBg

Stop the background music playback and rewind.

4.37.2.19 - (void) stopEverything

Stop all effects and bg music.

4.37.2.20 - OALSimpleAudio: (OALSimpleAudio)

Singleton implementation providing "sharedInstance" and "purgeSharedInstance" methods.

- (OALSimpleAudio*) sharedInstance: Get the shared singleton instance.

- (void) purgeSharedInstance: Purge (deallocate) the shared instance.

4.37.2.21 - (void) unloadAllEffects

Unload all preloaded effects that are not currently being played (paused or not).

Turning on debug logging will show which effects were not unloaded. It is useful to put a call to this method in "applicationDidReceiveMemoryWarning" in your app delegate.

4.37.2.22 - (bool) unloadEffect: (NSString*) filePath

Unload a preloaded effect.

Only unloads if no source is currently playing that effect (or paused with the effect loaded).

Parameters

<i>filePath</i>	The path containing the sound data that was previously loaded.
-----------------	--

Returns

YES if the effect was unloaded. Turn on debug logging to see why an effect was not unloaded.

4.37.3 Member Data Documentation**4.37.3.1 - (uint) pendingLoadCount** [protected]

keeping track of how many effects remain to be loaded

4.37.3.2 - (NSMutableDictionary*) preloadCache [protected]

Cache for preloaded sound samples.

4.37.4 Property Documentation**4.37.4.1 - (bool) allowIpod** [read], [write], [nonatomic], [assign]

If YES, allow ipod music to continue playing (NOT SUPPORTED ON THE SIMULATOR).

Note: If this is enabled, and another app is playing music, background audio playback will use the SOFTWARE codecs, NOT hardware.

If allowIpod = NO, the application will ALWAYS use hardware decoding.

iOS Only.

See Also

[useHardwareIfAvailable](#)

Default value: YES

4.37.4.2 - (OALAudioTrack*) backgroundTrack [read], [nonatomic], [retain]

Audio track to play background music.

Background audio track.

4.37.4.3 - (NSURL *) **backgroundTrackURL** [read], [nonatomic], [retain]

Background audio URL.

4.37.4.4 - (bool) **bgMuted** [read], [write], [nonatomic], [assign]

Mutes BG music playback.

4.37.4.5 - (bool) **bgPaused** [read], [write], [nonatomic], [assign]

Pauses BG music playback.

4.37.4.6 - (bool) **bgPlaying** [read], [nonatomic], [assign]

If true, BG music is currently playing.

4.37.4.7 - (float) **bgVolume** [read], [write], [nonatomic], [assign]

Background music playback gain/volume (0.0 - 1.0)

4.37.4.8 - (ALChannelSource *) **channel** [read], [nonatomic], [retain]

The sound channel used by this object.

The channel source used by [OALSimpleAudio](#).

Only mess with this if you know what you are doing!

4.37.4.9 - (ALContext *) **context** [read], [nonatomic], [retain]

The context we are using.

4.37.4.10 - (ALDevice *) **device** [read], [nonatomic], [retain]

The device we are using.

4.37.4.11 - (bool) **effectsMuted** [read], [write], [nonatomic], [assign]

Mutes effects playback.

4.37.4.12 - (bool) **effectsPaused** [read], [write], [nonatomic], [assign]

Pauses effects playback.

4.37.4.13 - (float) **effectsVolume** [read], [write], [nonatomic], [assign]

Master effects gain/volume (0.0 - 1.0)

4.37.4.14 - (bool) **honorSilentSwitch** [read], [write], [nonatomic], [assign]

If true, mute when backgrounded, screen locked, or the ringer switch is turned off (NOT SUPPORTED ON THE SIMULATOR).

iOS Only.

Default value: YES

4.37.4.15 - (bool) **interrupted** [read], [nonatomic], [assign]

If YES, the sound system is interrupted.

iOS Only.

4.37.4.16 - (bool) **manuallySuspended** [read], [write], [nonatomic], [assign]

Set to YES to manually suspend the sound system.

4.37.4.17 - (bool) **muted** [read], [write], [nonatomic], [assign]

Mutes all audio.

4.37.4.18 - (bool) **paused** [read], [write], [nonatomic], [assign]

Pauses everything.

4.37.4.19 - (NSInteger) **preloadCacheCount** [read], [nonatomic], [assign]

The number of items currently in the preload cache.

4.37.4.20 - (bool) **preloadCacheEnabled** [read], [write], [nonatomic], [assign]

Enables/disables the preload cache.

If the preload cache is disabled, effects preloading will do nothing (BG preloading will still work).

4.37.4.21 - (int) **reservedSources** [read], [write], [nonatomic], [assign]

The number of sources [OALSimpleAudio](#) is using (max 32 on current iOS devices).

4.37.4.22 - (bool) **suspended** [read], [nonatomic], [assign]

If YES, the sound system is suspended.

4.37.4.23 - (bool) **useHardwareIfAvailable** [read], [write], [nonatomic], [assign]

Determines what to do if no other application is playing audio and allowIpod = YES (NOT SUPPORTED ON THE SIMULATOR).

If NO, the application will ALWAYS use software decoding. The advantage to this is that the user can background your application and then start audio playing from another application. If useHardwareIfAvailable = YES, the user won't be able to do this.

If this is set to YES, the application will use hardware decoding if no other application is currently playing audio. However, no other application will be able to start playing audio if it wasn't playing already.

Note: This switch has no effect if allowIpod = NO.

iOS Only.

See Also

[allowIpod](#)

Default value: YES

The documentation for this class was generated from the following files:

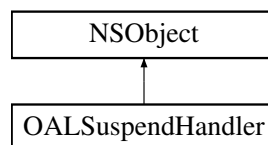
- OALSimpleAudio.h
- OALSimpleAudio.m

4.38 OALSuspendHandler Class Reference

Provides two controls (interrupted and manuallySuspended) for suspending a slave object, and also propagates such control messages to interested listeners.

```
#import <OALSuspendHandler.h>
```

Inheritance diagram for OALSuspendHandler:



Instance Methods

- (id) - [initWithTarget:selector:](#)
Initialize a handler with the specified slave target and selector.
- (void) - [addSuspendListener:](#)
Add a listener that will receive manual suspend and interrupt events.
- (void) - [removeSuspendListener:](#)
Remove a registered listener.

Class Methods

- (OALSuspendHandler *) + [handlerWithTarget:selector:](#)
Create a new handler with the specified slave target and selector.

Protected Attributes

- NSMutableArray * [listeners](#)
Listeners that will receive manualSuspend and interrupt events.
- NSMutableArray * [manualSuspendStates](#)
Holder for the state of manualSuspend in listeners when this object is manually suspended.
- SEL [suspendStatusChangeSelector](#)

Selector to be invoked on suspend or unsuspend.

- bool [manualSuspendLock](#)

Holds the current "manually suspended" state.

- bool [interruptLock](#)

Holds the current "interrupted" state.

Properties

- bool [manuallySuspended](#)

If YES, the manual suspend control is set.

- bool [interrupted](#)

If YES, the interrupt control is set.

- bool [suspended](#)

If YES, the slave object is suspended.

4.38.1 Detailed Description

Provides two controls (interrupted and manuallySuspended) for suspending a slave object, and also propagates such control messages to interested listeners.

"interrupted" is meant to be set by the system when an interrupt occurs.

"manuallySuspended" is a user-settable control for suspending an object.

"manuallySuspended" also has an extra step in its processing: When set, the handler makes a note of what its listeners' "manuallySuspended" values are. When cleared, it will only clear a listener's "manuallySuspended" value if it was not set at suspend time. This allows for ad-hoc setting/clearing of "manuallySuspended" in the middle of a handler/listener graph rather than only from the top level.

When either control is set, the slave object will be suspended. When both are cleared, the slave object will be unsuspended.

4.38.2 Method Documentation

4.38.2.1 - (void) addSuspendListener: (id<OALSuspendListener>) listener

Add a listener that will receive manual suspend and interrupt events.

Parameters

<i>listener</i>	The listener to register with this handler.
-----------------	---

4.38.2.2 + (OALSuspendHandler *) handlerWithTarget: (id) target selector:(SEL) selector

Create a new handler with the specified slave target and selector.

The selector provided must take a single boolean value like so:

- (void) setSuspended:(bool) value

Parameters

<i>target</i>	The slave object that will receive suspend/unsuspend events.
<i>selector</i>	The selector for a "set suspended" method, taking a single boolean parameter.

4.38.2.3 - (id) initWithTarget: (id) *target* selector:(SEL) *selector*

Initialize a handler with the specified slave target and selector.

The selector provided must take a single boolean value like so:

- (void) setSuspended:(bool) value

Parameters

<i>target</i>	The slave object that will receive suspend/unsuspend events.
<i>selector</i>	The selector for a "set suspended" method, taking a single boolean parameter.

4.38.2.4 - (void) removeSuspendListener: (id<OALSuspendListener>) *listener*

Remove a registered listener.

Parameters

<i>listener</i>	The listener to unregister from this handler.
-----------------	---

4.38.3 Member Data Documentation

4.38.3.1 - (bool) interruptLock [protected]

Holds the current "interrupted" state.

4.38.3.2 - (NSMutableArray*) listeners [protected]

Listeners that will receive manualSuspend and interrupt events.

4.38.3.3 - (bool) manualSuspendLock [protected]

Holds the current "manually suspended" state.

4.38.3.4 - (NSMutableArray*) manualSuspendStates [protected]

Holder for the state of manualSuspend in listeners when this object is manually suspended.

4.38.3.5 - (SEL) suspendStatusChangeSelector [protected]

Selector to be invoked on suspend or unsuspend.

Takes the signature: setSelected:(bool) value

4.38.4 Property Documentation

4.38.4.1 - (bool) interrupted [read], [write], [nonatomic], [assign]

If YES, the interrupt control is set.

4.38.4.2 - (bool) `manuallySuspended` `[read],[write],[nonatomic],[assign]`

If YES, the manual suspend control is set.

4.38.4.3 - (bool) `suspended` `[read],[nonatomic],[assign]`

If YES, the slave object is suspended.

The documentation for this class was generated from the following files:

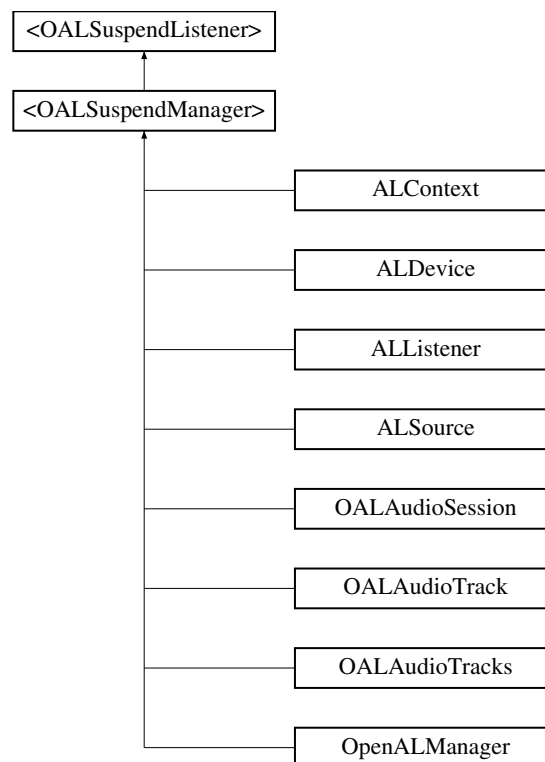
- `OALSuspendHandler.h`
- `OALSuspendHandler.m`

4.39 <OALSuspendListener> Protocol Reference

Allows an object to participate in interrupt and suspend operations.

