OpenAL Programmer's Guide

OpenAL Versions 1.0 and 1.1

CREATIVE®

Trademarks and Service Marks

Creative, Sound Blaster, and the Creative logo are registered trademarks, and Environmental Audio, EAX, and the Environmental Audio Extensions logo are trademarks of Creative Technology Ltd. in the United States and/or other countries.

All other brands and product names listed are trademarks or registered trademarks of their respective holders.

Acknowledgments

Documentation written by Garin Hiebert. Additional input by Keith Charley, Jean-Marc Jot, Daniel Peacock, Jean-Michel Trivi, and Carlo Vogelsang.

Revision History

Revision 1.0	October 2005	Garin Hiebert

Table of Contents

About this Document

Introduction

OpenAL is a cross-platform three-dimensional audio API. The API's primary purpose is to allow an application to position audio sources in a three-dimensional space around a listener, producing reasonable spatialization of the sources for the audio system (headphones, 2.1 speaker output, 5.1 speaker output, etc.) Through extensions, Creative Labs has also enhanced OpenAL with EAX and other capabilities. OpenAL is appropriate for many audio applications, but was designed to be most appropriate for gaming audio.

Intended Audience

This reference guide is most appropriate for a programmer. Experience with C or C++ is not required to learn the concepts in OpenAL, but will make understanding the OpenAL source as well as sample code easier. Since there are several sample applications included with the OpenAL SDKs as well as with the source distribution, it is recommended that interested programmers take advantage of those resources.

Other OpenAL Resources

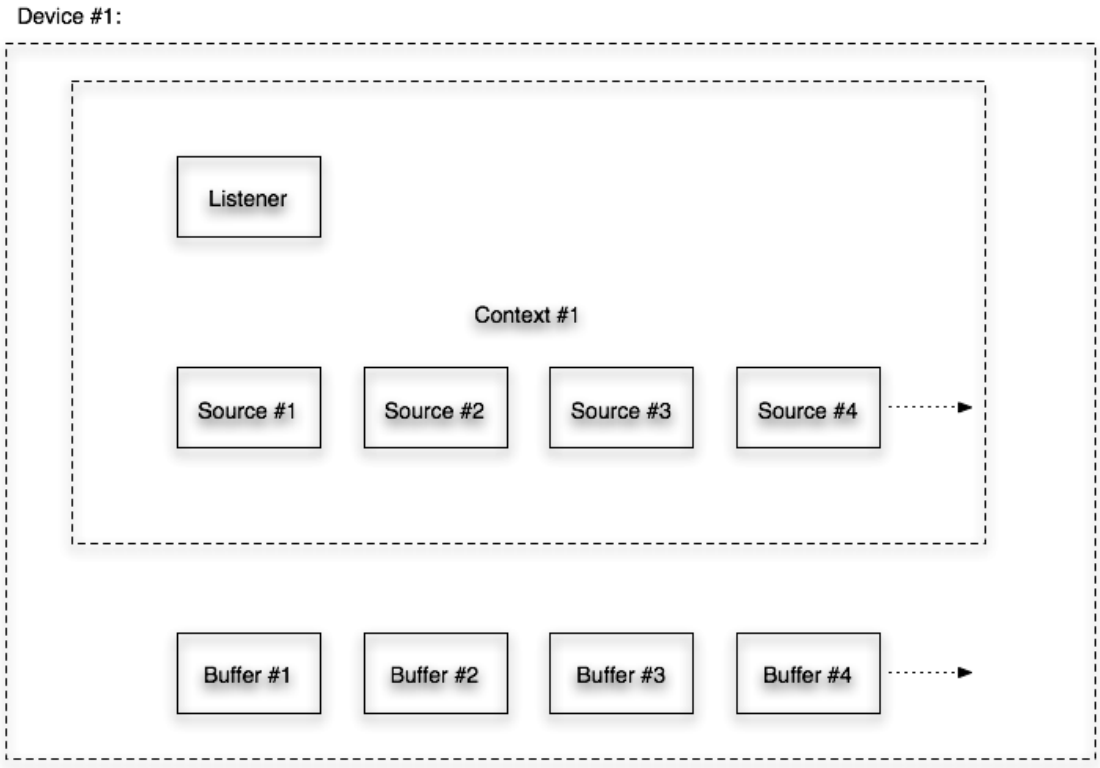
The two most important resources for additional information on OpenAL are the websites at *www.openal.org* and *http://developer.creative.com.* The main OpenAL site hosts the specification, the open source implementations, and sample code. The Creative developer site has a section dedicated to OpenAL with SDKs showing how to use OpenAL as well as various extensions.