`#import <OALSuspendHandler.h>`

Inheritance diagram for <OALSuspendListener>:



Properties

- bool `manuallySuspended`
Set to YES to manually suspend.
- bool `interrupted`
If YES, this object is interrupted.

4.39.1 Detailed Description

Allows an object to participate in interrupt and suspend operations.

Objects may hook into [OALAudioSession](#)'s interrupt and suspend model by calling `[[OALAudioSession sharedInstance] addSuspendListener:self]`.

Note: You must NOT set the "interrupted" property manually. It is designed to be set automatically by system interrupts.

See Also

[OALAudioSession](#)

4.39.2 Property Documentation

4.39.2.1 `-(bool) interrupted` `[read], [write], [nonatomic], [assign]`

If YES, this object is interrupted.

Note: This property must NOT be set by the user!

4.39.2.2 `-(bool) manuallySuspended` `[read], [write], [nonatomic], [assign]`

Set to YES to manually suspend.

The documentation for this protocol was generated from the following file:

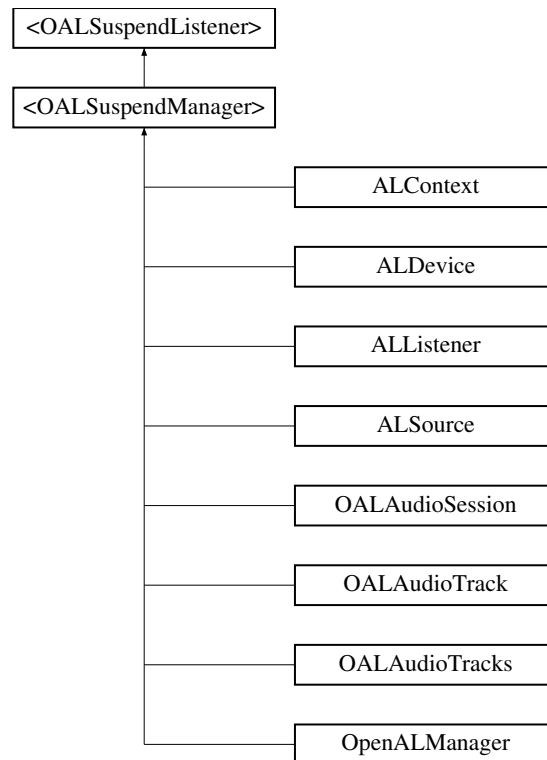
- `OALSuspendHandler.h`

4.40 <OALSuspendManager> Protocol Reference

A suspend manager is a listener that also allows other objects to subscribe to receive events as the manager receives them.

```
#import <OALSuspendHandler.h>
```

Inheritance diagram for <OALSuspendManager>:



Instance Methods

- (void) - [addSuspendListener:](#)
Add a listener that will receive manual suspend and interrupt events.
- (void) - [removeSuspendListener:](#)
Remove a registered listener.

Properties

- bool [suspended](#)
If YES, this object is suspended.

4.40.1 Detailed Description

A suspend manager is a listener that also allows other objects to subscribe to receive events as the manager receives them.

4.40.2 Method Documentation

4.40.2.1 - (void) [addSuspendListener:](#) (id< [OALSuspendListener](#) >) *listener*

Add a listener that will receive manual suspend and interrupt events.

Parameters

<i>listener</i>	The listener to register with this handler.
-----------------	---

4.40.2.2 - (void) removeSuspendListener: (id< OALSuspendListener >) listener

Remove a registered listener.

Parameters

<i>listener</i>	The listener to unregister from this handler.
-----------------	---

4.40.3 Property Documentation

4.40.3.1 - (bool) suspended [read],[nonatomic],[assign]

If YES, this object is suspended.

The documentation for this protocol was generated from the following file:

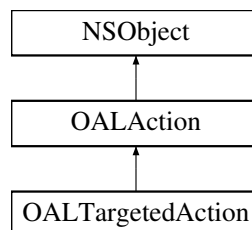
- OALSuspendHandler.h

4.41 OALTargetedAction Class Reference

Ignores whatever target it was invoked upon and applies the specified action on the target specified at creation time.

```
#import <OALUtilityActions.h>
```

Inheritance diagram for OALTargetedAction:



Instance Methods

- (id) - initWithTarget:action:
Initialize an action.

Class Methods

- (id) + actionWithTarget:action:
Create an action.

Protected Attributes

- OALAction * action_
The action that will be run on the target.

Properties

- id forcedTarget
The target which this action will actually be invoked upon.

4.41.1 Detailed Description

Ignores whatever target it was invoked upon and applies the specified action on the target specified at creation time.

4.41.2 Method Documentation

4.41.2.1 + (id) `actionWithTarget: (id) target action:(OALAction*) action`

Create an action.

Parameters

<i>target</i>	The target to run the action upon.
<i>action</i>	The action to run.

Returns

A new action.

4.41.2.2 - (id) `initWithTarget: (id) target action:(OALAction*) action`

Initialize an action.

Parameters

<i>target</i>	The target to run the action upon.
<i>action</i>	The action to run.

Returns

The initialized action.

4.41.3 Member Data Documentation

4.41.3.1 - (OALAction*) `action_` [protected]

The action that will be run on the target.

4.41.4 Property Documentation

4.41.4.1 - (id) `forcedTarget` [read], [write], [nonatomic], [assign]

The target which this action will actually be invoked upon.

The documentation for this class was generated from the following files:

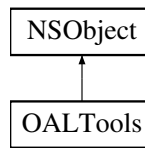
- OALUtilityActions.h
- OALUtilityActions.m

4.42 OALTools Class Reference

Miscellaneous tools used by ObjectAL.

```
#import <OALTools.h>
```

Inheritance diagram for OALTools:



Class Methods

- (void) + **setDefaultBundle:**
Set the default bundle to use when looking up paths.
- (NSBundle *) + **defaultBundle**
The default bundle used when looking up paths.
- (NSURL *) + **urlForPath:**
Returns the URL corresponding to the specified path.
- (NSURL *) + **urlForPath:bundle:**
Returns the URL corresponding to the specified path.
- (void) + **notifyExtAudioError:function:description:**
Notify an error if the specified ExtAudio error code indicates an error.
- (void) + **notifyAudioSessionError:function:description:**
Notify an error if the specified AudioSession error code indicates an error.

4.42.1 Detailed Description

Miscellaneous tools used by ObjectAL.

4.42.2 Method Documentation

4.42.2.1 + (NSBundle *) defaultBundle

The default bundle used when looking up paths.

return The default bundle.

4.42.2.2 + (void) notifyAudioSessionError: (OSStatus) errorCode function:(const char*) function description:(NSString*) description, ...

Notify an error if the specified AudioSession error code indicates an error.

This will log the error and also potentially post an audio error notification (OALAudioErrorNotification) if it is suspected that this error is a result of the audio session getting corrupted.

Parameters

<i>errorCode,:</i>	The error code returned from an OS call.
<i>function,:</i>	The function name where the error occurred.
<i>description,:</i>	A printf-style description of what happened.

4.42.2.3 **+(void) notifyExtAudioError: (OSStatus) *errorCode* function:(const char*) *function* description:(NSString*) *description*, ...**

Notify an error if the specified ExtAudio error code indicates an error.

This will log the error and also potentially post an audio error notification (OALAudioErrorNotification) if it is suspected that this error is a result of the audio session getting corrupted.

Parameters

<i>errorCode</i> ,:	The error code returned from an OS call.
<i>function</i> ,:	The function name where the error occurred.
<i>description</i> ,:	A printf-style description of what happened.

4.42.2.4 **+(void) setDefaultBundle: (NSBundle*) *bundle***

Set the default bundle to use when looking up paths.

Parameters

<i>bundle</i>	The new default bundle.
---------------	-------------------------

4.42.2.5 **+(NSURL *) urlForPath: (NSString*) *path***

Returns the URL corresponding to the specified path.

If the path is not absolute (starts with a "/"), this method will look for the file in the default bundle.

Parameters

<i>path</i>	The path to convert to a URL.
-------------	-------------------------------

Returns

The corresponding URL or nil if a URL could not be formed.

4.42.2.6 **+(NSURL *) urlForPath: (NSString*) *path* bundle:(NSBundle*) *bundle***

Returns the URL corresponding to the specified path.

If the path is not absolute (starts with a "/"), this method will look for the file in the specified bundle.

Parameters

<i>path</i>	The path to convert to a URL.
<i>bundle</i>	The bundle to look inside for relative paths.

Returns

The corresponding URL or nil if a URL could not be formed.

The documentation for this class was generated from the following files:

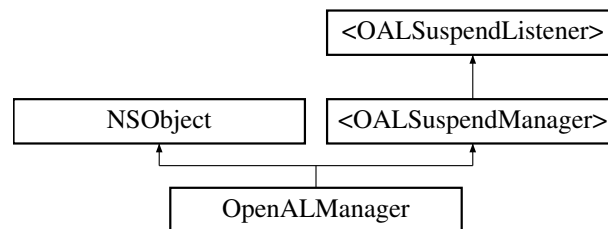
- OALTools.h
- OALTools.m

4.43 OpenALManager Class Reference

Manager class for OpenAL objects (ObjectAL).

```
#import <OpenALManager.h>
```

Inheritance diagram for OpenALManager:



Instance Methods

- (ALBuffer *) - [bufferFromFile:](#)
Load an OpenAL buffer with the contents of an audio file.
- (ALBuffer *) - [bufferFromFile:reduceToMono:](#)
Load an OpenAL buffer with the contents of an audio file.
- (ALBuffer *) - [bufferFromUrl:](#)
Load an OpenAL buffer with the contents of an audio file.
- (ALBuffer *) - [bufferFromUrl:reduceToMono:](#)
Load an OpenAL buffer with the contents of an audio file.
- (NSString *) - [bufferAsyncFromFile:target:selector:](#)
Load an OpenAL buffer with the contents of an audio file asynchronously.
- (NSString *) - [bufferAsyncFromFile:reduceToMono:target:selector:](#)
Load an OpenAL buffer with the contents of an audio file asynchronously.
- (NSString *) - [bufferAsyncFromUrl:target:selector:](#)
Load an OpenAL buffer with the contents of a URL asynchronously.
- (NSString *) - [bufferAsyncFromUrl:reduceToMono:target:selector:](#)
Load an OpenAL buffer with the contents of a URL asynchronously.
- (void) - [clearAllBuffers](#)
Clear all references to sound data from ALL buffers, managed or not.