Introduction to OpenAL

Use of OpenAL revolves around the use of three fundamental objects – Buffers, Sources, and a Listener. A buffer can be filled with audio data, and can then be attached to a source. The source can then be positioned and played. How the source is heard is determined by its position and orientation relative to the Listener object (there is only one Listener). Creating a number of sources and buffers and a single listener and then updating the positions and orientations of the sources and listener dynamically can present a convincing 3D audio world.

Objects

Here is a diagram showing the fundamental OpenAL objects and their relationships to the context and device objects:



When initializing OpenAL, at least one device has to be opened. Within that device, at least one context will be created. Within that context, one listener object is implied, and a multitude of source objects can be created. Each source can have one or more buffers objects attached to it. Buffer objects are not part of a specific context – they are shared among all contexts on one device.

Device Enumeration

How does an application know what devices are available to open? An OpenAL application can open a "default" device if it wishes, by specifying NULL for the device string when opening a device. An application that wants more control over which device is opened – or that wants to present the user with optional devices to open – can also enumerate the names of the available input and output devices (and explore some capabilities of each device as well).

Device enumeration revolves around the *alcGetString* call. If an application calls *alcGetString* with a NULL device specified and asks for ALC_DEVICE_SPECIFIER, a pointer is returned to a string which is actually a *list* of devices separated by NULL characters and terminated by two NULL characters. If an application calls *alcGetString* and asks for ALC_DEFAULT_DEVICE_SPECIFIER, then the string returned will be the default output device for the system. Retrieving the device list and the default device name allows an application to present the end user with a menu of available output devices.

Additional filtering can be done on the returned list by asking each device what specification version it supports as well as what extensions it supports (using *alcIsExtensionPresent* or *alIsExtensionPresent*).

Initializing/Exiting

As described above, the first step to initializing OpenAL is to open a device. Once that is successfully done, then a context is opened on that device. Now the fundamental OpenAL objects can be managed – the listener, various sources, and various buffers.

To generate a set of buffers for use, use *alGetError* to reset the error state, call *alGenBuffers* to generate the number of buffers desired, and then use *alGetError* again to detect if an error was generated.

Fill the buffers with PCM data using *alBufferData*.

To generate a set of sources for use, use *alGetError* to reset the error state, call *alGenSources* to generate the number of sources desired, and then use *alGetError* again to detect if an error was generated.

Buffers are attached to sources using *alSourcei*.

Once a buffer has been attached to a source, the source can play the buffer using *alSourcePlay*.

Source and Listener properties can be updated dynamically using property set and get calls such as *alGetListenerfv, alListener3f, alSourcei,* and *alGetSource3f*.

Example:

```
// INIT
Device = alcOpenDevice(NULL); // select the "preferred device"

if (Device) {
        Context=alcCreateContext(Device,NULL);
        alcMakeContextCurrent(Context);
}

// Check for EAX 2.0 support
g_bEAX = alIsExtensionPresent("EAX2.0");

// Generate Buffers
alGetError(); // clear error code
alGenBuffers(NUM_BUFFERS, g_Buffers);
if ((error = alGetError()) != AL_NO_ERROR)
{
        DisplayALError("alGenBuffers :", error);
        return;
}

// Load test.wav
loadWAVFile("test.wav",&format,&data,&size,&freq,&loop);
if ((error = alGetError()) != AL_NO_ERROR)
{
        DisplayALError("alutLoadWAVFile test.wav : ", error);
        alDeleteBuffers(NUM_BUFFERS, g_Buffers);
        return;
}

// Copy test.wav data into AL Buffer 0
alBufferData(g_Buffers[0],format,data,size,freq);
if ((error = alGetError()) != AL_NO_ERROR)
{
        DisplayALError("alBufferData buffer 0 : ", error);
```

```
        alDeleteBuffers(NUM_BUFFERS, g_Buffers);
        return;
}

// Unload test.wav
unloadWAV(format,data,size,freq);
if ((error = alGetError()) != AL_NO_ERROR)
{
        DisplayALError("alutUnloadWAV : ", error);
        alDeleteBuffers(NUM_BUFFERS, g_Buffers);
        return;
}


// Generate Sources
alGenSources(1,source);
if ((error = alGetError()) != AL_NO_ERROR)
{
        DisplayALError("alGenSources 1 : ", error);
        return;
}

// Attach buffer 0 to source
alSourcei(source[0], AL_BUFFER, g_Buffers[0]);
if ((error = alGetError()) != AL_NO_ERROR)
{
        DisplayALError("alSourcei AL_BUFFER 0 : ", error);
}

// EXIT
Context=alcGetCurrentContext();
Device=alcGetContextsDevice(Context);
alcMakeContextCurrent(NULL);
alcDestroyContext(Context);
alcCloseDevice(Device);
```