Protected Member Functions

- () - [SYNTHESIZE_SINGLETON_FOR_CLASS_HEADER](#)
Singleton implementation providing "sharedInstance" and "purgeSharedInstance" methods.

Protected Attributes

- NSMutableArray * [devices](#)
All opened devices.
- OALSuspendHandler * [suspendHandler](#)
Handles suspending and interrupting for this object.
- NSOperationQueue * [operationQueue](#)
Operation queue for asynchronous loading.

Properties

- NSArray * [availableDevices](#)
List of available playback devices (NSString).*
- NSArray * [availableCaptureDevices](#)
List of available capture devices (NSString).*
- ALContext * [currentContext](#)
The current context (some context operations require the context to be the "current" one).
- NSString * [defaultCaptureDeviceSpecifier](#)
Name of the default capture device.
- NSString * [defaultDeviceSpecifier](#)
Name of the default playback device.
- NSArray * [devices](#)
List of all open devices (ALDevice).*
- ALdouble [mixerOutputFrequency](#)
The frequency of the output mixer.
- ALint [renderingQuality](#)
The rendering quality.

4.43.1 Detailed Description

Manager class for OpenAL objects (ObjectAL).

Keeps track of devices that have been opened, and allows high level OpenAL management.

Provides methods for loading [ALBuffer](#) objects from audio files.

The OpenAL 1.1 specification is available at <http://connect.creativelabs.com/openal/-Documentation>

Be sure to read through it (especially the part about distance models) as ObjectAL follows the OpenAL object model.

Alternatively, you may opt to use [OALSimpleAudio](#) for a simpler interface.

4.43.2 Method Documentation

4.43.2.1 - (NSString *) [bufferAsyncFromFile:](#) (NSString*) *filePath* [reduceToMono:](#)(bool) *reduceToMono* [target:](#)(id) *target* [selector:](#)(SEL) *selector*

Load an OpenAL buffer with the contents of an audio file asynchronously.

This method will schedule a request to have the buffer created and filled, and then call the specified selector with the newly created buffer.

The buffer's name will be the fully qualified URL of the path.

Returns the fully qualified URL of the path, which you can match up to the buffer name in your callback method.

See the class description note regarding sound file formats.

Parameters

<i>filePath</i>	The path of the file containing the audio data.
<i>reduceToMono</i>	If true, reduce the sample to mono (stereo samples don't support panning or positional audio).
<i>target</i>	The target to call when the buffer is loaded.
<i>selector</i>	The selector to invoke when the buffer is loaded.

Returns

The fully qualified URL of the path.

4.43.2.2 - (NSString *) bufferAsyncFromFile: (NSString*) *filePath* target:(id) *target* selector:(SEL) *selector*

Load an OpenAL buffer with the contents of an audio file asynchronously.

This method will schedule a request to have the buffer created and filled, and then call the specified selector with the newly created buffer.

The buffer's name will be the fully qualified URL of the path.

Returns the fully qualified URL of the path, which you can match up to the buffer name in your callback method.

See the class description note regarding sound file formats.

Parameters

<i>filePath</i>	The path of the file containing the audio data.
<i>target</i>	The target to call when the buffer is loaded.
<i>selector</i>	The selector to invoke when the buffer is loaded.

Returns

The fully qualified URL of the path.

4.43.2.3 - (NSString *) bufferAsyncFromUrl: (NSURL*) *url* reduceToMono:(bool) *reduceToMono* target:(id) *target* selector:(SEL) *selector*

Load an OpenAL buffer with the contents of a URL asynchronously.

This method will schedule a request to have the buffer created and filled, and then call the specified selector with the newly created buffer.

The buffer's name will be the fully qualified URL.

Returns the fully qualified URL, which you can match up to the buffer name in your callback method.

See the class description note regarding sound file formats.

Parameters

<i>url</i>	The URL of the file containing the audio data.
<i>reduceToMono</i>	If true, reduce the sample to mono (stereo samples don't support panning or positional audio).
<i>target</i>	The target to call when the buffer is loaded.
<i>selector</i>	The selector to invoke when the buffer is loaded.

Returns

The fully qualified URL of the path.

4.43.2.4 - (NSString *) bufferAsyncFromUrl: (NSURL*) *url* target:(id) *target* selector:(SEL) *selector*

Load an OpenAL buffer with the contents of a URL asynchronously.

This method will schedule a request to have the buffer created and filled, and then call the specified selector with the newly created buffer.

The buffer's name will be the fully qualified URL.

Returns the fully qualified URL, which you can match up to the buffer name in your callback method.

See the class description note regarding sound file formats.

Parameters

<i>url</i>	The URL of the file containing the audio data.
<i>target</i>	The target to call when the buffer is loaded.
<i>selector</i>	The selector to invoke when the buffer is loaded.

Returns

The fully qualified URL of the path.

4.43.2.5 - (ALBuffer *) bufferFromFile: (NSString*) filePath

Load an OpenAL buffer with the contents of an audio file.

The buffer's name will be the fully qualified URL of the path.

See the class description note regarding sound file formats.

Parameters

<i>filePath</i>	The path of the file containing the audio data.
-----------------	---

Returns

An [ALBuffer](#) containing the audio data.

4.43.2.6 - (ALBuffer *) bufferFromFile: (NSString*) filePath reduceToMono:(bool) reduceToMono

Load an OpenAL buffer with the contents of an audio file.

The buffer's name will be the fully qualified URL of the path.

See the class description note regarding sound file formats.

Parameters

<i>filePath</i>	The path of the file containing the audio data.
<i>reduceToMono</i>	If true, reduce the sample to mono (stereo samples don't support panning or positional audio).

Returns

An [ALBuffer](#) containing the audio data.

4.43.2.7 - (ALBuffer *) bufferFromUrl: (NSURL*) url

Load an OpenAL buffer with the contents of an audio file.

The buffer's name will be the fully qualified URL.

See the class description note regarding sound file formats.

Parameters

<i>url</i>	The URL of the file containing the audio data.
------------	--

Returns

An [ALBuffer](#) containing the audio data.

4.43.2.8 - (ALBuffer *) bufferFromUrl: (NSURL*) url reduceToMono:(bool) reduceToMono

Load an OpenAL buffer with the contents of an audio file.

The buffer's name will be the fully qualified URL.

See the class description note regarding sound file formats.

Parameters

<i>url</i>	The URL of the file containing the audio data.
<i>reduceToMono</i>	If true, reduce the sample to mono (stereo samples don't support panning or positional audio).

Returns

An [ALBuffer](#) containing the audio data.

4.43.2.9 - (void) clearAllBuffers

Clear all references to sound data from ALL buffers, managed or not.

4.43.2.10 - OpenALManager: (OpenALManager)

Singleton implementation providing "sharedInstance" and "purgeSharedInstance" methods.

- (OpenALManager*) sharedInstance: Get the shared singleton instance.

- (void) purgeSharedInstance: Purge (deallocate) the shared instance.

4.43.3 Member Data Documentation

4.43.3.1 - (NSMutableArray*) devices [protected]

All opened devices.

4.43.3.2 - (NSOperationQueue*) operationQueue [protected]

Operation queue for asynchronous loading.

4.43.3.3 - (OALSuspendHandler*) suspendHandler [protected]

Handles suspending and interrupting for this object.

4.43.4 Property Documentation

4.43.4.1 - (NSArray *) availableCaptureDevices [read], [nonatomic], [retain]

List of available capture devices (NSString*).

4.43.4.2 - (NSArray *) availableDevices [read], [nonatomic], [retain]

List of available playback devices (NSString*).

4.43.4.3 - (ALContext *) currentContext [read], [write], [nonatomic], [assign]

The current context (some context operations require the context to be the "current" one).

WEAK reference.

4.43.4.4 - (NSString *) defaultCaptureDeviceSpecifier [read], [nonatomic], [retain]

Name of the default capture device.

4.43.4.5 - (NSString *) defaultDeviceSpecifier [read], [nonatomic], [retain]

Name of the default playback device.

4.43.4.6 - (NSArray*) devices [read], [nonatomic], [retain]

List of all open devices (ALDevice*).

4.43.4.7 - (ALdouble) mixerOutputFrequency [read], [write], [nonatomic], [assign]

The frequency of the output mixer.

4.43.4.8 - (ALint) renderingQuality [read], [write], [nonatomic], [assign]

The rendering quality.

Can be one of:

- ALC_MAC_OSX_SPATIAL_RENDERING_QUALITY_HIGH
- ALC_MAC_OSX_SPATIAL_RENDERING_QUALITY_LOW
- ALC_IPHONE_SPATIAL_RENDERING_QUALITY_HEADPHONES (iOS only)

The documentation for this class was generated from the following files:

- OpenALManager.h
- OpenALManager.m

Index

- [<ALSoundSource>, 42](#)
- [<OALSuspendListener>, 155](#)
- [<OALSuspendManager>, 156](#)
- [ALBuffer, 15](#)
 - [bits, 17](#)
 - [bufferData, 17](#)
 - [bufferId, 17](#)
 - [bufferWithName:data:size:format:frequency:, 16](#)
 - [channels, 17](#)
 - [device, 17](#)
 - [duration, 18](#)
 - [format, 18](#)
 - [freeDataOnDestroy, 18](#)
 - [frequency, 18](#)
 - [initWithName:data:size:format:frequency:, 16](#)
 - [name, 18](#)
 - [parentBuffer, 17](#)
 - [size, 18](#)
 - [sliceWithName:offset:size:, 17](#)
- [ALCaptureDevice, 18](#)
 - [captureSamples, 21](#)
 - [device, 21](#)
 - [deviceWithDeviceSpecifier:frequency:format:bufferSize:, 19](#)
 - [extensions, 21](#)
 - [getProcAddress:, 20](#)
 - [initWithDeviceSpecifier:frequency:format:bufferSize:, 20](#)
 - [isExtensionPresent:, 20](#)
 - [majorVersion, 21](#)
 - [minorVersion, 21](#)
 - [moveSamples:toBuffer:, 20](#)
 - [startCapture, 21](#)
 - [stopCapture, 21](#)
- [ALChannelSource, 21](#)
 - [addChannel:, 24](#)
 - [addSource:, 24](#)
 - [channelWithSources:, 25](#)
 - [clearUnusedBuffers, 25](#)
 - [context, 29](#)
 - [currentFadeCallbackCount, 26](#)
 - [currentPanCallbackCount, 26](#)
 - [currentPitchCallbackCount, 26](#)
 - [defaultConeInnerAngle, 26](#)
 - [defaultConeOuterAngle, 26](#)
 - [defaultConeOuterGain, 27](#)
 - [defaultDirection, 27](#)
 - [defaultGain, 27](#)
 - [defaultLooping, 27](#)
 - [defaultMaxDistance, 27](#)
 - [defaultMaxGain, 27](#)
 - [defaultMinGain, 27](#)
 - [defaultPitch, 27](#)
 - [defaultPosition, 27](#)
 - [defaultReferenceDistance, 27](#)
 - [defaultReverbObstruction, 27](#)
 - [defaultReverbOcclusion, 27](#)
 - [defaultReverbSendLevel, 28](#)
 - [defaultRolloffFactor, 28](#)
 - [defaultSourceRelative, 28](#)
 - [defaultSourceType, 28](#)
 - [defaultVelocity, 28](#)
 - [defaultsInitialized, 28](#)
 - [expectedFadeCallbackCount, 28](#)
 - [expectedPanCallbackCount, 28](#)
 - [expectedPitchCallbackCount, 28](#)
 - [fadeCompleteSelector, 28](#)
 - [fadeCompleteTarget, 28](#)
 - [initWithSources:, 25](#)
 - [panCompleteSelector, 28](#)
 - [panCompleteTarget, 29](#)
 - [pitchCompleteSelector, 29](#)
 - [pitchCompleteTarget, 29](#)
 - [removeBuffersNamed:, 25](#)
 - [removeSource:, 25](#)
 - [reservedSources, 29](#)
 - [resetToDefault, 26](#)
 - [setDefaultFromSource:, 26](#)
 - [sourcePool, 29](#)
 - [splitChannelWithSources:, 26](#)
- [ALContext, 29](#)
 - [alVersion, 34](#)
 - [attributes, 33, 34](#)
 - [clearBuffers, 31](#)
 - [context, 34](#)
 - [contextOnDevice:attributes:, 31](#)
 - [contextOnDevice:outputFrequency:refreshIntervals:synchronousContext:monoSources:stereoSources:, 31](#)
 - [device, 34](#)
 - [distanceModel, 34](#)
 - [dopplerFactor, 34](#)
 - [ensureContextIsCurrent, 32](#)
 - [extensions, 34](#)
 - [getProcAddress:, 32](#)
 - [initOnDevice:attributes:, 32](#)
 - [initOnDevice:outputFrequency:refreshIntervals:synchronousContext:monoSources:stereo-](#)

- Sources:, 33
- isExtensionPresent:, 33
- listener, 34
- process, 33
- renderer, 35
- sources, 33, 35
- speedOfSound, 35
- stopAllSounds, 33
- suspendHandler, 34
- vendor, 35
- ALContext(), 35
- ALDevice, 35
 - clearBuffers, 37
 - contexts, 38
 - device, 38
 - deviceWithDeviceSpecifier:, 37
 - extensions, 38
 - getProcAddress:, 37
 - initWithDeviceSpecifier:, 37
 - isExtensionPresent:, 37
 - majorVersion, 38
 - minorVersion, 38
 - suspendHandler, 38
- ALListener, 38
 - context, 39
 - gain, 39
 - globalReverbLevel, 40
 - muted, 40
 - orientation, 40
 - position, 40
 - reverbEQBandwidth, 40
 - reverbEQFrequency, 40
 - reverbEQGain, 40
 - reverbOn, 40
 - reverbRoomType, 40
 - suspendHandler, 39
 - velocity, 41
- ALOrientation, 41
 - at, 41
 - up, 41
- ALPoint, 42
 - x, 42
 - y, 42
 - z, 42
- ALSoundSource-p
 - clear, 44
 - coneInnerAngle, 47
 - coneOuterAngle, 47
 - coneOuterGain, 47
 - direction, 47
 - fadeTo:duration:target:selector:, 44
 - gain, 47
 - interruptible, 47
 - looping, 47
 - maxDistance, 47
 - maxGain, 47
 - minGain, 47
 - muted, 47
 - pan, 47
 - panTo:duration:target:selector:, 45
 - paused, 48
 - pitch, 48
 - pitchTo:duration:target:selector:, 45
 - play:, 45
 - play:gain:pitch:pan:loop:, 45
 - play:loop:, 46
 - playing, 48
 - position, 48
 - referenceDistance, 48
 - reverbObstruction, 48
 - reverbOcclusion, 48
 - reverbSendLevel, 48
 - rewind, 46
 - rolloffFactor, 48
 - sourceRelative, 48
 - sourceType, 48
 - stop, 46
 - stopActions, 46
 - stopFade, 46
 - stopPan, 46
 - stopPitch, 46
 - velocity, 49
 - volume, 49
- ALSoundSourcePool, 49
 - addSource:, 50
 - getFreeSource:, 50
 - pool, 50
 - removeSource:, 50
 - sources, 50, 51
- ALSoundSourcePool(Private), 51
 - moveToHead:, 51
- ALSource, 51
 - abortPlaybackResume, 56
 - buffer, 57
 - buffersProcessed, 57
 - buffersQueued, 57
 - context, 57
 - gainAction, 56
 - initOnContext:, 53
 - offsetInBytes, 57
 - offsetInSamples, 57
 - offsetInSeconds, 57
 - panAction, 56
 - pitchAction, 56
 - play, 53
 - queueBuffer:, 54
 - queueBuffer:repeats:, 54
 - queueBuffers:, 54
 - queueBuffers:repeats:, 54
 - registerNotification:callback:userData:, 55
 - shadowState, 56
 - source, 55
 - sourceId, 57
 - sourceOnContext:, 55
 - state, 57
 - suspendHandler, 56

- unqueueBuffer:, 55
- unqueueBuffers:, 55
- unregisterAllNotifications, 56
- unregisterNotification:, 56
- ALVector, 58
 - x, 58
 - y, 58
 - z, 58
- ALWrapper, 58
 - addNotification:onSource:callback:userData:, 64
 - asaGetListenerb:, 64
 - asaGetListenerf:, 64
 - asaGetListeneri:, 64
 - asaGetSourceb:property:, 65
 - asaGetSourcef:property:, 65
 - asaGetSourcei:property:, 65
 - asaListenerb:value:, 65
 - asaListenerf:value:, 66
 - asaListeneri:value:, 66
 - asaSourceb:property:value:, 66
 - asasourcef:property:value:, 66
 - asasourcei:property:value:, 67
 - buffer3f:parameter:v1:v2:v3:, 67
 - buffer3i:parameter:v1:v2:v3:, 67
 - bufferData:format:data:size:frequency:, 67
 - bufferDataStatic:format:data:size:frequency:, 68
 - bufferf:parameter:value:, 68
 - bufferfv:parameter:values:, 68
 - bufferi:parameter:value:, 68
 - bufferiv:parameter:values:, 69
 - captureSamples:buffer:numSamples:, 69
 - closeCaptureDevice:, 69
 - closeDevice:, 69
 - createContext:attributes:, 70
 - deleteBuffer:, 70
 - deleteBuffers:numBuffers:, 70
 - deleteSource:, 70
 - deleteSources:numSources:, 71
 - destroyContext:, 71
 - disable:, 71
 - distanceModel:, 71
 - dopplerFactor:, 71
 - enable:, 72
 - genBuffer, 72
 - genBuffers:numBuffers:, 72
 - genSource, 72
 - genSources:numSources:, 72
 - getBoolean:, 73
 - getBooleanv:values:, 73
 - getBuffer3f:parameter:v1:v2:v3:, 73
 - getBuffer3i:parameter:v1:v2:v3:, 73
 - getBufferf:parameter:, 74
 - getBufferfv:parameter:values:, 74
 - getBufferi:parameter:, 74
 - getBufferiv:parameter:values:, 75
 - getContextsDevice:, 75
 - getContextsDevice:deviceReference:, 75
 - getCurrentContext, 75
 - getDouble:, 75
 - getDoublev:values:, 76
 - getEnumValue:, 76
 - getEnumValue:name:, 76
 - getFloat:, 76
 - getFloatv:values:, 77
 - getInteger:, 77
 - getInteger:attribute:, 77
 - getInteger:attribute:size:data:, 77
 - getInteger:values:, 78
 - getListener3f:v1:v2:v3:, 78
 - getListener3i:v1:v2:v3:, 78
 - getListenerf:, 78
 - getListenerfv:values:, 79
 - getListeneri:, 79
 - getListeneriv:values:, 79
 - getMixerOutputDataRate, 79
 - getNullSeparatedStringList:, 79
 - getNullSeparatedStringList:attribute:, 80
 - getProcAddress:, 80
 - getProcAddress:name:, 80
 - getRenderingQuality, 80
 - getSource3f:parameter:v1:v2:v3:, 81
 - getSource3i:parameter:v1:v2:v3:, 81
 - getSourcef:parameter:, 81
 - getSourcefv:parameter:values:, 81
 - getSourcei:parameter:, 82
 - getSourceiv:parameter:values:, 82
 - getSpaceSeparatedStringList:, 82
 - getSpaceSeparatedStringList:attribute:, 82
 - getString:, 83
 - getString:attribute:, 83
 - isBuffer:, 83
 - isEnabled:, 83
 - isExtensionPresent:, 84
 - isExtensionPresent:name:, 84
 - isSource:, 84
 - listener3f:v1:v2:v3:, 84
 - listener3i:v1:v2:v3:, 85
 - listenerf:value:, 85
 - listenerfv:values:, 85
 - listeneri:value:, 85
 - listeneriv:values:, 86
 - makeContextCurrent:, 86
 - makeContextCurrent:deviceReference:, 86
 - openCaptureDevice:frequency:format:bufferSize:, 86
 - openDevice:, 87
 - processContext:, 87
 - removeNotification:onSource:callback:userData:, 87
 - setMixerOutputDataRate:, 87
 - setRenderingQuality:, 87
 - source3f:parameter:v1:v2:v3:, 88
 - source3i:parameter:v1:v2:v3:, 88
 - sourcePause:, 89
 - sourcePausev:numSources:, 90
 - sourcePlay:, 90

- sourcePlayv:numSources:, 90
- sourceQueueBuffers:numBuffers:bufferIds:, 90
- sourceRewind:, 91
- sourceRewindv:numSources:, 91
- sourceStop:, 91
- sourceStopv:numSources:, 91
- sourceUnqueueBuffers:numBuffers:bufferIds:, 91
- sourcecf:parameter:value:, 88
- sourcecfv:parameter:values:, 89
- sourceci:parameter:value:, 89
- sourceciv:parameter:values:, 89
- speedOfSound:, 92
- startCapture:, 92
- stopCapture:, 92
- suspendContext:, 92
- ALWrapper(Private), 93
 - checkIfSuccessful, 93
 - checkIfSuccessfulWithDevice, 94
 - decodeNullSeparatedStringList:, 94
 - decodeSpaceSeparatedStringList:, 94
- abortPlaybackResume
 - ALSource, 56
- action_
 - OALTargetedAction, 159
- actionIndex_
 - OALSequentialActions, 140
- actionWithCallTarget:selector:
 - OALCallAction, 124
- actionWithCallTarget:selector:withObject:
 - OALCallAction, 124
- actionWithCallTarget:selector:withObject:withObject:
 - OALCallAction, 125
- actionWithDuration:delta:
 - OALMoveByAction, 131
- actionWithDuration:position:
 - OALMoveToAction, 133
- actionWithDuration:propertyKey:endValue:
 - OALPropertyAction, 137
- actionWithDuration:propertyKey:startValue:endValue:
 - OALPropertyAction, 137
- actionWithPosition:
 - OALPlaceAction, 135
- actionWithShape:phase:action:
 - OALEaseAction, 129
- actionWithTarget:action:
 - OALTargetedAction, 159
- actionWithUnitsPerSecond:delta:
 - OALMoveByAction, 131
- actionWithUnitsPerSecond:position:
 - OALMoveToAction, 133
- actions
 - OALConcurrentActions, 128
 - OALSequentialActions, 141
- actions:
 - OALConcurrentActions, 127
 - OALSequentialActions, 140
- actionsFromArray:
 - OALConcurrentActions, 127
 - OALSequentialActions, 140
- actionsToAdd
 - OALActionManager, 101
- actionsToRemove
 - OALActionManager, 101
- addChannel:
 - ALChannelSource, 24
- addNotification:onSource:callback:userData:
 - ALWrapper, 64
- addSource:
 - ALChannelSource, 24
 - ALSoundSourcePool, 50
- addSuspendListener:
 - OALSuspendHandler, 153
 - OALSuspendManager-p, 157
- alVersion
 - ALContext, 34
- allowIpod
 - OALAudioSession, 107
 - OALSimpleAudio, 149
- asaGetListenerb:
 - ALWrapper, 64
- asaGetListenerf:
 - ALWrapper, 64
- asaGetListeneri:
 - ALWrapper, 64
- asaGetSourcecb:property:
 - ALWrapper, 65
- asaGetSourcecf:property:
 - ALWrapper, 65
- asaGetSourceci:property:
 - ALWrapper, 65
- asaListenerb:value:
 - ALWrapper, 65
- asaListenerf:value:
 - ALWrapper, 66
- asaListeneri:value:
 - ALWrapper, 66
- asaSourcecb:property:value:
 - ALWrapper, 66
- asaSourcecf:property:value:
 - ALWrapper, 66
- asaSourceci:property:value:
 - ALWrapper, 67
- at
 - ALOrientation, 41
- attributes
 - ALContext, 33, 34
- audioDataWithStartFrame:numFrames:bufferSize:
 - OALAudioFile, 103
- audioRoute
 - OALAudioSession, 107
- audioSessionActive
 - OALAudioSession, 108
- audioSessionCategory
 - OALAudioSession, 108
- audioSessionWasActive
 - OALAudioSession, 107

- autoPreload
 - OALAudioTrack, [119](#)
- availableCaptureDevices
 - OpenALManager, [166](#)
- availableDevices
 - OpenALManager, [166](#)
- averagePowerForChannel:
 - OALAudioTrack, [112](#)
- backgroundTrack
 - OALSimpleAudio, [149](#)
- backgroundTrackURL
 - OALSimpleAudio, [149](#)
- bgMuted
 - OALSimpleAudio, [150](#)
- bgPaused
 - OALSimpleAudio, [150](#)
- bgPlaying
 - OALSimpleAudio, [150](#)
- bgVolume
 - OALSimpleAudio, [150](#)
- bits
 - ALBuffer, [17](#)
- buffer
 - ALSource, [57](#)
- buffer3f:parameter:v1:v2:v3:
 - ALWrapper, [67](#)
- buffer3i:parameter:v1:v2:v3:
 - ALWrapper, [67](#)
- bufferAsyncFromFile:reduceToMono:target:selector:
 - OpenALManager, [163](#)
- bufferAsyncFromFile:target:selector:
 - OpenALManager, [164](#)
- bufferAsyncFromUrl:reduceToMono:target:selector:
 - OpenALManager, [164](#)
- bufferAsyncFromUrl:target:selector:
 - OpenALManager, [164](#)
- bufferData
 - ALBuffer, [17](#)
- bufferData:format:data:size:frequency:
 - ALWrapper, [67](#)
- bufferDataStatic:format:data:size:frequency:
 - ALWrapper, [68](#)
- bufferFromFile:
 - OpenALManager, [165](#)
- bufferFromFile:reduceToMono:
 - OpenALManager, [165](#)
- bufferFromUrl:
 - OpenALManager, [165](#)
- bufferFromUrl:reduceToMono:
 - OALAudioFile, [103](#)
 - OpenALManager, [166](#)
- bufferId
 - ALBuffer, [17](#)
- bufferNamed:startFrame:numFrames:
 - OALAudioFile, [104](#)
- bufferWithName:data:size:format:frequency:
 - ALBuffer, [16](#)
- bufferf:parameter:value:
 - ALWrapper, [68](#)
- bufferfv:parameter:values:
 - ALWrapper, [68](#)
- bufferi:parameter:value:
 - ALWrapper, [68](#)
- bufferiv:parameter:values:
 - ALWrapper, [69](#)
- buffersProcessed
 - ALSource, [57](#)
- buffersQueued
 - ALSource, [57](#)
- callTarget_
 - OALCallAction, [126](#)
- captureSamples
 - ALCaptureDevice, [21](#)
- captureSamples:buffer:numSamples:
 - ALWrapper, [69](#)
- channel
 - OALSimpleAudio, [150](#)
- channelWithSources:
 - ALChannelSource, [25](#)
- channels
 - ALBuffer, [17](#)
- checkIfSuccessful
 - ALWrapper(Private), [93](#)
- checkIfSuccessfulWithDevice
 - ALWrapper(Private), [94](#)
- clear
 - ALSoundSource-p, [44](#)
 - OALAudioTrack, [112](#)
- clearAllBuffers
 - OpenALManager, [166](#)
- clearBuffers
 - ALContext, [31](#)
 - ALDevice, [37](#)
- clearUnusedBuffers
 - ALChannelSource, [25](#)
- closeCaptureDevice:
 - ALWrapper, [69](#)
- closeDevice:
 - ALWrapper, [69](#)
- coneInnerAngle
 - ALSoundSource-p, [47](#)
- coneOuterAngle
 - ALSoundSource-p, [47](#)
- coneOuterGain
 - ALSoundSource-p, [47](#)
- context
 - ALChannelSource, [29](#)
 - ALContext, [34](#)
 - ALListener, [39](#)
 - ALSource, [57](#)
 - OALSimpleAudio, [150](#)
- contextOnDevice:attributes:
 - ALContext, [31](#)
- contextOnDevice:outputFrequency:refreshIntervals-
:synchronousContext:monoSources:stereo-
Sources:
 -

- ALContext, [31](#)
- contexts
 - ALDevice, [38](#)
- createContext:attributes:
 - ALWrapper, [70](#)
- currentContext
 - OpenALManager, [167](#)
- currentFadeCallbackCount
 - ALChannelSource, [26](#)
- currentPanCallbackCount
 - ALChannelSource, [26](#)
- currentPitchCallbackCount
 - ALChannelSource, [26](#)
- currentTime
 - OALAudioTrack, [119](#)
- currentlyLoadedUrl
 - OALAudioTrack, [119](#)
- decodeNullSeparatedStringList:
 - ALWrapper(Private), [94](#)
- decodeSpaceSeparatedStringList:
 - ALWrapper(Private), [94](#)
- defaultBundle
 - OALTools, [160](#)
- defaultCaptureDeviceSpecifier
 - OpenALManager, [167](#)
- defaultConeInnerAngle
 - ALChannelSource, [26](#)
- defaultConeOuterAngle
 - ALChannelSource, [26](#)
- defaultConeOuterGain
 - ALChannelSource, [27](#)
- defaultDeviceSpecifier
 - OpenALManager, [167](#)
- defaultDirection
 - ALChannelSource, [27](#)
- defaultGain
 - ALChannelSource, [27](#)
- defaultLooping
 - ALChannelSource, [27](#)
- defaultMaxDistance
 - ALChannelSource, [27](#)
- defaultMaxGain
 - ALChannelSource, [27](#)
- defaultMinGain
 - ALChannelSource, [27](#)
- defaultPitch
 - ALChannelSource, [27](#)
- defaultPosition
 - ALChannelSource, [27](#)
- defaultReferenceDistance
 - ALChannelSource, [27](#)
- defaultReverbObstruction
 - ALChannelSource, [27](#)
- defaultReverbOcclusion
 - ALChannelSource, [27](#)
- defaultReverbSendLevel
 - ALChannelSource, [28](#)
- defaultRolloffFactor
 - ALChannelSource, [28](#)
- defaultSourceRelative
 - ALChannelSource, [28](#)
- defaultSourceType
 - ALChannelSource, [28](#)
- defaultVelocity
 - ALChannelSource, [28](#)
- defaultsInitialized
 - ALChannelSource, [28](#)
- delegate
 - OALAudioTrack, [119](#)
- deleteBuffer:
 - ALWrapper, [70](#)
- deleteBuffers:numBuffers:
 - ALWrapper, [70](#)
- deleteSource:
 - ALWrapper, [70](#)
- deleteSources:numSources:
 - ALWrapper, [71](#)
- delta
 - OALMoveByAction, [132](#)
 - OALMoveToAction, [134](#)
- destroyContext:
 - ALWrapper, [71](#)
- device
 - ALBuffer, [17](#)
 - ALCaptureDevice, [21](#)
 - ALContext, [34](#)
 - ALDevice, [38](#)
 - OALSimpleAudio, [150](#)
- deviceCurrentTime
 - OALAudioTrack, [120](#)
- deviceTimePoller
 - OALAudioTracks, [122](#)
- deviceWithDeviceSpecifier:
 - ALDevice, [37](#)
- deviceWithDeviceSpecifier:frequency:format:bufferSize:
 - ALCaptureDevice, [19](#)
- devices
 - OpenALManager, [166](#), [167](#)
- direction
 - ALSoundSource-p, [47](#)
- disable:
 - ALWrapper, [71](#)
- distanceModel
 - ALContext, [34](#)
- distanceModel:
 - ALWrapper, [71](#)
- dopplerFactor
 - ALContext, [34](#)
- dopplerFactor:
 - ALWrapper, [71](#)
- duration
 - ALBuffer, [18](#)
 - OALAction, [100](#)
 - OALAudioTrack, [120](#)
- easeFunctionForShape:phase:
 - OALEaseAction, [129](#)

- effectsMuted
 - OALSimpleAudio, [150](#)
- effectsPaused
 - OALSimpleAudio, [150](#)
- effectsVolume
 - OALSimpleAudio, [150](#)
- elapsed
 - OALAction, [100](#)
- enable:
 - ALWrapper, [72](#)
- endValue
 - OALPropertyAction, [138](#)
- ensureContextIsCurrent
 - ALContext, [32](#)
- expectedFadeCallbackCount
 - ALChannelSource, [28](#)
- expectedPanCallbackCount
 - ALChannelSource, [28](#)
- expectedPitchCallbackCount
 - ALChannelSource, [28](#)
- extensions
 - ALCaptureDevice, [21](#)
 - ALContext, [34](#)
 - ALDevice, [38](#)
- fadeCompleteSelector
 - ALChannelSource, [28](#)
- fadeCompleteTarget
 - ALChannelSource, [28](#)
- fadeTo:duration:target:selector:
 - ALSoundSource-p, [44](#)
 - OALAudioTrack, [112](#)
- fileHandle
 - OALAudioFile, [104](#)
- fileWithURL:reduceToMono:
 - OALAudioFile, [104](#)
- forceEndInterruption
 - OALAudioSession, [107](#)
- forcedTarget
 - OALTargetedAction, [159](#)
- format
 - ALBuffer, [18](#)
- freeDataOnDestroy
 - ALBuffer, [18](#)
- frequency
 - ALBuffer, [18](#)
- gain
 - ALListener, [39](#)
 - ALSoundSource-p, [47](#)
 - OALAudioTrack, [120](#)
- gainAction
 - ALSource, [56](#)
 - OALAudioTrack, [119](#)
- genBuffer
 - ALWrapper, [72](#)
- genBuffers:numBuffers:
 - ALWrapper, [72](#)
- genSource
 - ALWrapper, [72](#)
- genSources:numSources:
 - ALWrapper, [72](#)
- getBoolean:
 - ALWrapper, [73](#)
- getBooleanv:values:
 - ALWrapper, [73](#)
- getBuffer3f:parameter:v1:v2:v3:
 - ALWrapper, [73](#)
- getBuffer3i:parameter:v1:v2:v3:
 - ALWrapper, [73](#)
- getBufferf:parameter:
 - ALWrapper, [74](#)
- getBufferfv:parameter:values:
 - ALWrapper, [74](#)
- getBufferi:parameter:
 - ALWrapper, [74](#)
- getBufferiv:parameter:values:
 - ALWrapper, [75](#)
- getContextsDevice:
 - ALWrapper, [75](#)
- getContextsDevice:deviceReference:
 - ALWrapper, [75](#)
- getCurrentContext
 - ALWrapper, [75](#)
- getDouble:
 - ALWrapper, [75](#)
- getDoublev:values:
 - ALWrapper, [76](#)
- getEnumValue:
 - ALWrapper, [76](#)
- getEnumValue:name:
 - ALWrapper, [76](#)
- getFloat:
 - ALWrapper, [76](#)
- getFloatv:values:
 - ALWrapper, [77](#)
- getFreeSource:
 - ALSoundSourcePool, [50](#)
- getInteger:
 - ALWrapper, [77](#)
- getInteger:attribute:
 - ALWrapper, [77](#)
- getIntegerv:attribute:size:data:
 - ALWrapper, [77](#)
- getIntegerv:values:
 - ALWrapper, [78](#)
- getListener3f:v1:v2:v3:
 - ALWrapper, [78](#)
- getListener3i:v1:v2:v3:
 - ALWrapper, [78](#)
- getListenerf:
 - ALWrapper, [78](#)
- getListenerfv:values:
 - ALWrapper, [79](#)
- getListeneri:
 - ALWrapper, [79](#)
- getListeneriv:values:

- ALWrapper, 79
- getMixerOutputDataRate
 - ALWrapper, 79
- getNullSeparatedStringList:
 - ALWrapper, 79
- getNullSeparatedStringList:attribute:
 - ALWrapper, 80
- getProcAddress:
 - ALCaptureDevice, 20
 - ALContext, 32
 - ALDevice, 37
 - ALWrapper, 80
- getProcAddress:name:
 - ALWrapper, 80
- getRenderingQuality
 - ALWrapper, 80
- getSource3f:parameter:v1:v2:v3:
 - ALWrapper, 81
- getSource3i:parameter:v1:v2:v3:
 - ALWrapper, 81
- getSourcef:parameter:
 - ALWrapper, 81
- getSourcefv:parameter:values:
 - ALWrapper, 81
- getSourcei:parameter:
 - ALWrapper, 82
- getSourceiv:parameter:values:
 - ALWrapper, 82
- getSpaceSeparatedStringList:
 - ALWrapper, 82
- getSpaceSeparatedStringList:attribute:
 - ALWrapper, 82
- getString:
 - ALWrapper, 83
- getString:attribute:
 - ALWrapper, 83
- globalReverbLevel
 - ALListener, 40
- handleInterruptions
 - OALAudioSession, 108
- handlerWithTarget:selector:
 - OALSuspendHandler, 153
- handlingErrorNotification
 - OALAudioSession, 107
- hardwareMuted
 - OALAudioSession, 108
- hardwareVolume
 - OALAudioSession, 108
- honorSilentSwitch
 - OALAudioSession, 108
 - OALSimpleAudio, 150
- IOSVersion, 94
 - version, 95
- initWithContext:
 - ALSource, 53
- initWithDevice:attributes:
 - ALContext, 32
- initWithDevice:outputFrequency:refreshIntervals:synchronous-Context:monoSources:stereoSources:
 - ALContext, 33
- initWithActions:
 - OALConcurrentActions, 127
 - OALSequentialActions, 140
- initWithCallTarget:selector:
 - OALCallAction, 125
- initWithCallTarget:selector:withObject:
 - OALCallAction, 125
- initWithCallTarget:selector:withObject:withObject:
 - OALCallAction, 125
- initWithDeviceSpecifier:
 - ALDevice, 37
- initWithDeviceSpecifier:frequency:format:bufferSize:
 - ALCaptureDevice, 20
- initWithDuration:
 - OALAction, 99
- initWithDuration:delta:
 - OALMoveByAction, 131
- initWithDuration:position:
 - OALMoveToAction, 133
- initWithDuration:propertyKey:endValue:
 - OALPropertyAction, 137
- initWithDuration:propertyKey:startValue:endValue:
 - OALPropertyAction, 138
- initWithName:data:size:format:frequency:
 - ALBuffer, 16
- initWithPosition:
 - OALPlaceAction, 135
- initWithShape:phase:action:
 - OALEaseAction, 129
- initWithSources:
 - ALChannelSource, 25
- initWithTarget:action:
 - OALTargetedAction, 159
- initWithTarget:selector:
 - OALSuspendHandler, 153
- initWithUnitsPerSecond:delta:
 - OALMoveByAction, 131
- initWithUnitsPerSecond:position:
 - OALMoveToAction, 134
- initWithUrl:reduceToMono:
 - OALAudioFile, 104
- interruptLock
 - OALSuspendHandler, 154
- interrupted
 - OALSimpleAudio, 151
 - OALSuspendHandler, 154
 - OALSuspendListener-p, 156
- interruptible
 - ALSoundSource-p, 47
- ipodDucking
 - OALAudioSession, 108
- ipodPlaying
 - OALAudioSession, 108
- isBuffer:
 - ALWrapper, 83

- isEnabled:
 - ALWrapper, 83
- isExtensionPresent:
 - ALCaptureDevice, 20
 - ALContext, 33
 - ALDevice, 37
 - ALWrapper, 84
- isExtensionPresent:name:
 - ALWrapper, 84
- isSource:
 - ALWrapper, 84
- lastResetTime
 - OALAudioSession, 107
- lastTimestamp
 - OALActionManager, 101
- listener
 - ALContext, 34
- listener3f:v1:v2:v3:
 - ALWrapper, 84
- listener3i:v1:v2:v3:
 - ALWrapper, 85
- listenerf:value:
 - ALWrapper, 85
- listenerfv:values:
 - ALWrapper, 85
- listeneri:value:
 - ALWrapper, 85
- listeneriv:values:
 - ALWrapper, 86
- listeners
 - OALSuspendHandler, 154
- looping
 - ALSoundSource-p, 47
- majorVersion
 - ALCaptureDevice, 21
 - ALDevice, 38
- makeContextCurrent:
 - ALWrapper, 86
- makeContextCurrent:deviceReference:
 - ALWrapper, 86
- manualSuspendLock
 - OALSuspendHandler, 154
- manualSuspendStates
 - OALSuspendHandler, 154
- manuallySuspended
 - OALSimpleAudio, 151
 - OALSuspendHandler, 154
 - OALSuspendListener-p, 156
- maxDistance
 - ALSoundSource-p, 47
- maxGain
 - ALSoundSource-p, 47
- meteringEnabled
 - OALAudioTrack, 120
- minGain
 - ALSoundSource-p, 47
- minorVersion
 - ALCaptureDevice, 21
 - ALDevice, 38
- mixerOutputFrequency
 - OpenALManager, 167
- moveSamples:toBuffer:
 - ALCaptureDevice, 20
- moveToHead:
 - ALSoundSourcePool(Private), 51
- mutableArrayUsingWeakReferences
 - NSMutableArray(WeakReferences), 96
- mutableArrayUsingWeakReferencesWithCapacity:
 - NSMutableArray(WeakReferences), 96
- mutableDictionaryUsingWeakReferences
 - NSMutableDictionary(WeakReferences), 97
- mutableDictionaryUsingWeakReferencesWithCapacity:
 - NSMutableDictionary(WeakReferences), 97
- muted
 - ALListener, 40
 - ALSoundSource-p, 47
 - OALAudioTrack, 120
 - OALAudioTracks, 123
 - OALSimpleAudio, 151
- NSMutableArray(WeakReferences), 95
 - mutableArrayUsingWeakReferences, 96
 - mutableArrayUsingWeakReferencesWithCapacity:, 96
 - newMutableArrayUsingWeakReferences, 96
 - newMutableArrayUsingWeakReferencesWithCapacity:, 96
- NSMutableDictionary(WeakReferences), 96
 - mutableDictionaryUsingWeakReferences, 97
 - mutableDictionaryUsingWeakReferencesWithCapacity:, 97
 - newMutableDictionaryUsingWeakReferences, 97
 - newMutableDictionaryUsingWeakReferencesWithCapacity:, 97
- name
 - ALBuffer, 18
- newMutableArrayUsingWeakReferences
 - NSMutableArray(WeakReferences), 96
- newMutableArrayUsingWeakReferencesWithCapacity:
 - NSMutableArray(WeakReferences), 96
- newMutableDictionaryUsingWeakReferences
 - NSMutableDictionary(WeakReferences), 97
- newMutableDictionaryUsingWeakReferencesWithCapacity:
 - NSMutableDictionary(WeakReferences), 97
- notifyAudioSessionError:function:description:
 - OALTools, 160
- notifyExtAudioError:function:description:
 - OALTools, 160
- numObjects_
 - OALCallAction, 126
- numberOfChannels
 - OALAudioTrack, 120
- numberOfLoops
 - OALAudioTrack, 120

- OALAction, 97
 - duration, 100
 - elapsed, 100
 - initWithDuration:, 99
 - prepareWithTarget:, 99
 - runWithTarget:, 99
 - running, 100
 - runningInManager_, 100
 - startAction, 99
 - stopAction, 99
 - target, 100
 - updateCompletion:, 99
- OALActionManager, 100
 - actionsToAdd, 101
 - actionsToRemove, 101
 - lastTimestamp, 101
 - stepTimer, 102
 - stopAllActions, 101
 - targetActions, 102
 - targets, 102
- OALAudioFile, 102
 - audioDataWithStartFrame:numFrames:bufferSize:, 103
 - bufferFromUrl:reduceToMono:, 103
 - bufferNamed:startFrame:numFrames:, 104
 - fileHandle, 104
 - fileWithUrl:reduceToMono:, 104
 - initWithUrl:reduceToMono:, 104
 - originalChannelsPerFrame, 104
 - reduceToMono, 105
 - streamDescription, 105
 - totalFrames, 105
 - url, 105
- OALAudioSession, 105
 - allowIpad, 107
 - audioRoute, 107
 - audioSessionActive, 108
 - audioSessionCategory, 108
 - audioSessionWasActive, 107
 - forceEndInterruption, 107
 - handleInterruptions, 108
 - handlingErrorNotification, 107
 - hardwareMuted, 108
 - hardwareVolume, 108
 - honorSilentSwitch, 108
 - ipodDucking, 108
 - ipodPlaying, 108
 - lastResetTime, 107
 - preferredIOBufferDuration, 108
 - suspendHandler, 107
 - useHardwareIfAvailable, 109
- OALAudioTrack, 109
 - autoPreload, 119
 - averagePowerForChannel:, 112
 - clear, 112
 - currentTime, 119
 - currentlyLoadedUrl, 119
 - delegate, 119
 - deviceCurrentTime, 120
 - duration, 120
 - fadeTo:duration:target:selector:, 112
 - gain, 120
 - gainAction, 119
 - meteringEnabled, 120
 - muted, 120
 - numberOfChannels, 120
 - numberOfLoops, 120
 - operationQueue, 119
 - pan, 120
 - panAction, 119
 - panTo:duration:target:selector:, 112
 - paused, 121
 - peakPowerForChannel:, 113
 - play, 113
 - playAfterTrack:, 113
 - playAfterTrack:timeAdjust:, 113
 - playAtTime:, 114
 - playFile:, 114
 - playFile:loops:, 114
 - playFileAsync:loops:target:selector:, 114
 - playFileAsync:target:selector:, 115
 - playUrl:, 115
 - playUrl:loops:, 115
 - playUrlAsync:loops:target:selector:, 115
 - playUrlAsync:target:selector:, 115
 - player, 121
 - playing, 121
 - preloadFile:, 116
 - preloadFile:seekTime:, 116
 - preloadFileAsync:seekTime:target:selector:, 116
 - preloadFileAsync:target:selector:, 117
 - preloadUrl:, 117
 - preloadUrl:seekTime:, 117
 - preloadUrlAsync:seekTime:target:selector:, 117
 - preloadUrlAsync:target:selector:, 118
 - preloaded, 121
 - simulatorPlayerRef, 119
 - stop, 118
 - stopActions, 118
 - stopFade, 118
 - stopPan, 118
 - suspendHandler, 119
 - track, 118
 - updateMeters, 118
 - volume, 121
- OALAudioTracks, 121
 - deviceTimePoller, 122
 - muted, 123
 - paused, 123
 - stopAllTracks, 122
 - suspendHandler, 122
 - tracks, 122, 123
- OALCallAction, 123
 - actionWithCallTarget:selector:, 124
 - actionWithCallTarget:selector:withObject:, 124

- actionWithCallTarget:selector:withObject:withObject:, 125
 - callTarget_, 126
 - initWithCallTarget:selector:, 125
 - initWithCallTarget:selector:withObject:, 125
 - initWithCallTarget:selector:withObject:withObject:, 125
 - numObjects_, 126
 - object1_, 126
 - object2_, 126
 - selector_, 126
- OALConcurrentActions, 126
 - actions, 128
 - actions:, 127
 - actionsFromArray:, 127
 - initWithActions:, 127
- OALEaseAction, 128
 - actionWithShape:phase:action:, 129
 - easeFunctionForShape:phase:, 129
 - initWithShape:phase:action:, 129
- OALEaseAction(), 130
- OALMoveByAction, 130
 - actionWithDuration:delta:, 131
 - actionWithUnitsPerSecond:delta:, 131
 - delta, 132
 - initWithDuration:delta:, 131
 - initWithUnitsPerSecond:delta:, 131
 - startPoint, 132
 - unitsPerSecond, 132
- OALMoveToAction, 132
 - actionWithDuration:position:, 133
 - actionWithUnitsPerSecond:position:, 133
 - delta, 134
 - initWithDuration:position:, 133
 - initWithUnitsPerSecond:position:, 134
 - position, 134
 - startPoint, 134
 - unitsPerSecond, 134
- OALPlaceAction, 135
 - actionWithPosition:, 135
 - initWithPosition:, 135
 - position, 136
- OALPropertyAction, 136
 - actionWithDuration:propertyKey:endValue:, 137
 - actionWithDuration:propertyKey:startValue:endValue:, 137
 - endValue, 138
 - initWithDuration:propertyKey:endValue:, 137
 - initWithDuration:propertyKey:startValue:endValue:, 138
 - startValue, 138
- OALPropertyAction(), 138
- OALPropertyAction(Audio), 138
- OALSequentialActions, 139
 - actionIndex_, 140
 - actions, 141
 - actions:, 140
 - actionsFromArray:, 140
 - initWithActions:, 140
 - pCurrentActionComplete_, 141
 - pCurrentActionDuration_, 141
 - pLastComplete_, 141
- OALSimpleAudio, 141
 - allowIpod, 149
 - backgroundTrack, 149
 - backgroundTrackURL, 149
 - bgMuted, 150
 - bgPaused, 150
 - bgPlaying, 150
 - bgVolume, 150
 - channel, 150
 - context, 150
 - device, 150
 - effectsMuted, 150
 - effectsPaused, 150
 - effectsVolume, 150
 - honorSilentSwitch, 150
 - interrupted, 151
 - manuallySuspended, 151
 - muted, 151
 - paused, 151
 - pendingLoadCount, 149
 - playBg, 144
 - playBg:, 144
 - playBg:loop:, 144
 - playBg:volume:pan:loop:, 145
 - playBgWithLoop:, 145
 - playBuffer:volume:pitch:pan:loop:, 145
 - playEffect:, 145
 - playEffect:loop:, 146
 - playEffect:volume:pitch:pan:loop:, 146
 - preloadBg:, 146
 - preloadBg:seekTime:, 147
 - preloadCache, 149
 - preloadCacheCount, 151
 - preloadCacheEnabled, 151
 - preloadEffect:, 147
 - preloadEffect:reduceToMono:, 147
 - reservedSources, 151
 - resetToDefault, 147
 - sharedInstanceWithReservedSources:mono-Sources:stereoSources:, 147
 - sharedInstanceWithSources:, 148
 - stopAllEffects, 148
 - stopBg, 148
 - stopEverything, 148
 - suspended, 151
 - unloadAllEffects, 148
 - unloadEffect:, 149
 - useHardwareIfAvailable, 151
- OALSuspendHandler, 152
 - addSuspendListener:, 153
 - handlerWithTarget:selector:, 153
 - initWithTarget:selector:, 153
 - interruptLock, 154
 - interrupted, 154

- listeners, 154
- manualSuspendLock, 154
- manualSuspendStates, 154
- manuallySuspended, 154
- removeSuspendListener:, 154
- suspendStatusChangeSelector, 154
- suspended, 155
- OALSuspendListener-p
 - interrupted, 156
 - manuallySuspended, 156
- OALSuspendManager-p
 - addSuspendListener:, 157
 - removeSuspendListener:, 157
 - suspended, 158
- OALTargetedAction, 158
 - action_, 159
 - actionWithTarget:action:, 159
 - forcedTarget, 159
 - initWithTarget:action:, 159
- OALTools, 159
 - defaultBundle, 160
 - notifyAudioSessionError:function:description:, 160
 - notifyExtAudioError:function:description:, 160
 - setDefaultBundle:, 161
 - urlForPath:, 161
 - urlForPath:bundle:, 161
- object1_
 - OALCallAction, 126
- object2_
 - OALCallAction, 126
- offsetInBytes
 - ALSource, 57
- offsetInSamples
 - ALSource, 57
- offsetInSeconds
 - ALSource, 57
- OpenALManager, 162
 - availableCaptureDevices, 166
 - availableDevices, 166
 - bufferAsyncFromFile:reduceToMono:target:selector-:, 163
 - bufferAsyncFromFile:target:selector:, 164
 - bufferAsyncFromUrl:reduceToMono:target:selector-:, 164
 - bufferAsyncFromUrl:target:selector:, 164
 - bufferFromFile:, 165
 - bufferFromFile:reduceToMono:, 165
 - bufferFromUrl:, 165
 - bufferFromUrl:reduceToMono:, 166
 - clearAllBuffers, 166
 - currentContext, 167
 - defaultCaptureDeviceSpecifier, 167
 - defaultDeviceSpecifier, 167
 - devices, 166, 167
 - mixerOutputFrequency, 167
 - operationQueue, 166
 - renderingQuality, 167
 - suspendHandler, 166
 - openCaptureDevice:frequency:format:bufferSize:
 - ALWrapper, 86
 - openDevice:
 - ALWrapper, 87
 - operationQueue
 - OALAudioTrack, 119
 - OpenALManager, 166
 - orientation
 - ALListener, 40
 - originalChannelsPerFrame
 - OALAudioFile, 104
 - pCurrentActionComplete_
 - OALSequentialActions, 141
 - pCurrentActionDuration_
 - OALSequentialActions, 141
 - pLastComplete_
 - OALSequentialActions, 141
 - pan
 - ALSoundSource-p, 47
 - OALAudioTrack, 120
 - panAction
 - ALSource, 56
 - OALAudioTrack, 119
 - panCompleteSelector
 - ALChannelSource, 28
 - panCompleteTarget
 - ALChannelSource, 29
 - panTo:duration:target:selector:
 - ALSoundSource-p, 45
 - OALAudioTrack, 112
 - parentBuffer
 - ALBuffer, 17
 - paused
 - ALSoundSource-p, 48
 - OALAudioTrack, 121
 - OALAudioTracks, 123
 - OALSimpleAudio, 151
 - peakPowerForChannel:
 - OALAudioTrack, 113
 - pendingLoadCount
 - OALSimpleAudio, 149
 - pitch
 - ALSoundSource-p, 48
 - pitchAction
 - ALSource, 56
 - pitchCompleteSelector
 - ALChannelSource, 29
 - pitchCompleteTarget
 - ALChannelSource, 29
 - pitchTo:duration:target:selector:
 - ALSoundSource-p, 45
 - play
 - ALSource, 53
 - OALAudioTrack, 113
 - play:
 - ALSoundSource-p, 45
 - play:gain:pitch:pan:loop:
 - ALSoundSource-p, 45

- play:loop:
 - ALSoundSource-p, [46](#)
- playAfterTrack:
 - OALAudioTrack, [113](#)
- playAfterTrack:timeAdjust:
 - OALAudioTrack, [113](#)
- playAtTime:
 - OALAudioTrack, [114](#)
- playBg
 - OALSimpleAudio, [144](#)
- playBg:
 - OALSimpleAudio, [144](#)
- playBg:loop:
 - OALSimpleAudio, [144](#)
- playBg:volume:pan:loop:
 - OALSimpleAudio, [145](#)
- playBgWithLoop:
 - OALSimpleAudio, [145](#)
- playBuffer:volume:pitch:pan:loop:
 - OALSimpleAudio, [145](#)
- playEffect:
 - OALSimpleAudio, [145](#)
- playEffect:loop:
 - OALSimpleAudio, [146](#)
- playEffect:volume:pitch:pan:loop:
 - OALSimpleAudio, [146](#)
- playFile:
 - OALAudioTrack, [114](#)
- playFile:loops:
 - OALAudioTrack, [114](#)
- playFileAsync:loops:target:selector:
 - OALAudioTrack, [114](#)
- playFileAsync:target:selector:
 - OALAudioTrack, [115](#)
- playUrl:
 - OALAudioTrack, [115](#)
- playUrl:loops:
 - OALAudioTrack, [115](#)
- playUrlAsync:loops:target:selector:
 - OALAudioTrack, [115](#)
- playUrlAsync:target:selector:
 - OALAudioTrack, [115](#)
- player
 - OALAudioTrack, [121](#)
- playing
 - ALSoundSource-p, [48](#)
 - OALAudioTrack, [121](#)
- pool
 - ALSoundSourcePool, [50](#)
- position
 - ALListener, [40](#)
 - ALSoundSource-p, [48](#)
 - OALMoveToAction, [134](#)
 - OALPlaceAction, [136](#)
- preferredIOBufferDuration
 - OALAudioSession, [108](#)
- preloadBg:
 - OALSimpleAudio, [146](#)
- preloadBg:seekTime:
 - OALSimpleAudio, [147](#)
- preloadCache
 - OALSimpleAudio, [149](#)
- preloadCacheCount
 - OALSimpleAudio, [151](#)
- preloadCacheEnabled
 - OALSimpleAudio, [151](#)
- preloadEffect:
 - OALSimpleAudio, [147](#)
- preloadEffect:reduceToMono:
 - OALSimpleAudio, [147](#)
- preloadFile:
 - OALAudioTrack, [116](#)
- preloadFile:seekTime:
 - OALAudioTrack, [116](#)
- preloadFileAsync:seekTime:target:selector:
 - OALAudioTrack, [116](#)
- preloadFileAsync:target:selector:
 - OALAudioTrack, [117](#)
- preloadUrl:
 - OALAudioTrack, [117](#)
- preloadUrl:seekTime:
 - OALAudioTrack, [117](#)
- preloadUrlAsync:seekTime:target:selector:
 - OALAudioTrack, [117](#)
- preloadUrlAsync:target:selector:
 - OALAudioTrack, [118](#)
- preloaded
 - OALAudioTrack, [121](#)
- prepareWithTarget:
 - OALAction, [99](#)
- process
 - ALContext, [33](#)
- processContext:
 - ALWrapper, [87](#)
- queueBuffer:
 - ALSource, [54](#)
- queueBuffer:repeats:
 - ALSource, [54](#)
- queueBuffers:
 - ALSource, [54](#)
- queueBuffers:repeats:
 - ALSource, [54](#)
- reduceToMono
 - OALAudioFile, [105](#)
- referenceDistance
 - ALSoundSource-p, [48](#)
- registerNotification:callback:userData:
 - ALSource, [55](#)
- removeBuffersNamed:
 - ALChannelSource, [25](#)
- removeNotification:onSource:callback:userData:
 - ALWrapper, [87](#)
- removeSource:
 - ALChannelSource, [25](#)
 - ALSoundSourcePool, [50](#)

- removeSuspendListener:
 - OALSuspendHandler, [154](#)
 - OALSuspendManager-p, [157](#)
- renderer
 - ALContext, [35](#)
- renderingQuality
 - OpenALManager, [167](#)
- reservedSources
 - ALChannelSource, [29](#)
 - OALSimpleAudio, [151](#)
- resetToDefault
 - ALChannelSource, [26](#)
 - OALSimpleAudio, [147](#)
- reverbEQBandwidth
 - ALListener, [40](#)
- reverbEQFrequency
 - ALListener, [40](#)
- reverbEQGain
 - ALListener, [40](#)
- reverbObstruction
 - ALSoundSource-p, [48](#)
- reverbOcclusion
 - ALSoundSource-p, [48](#)
- reverbOn
 - ALListener, [40](#)
- reverbRoomType
 - ALListener, [40](#)
- reverbSendLevel
 - ALSoundSource-p, [48](#)
- rewind
 - ALSoundSource-p, [46](#)
- rolloffFactor
 - ALSoundSource-p, [48](#)
- runWithTarget:
 - OALAction, [99](#)
- running
 - OALAction, [100](#)
- runningInManager_
 - OALAction, [100](#)
- selector_
 - OALCallAction, [126](#)
- setDefaultBundle:
 - OALTools, [161](#)
- setDefaultFromSource:
 - ALChannelSource, [26](#)
- setMixerOutputDataRate:
 - ALWrapper, [87](#)
- setRenderingQuality:
 - ALWrapper, [87](#)
- shadowState
 - ALSource, [56](#)
- sharedInstanceWithReservedSources:monoSources-:stereoSources:
 - OALSimpleAudio, [147](#)
- sharedInstanceWithSources:
 - OALSimpleAudio, [148](#)
- simulatorPlayerRef
 - OALAudioTrack, [119](#)
- size
 - ALBuffer, [18](#)
- sliceWithName:offset:size:
 - ALBuffer, [17](#)
- source
 - ALSource, [55](#)
- source3f:parameter:v1:v2:v3:
 - ALWrapper, [88](#)
- source3i:parameter:v1:v2:v3:
 - ALWrapper, [88](#)
- sourceceld
 - ALSource, [57](#)
- sourceOnContext:
 - ALSource, [55](#)
- sourcePause:
 - ALWrapper, [89](#)
- sourcePausev:numSources:
 - ALWrapper, [90](#)
- sourcePlay:
 - ALWrapper, [90](#)
- sourcePlayv:numSources:
 - ALWrapper, [90](#)
- sourcePool
 - ALChannelSource, [29](#)
- sourceQueueBuffers:numBuffers:bufferIds:
 - ALWrapper, [90](#)
- sourceRelative
 - ALSoundSource-p, [48](#)
- sourceRewind:
 - ALWrapper, [91](#)
- sourceRewindv:numSources:
 - ALWrapper, [91](#)
- sourceStop:
 - ALWrapper, [91](#)
- sourceStopv:numSources:
 - ALWrapper, [91](#)
- sourceType
 - ALSoundSource-p, [48](#)
- sourceUnqueueBuffers:numBuffers:bufferIds:
 - ALWrapper, [91](#)
- sourcef:parameter:value:
 - ALWrapper, [88](#)
- sourcefv:parameter:values:
 - ALWrapper, [89](#)
- sourcei:parameter:value:
 - ALWrapper, [89](#)
- sourceiv:parameter:values:
 - ALWrapper, [89](#)
- sources
 - ALContext, [33](#), [35](#)
 - ALSoundSourcePool, [50](#), [51](#)
- speedOfSound
 - ALContext, [35](#)
- speedOfSound:
 - ALWrapper, [92](#)
- splitChannelWithSources:
 - ALChannelSource, [26](#)
- startAction

- OALAction, 99
- startCapture
 - ALCaptureDevice, 21
- startCapture:
 - ALWrapper, 92
- startPoint
 - OALMoveByAction, 132
 - OALMoveToAction, 134
- startValue
 - OALPropertyAction, 138
- state
 - ALSource, 57
- stepTimer
 - OALActionManager, 102
- stop
 - ALSoundSource-p, 46
 - OALAudioTrack, 118
- stopAction
 - OALAction, 99
- stopActions
 - ALSoundSource-p, 46
 - OALAudioTrack, 118
- stopAllActions
 - OALActionManager, 101
- stopAllEffects
 - OALSimpleAudio, 148
- stopAllSounds
 - ALContext, 33
- stopAllTracks
 - OALAudioTracks, 122
- stopBg
 - OALSimpleAudio, 148
- stopCapture
 - ALCaptureDevice, 21
- stopCapture:
 - ALWrapper, 92
- stopEverything
 - OALSimpleAudio, 148
- stopFade
 - ALSoundSource-p, 46
 - OALAudioTrack, 118
- stopPan
 - ALSoundSource-p, 46
 - OALAudioTrack, 118
- stopPitch
 - ALSoundSource-p, 46
- streamDescription
 - OALAudioFile, 105
- suspendContext:
 - ALWrapper, 92
- suspendHandler
 - ALContext, 34
 - ALDevice, 38
 - ALListener, 39
 - ALSource, 56
 - OALAudioSession, 107
 - OALAudioTrack, 119
 - OALAudioTracks, 122
 - OpenALManager, 166
 - suspendStatusChangeSelector
 - OALSuspendHandler, 154
 - suspended
 - OALSimpleAudio, 151
 - OALSuspendHandler, 155
 - OALSuspendManager-p, 158
 - target
 - OALAction, 100
 - targetActions
 - OALActionManager, 102
 - targets
 - OALActionManager, 102
 - totalFrames
 - OALAudioFile, 105
 - track
 - OALAudioTrack, 118
 - tracks
 - OALAudioTracks, 122, 123
 - unitsPerSecond
 - OALMoveByAction, 132
 - OALMoveToAction, 134
 - unloadAllEffects
 - OALSimpleAudio, 148
 - unloadEffect:
 - OALSimpleAudio, 149
 - unqueueBuffer:
 - ALSource, 55
 - unqueueBuffers:
 - ALSource, 55
 - unregisterAllNotifications
 - ALSource, 56
 - unregisterNotification:
 - ALSource, 56
 - up
 - ALOrientation, 41
 - updateCompletion:
 - OALAction, 99
 - updateMeters
 - OALAudioTrack, 118
 - url
 - OALAudioFile, 105
 - urlForPath:
 - OALTools, 161
 - urlForPath:bundle:
 - OALTools, 161
 - useHardwareIfAvailable
 - OALAudioSession, 109
 - OALSimpleAudio, 151
 - velocity
 - ALListener, 41
 - ALSoundSource-p, 49
 - vendor
 - ALContext, 35
 - version
 - IOSVersion, 95

volume

ALSoundSource-p, [49](#)

OALAudioTrack, [121](#)

x

ALPoint, [42](#)

ALVector, [58](#)

y

ALPoint, [42](#)

ALVector, [58](#)

z

ALPoint, [42](#)

ALVector, [58](#)