Listener Properties

For every context, there is automatically one Listener object. The *alListener[f, 3f, fv, i]* and *alGetListener[f, 3f, fv, i]* families of functions can be used to set or retrieve the following listener properties:

Property	Data Type	Description
AL_GAIN	f, fv	“master gain” value should be positive
AL_POSITION	fv, 3f, iv, 3i	X, Y, Z position
AL_VELOCITY	fv, 3f, iv, 3i	velocity vector
AL_ORIENTATION	fv, iv	orientation expressed as “at” and “up” vectors

Example:

```
ALfloat listenerPos[]={0.0,0.0,0.0};
ALfloat listenerVel[]={0.0,0.0,0.0};
ALfloat listenerOri[]={0.0,0.0,-1.0, 0.0,1.0,0.0}; // “at”, then “up”

// Set Listener attributes

// Position ...
alListenerfv(AL_POSITION,listenerPos);
if ((error = alGetError()) != AL_NO_ERROR)
{
        DisplayALError("alListenerfv POSITION : ", error);
        return;
}

// Velocity ...
alListenerfv(AL_VELOCITY,listenerVel);
if ((error = alGetError()) != AL_NO_ERROR)
{
        DisplayALError("alListenerfv VELOCITY : ", error);
        return;
}

// Orientation ...
alListenerfv(AL_ORIENTATION,listenerOri);
```

```
if ((error = alGetError()) != AL_NO_ERROR)
{
        DisplayALError("alListenerfv ORIENTATION : ", error);
        return;
}
```

Buffer Properties

Each buffer generated by *alGenBuffers* has properties which can be retrieved. The *alGetBuffer[f, i]* functions can be used to retrieve the following buffer properties:

Property	Data Type	Description
AL_ FREQUENCY	i, iv	frequency of buffer in Hz
AL_ BITS	i, iv	bit depth of buffer
AL_ CHANNELS	i, iv	number of channels in buffer > 1 is valid, but buffer won't be positioned when played
AL_ SIZE	i, iv	size of buffer in bytes
AL_DATA	i, iv	original location where data was copied from generally useless, as was probably freed after buffer creation

Example:

```
// Retrieve Buffer Frequency
alBufferi(g_Buffers[0], AL_FREQUENCY, iFreq);
```

Source Properties

Each source generated by *alGenSources* has properties which can be set or retrieved. The *alSource[f, 3f, fv, i]* and *alGetSource[f, 3f, fv, i]* families of functions can be used to set or retrieve the following source properties:

Property	Data Type	Description
AL_PITCH	f, fv	pitch multiplier always positive
AL_GAIN	f, fv	source gain value should be positive
AL_MAX_DISTANCE	f, fv, i, iv	used with the Inverse Clamped Distance Model to set the distance where there will no longer be any attenuation of the source
AL_ROLLOFF_FACTOR	f, fv, i, iv	the rolloff rate for the source default is 1.0
AL_REFERENCE_DISTANCE	f, fv, i, iv	the distance under which the volume for the source would normally drop by half (before being influenced by rolloff factor or AL_MAX_DISTANCE)
AL_MIN_GAIN	f, fv	the minimum gain for this source
AL_MAX_GAIN	f, fv	the maximum gain for this source
AL_CONE_OUTER_GAIN	f, fv	the gain when outside the oriented cone
AL_CONE_INNER_ANGLE	f, fv, i, iv	the gain when inside the oriented cone
AL_CONE_OUTER_ANGLE	f, fv, i, iv	outer angle of the sound cone, in degrees default is 360
AL_POSITION	fv, 3f	X, Y, Z position
AL_VELOCITY	fv, 3f	velocity vector
AL_DIRECTION	fv, 3f, iv, 3i	direction vector
AL_SOURCE_RELATIVE	i, iv	determines if the positions are relative to the listener default is AL_FALSE
AL_SOURCE_TYPE	i, iv	the soruce type – AL_UNDETERMINED, AL_STATIC, or AL_STREAMING
AL_LOOPING	i, iv	turns looping on (AL_TRUE) or off (AL_FALSE)
AL_BUFFER	i, iv	the ID of the attached buffer
AL_SOURCE_STATE	i, iv	the state of the source (AL_STOPPED, AL_PLAYING, …)
AL_BUFFERS_QUEUED*	i, iv	the number of buffers queued on this source
AL_BUFFERS_PROCESSED*	i, iv	the number of buffers in the queue that have been processed

AL_SEC_OFFSET	f, fv, i, iv	the playback position, expressed in seconds
AL_SAMPLE_OFFSET	f, fv, i, iv	the playback position, expressed in samples
AL_BYTE_OFFSET	f, fv, i, iv	the playback position, expressed in bytes

** Read Only (alGetSourcei)*

Example:

```
alGetError(); // clear error state
alSourcef(source[0],AL_PITCH,1.0f);
if ((error = alGetError()) != AL_NO_ERROR)
        DisplayALError("alSourcef 0 AL_PITCH : \n", error);

alGetError(); // clear error state
alSourcef(source[0],AL_GAIN,1.0f);
if ((error = alGetError()) != AL_NO_ERROR)
        DisplayALError("alSourcef 0 AL_GAIN : \n", error);

alGetError(); // clear error state
alSourcefv(source[0],AL_POSITION,source0Pos);
if ((error = alGetError()) != AL_NO_ERROR)
        DisplayALError("alSourcefv 0 AL_POSITION : \n", error);

alGetError(); // clear error state
alSourcefv(source[0],AL_VELOCITY,source0Vel);
if ((error = alGetError()) != AL_NO_ERROR)
        DisplayALError("alSourcefv 0 AL_VELOCITY : \n", error);

alGetError(); // clear error state
alSourcei(source[0],AL_LOOPING,AL_FALSE);
if ((error = alGetError()) != AL_NO_ERROR)
        DisplayALError("alSourcei 0 AL_LOOPING true: \n", error);
```

Queuing Buffers on a Source

To continuously stream audio from a source without interruption, buffer queuing is required. To use buffer queuing, the buffers and sources are generated in the normal way, but *alSourcei* is not used to attach the buffers to the source. Instead, the functions *alSourceQueueBuffers* and *alSourceUnqueueBuffers* are used. The program can attach a buffer or a set of buffers to a source using *alSourceQueueBuffers*, and then call alSourcePlay on that source. While the source is playing, *alSourceUnqueueBuffers* can be called to remove buffers which have already played. Those buffers can then be filled with new data or discarded. New or refilled buffers can then be attached to the playing source using *alSourceQueueBuffers*. As long as there is always a new buffer to play in the queue, the source will continue to play.

Although some 1.0 implementations of OpenAL may not enforce the following restrictions on queuing, it is recommended to observe the following additional rules, which do universally apply to 1.1 implementations:

1) A source that will be used for streaming should not have its first buffer attached using *alSourcei* – always use *alSourceQueueBuffers* to attach buffers to streaming sources. Any source can have all buffers detached from it using *alSourcei*(..., AL_BUFFER, 0), and can then be used for either streaming or non-streaming buffers depending on how data is then attached to the source (with *alSourcei* or with *alSourceQueueBuffers*).

2) All buffers attached to a source using *alSourceQueueBuffers* should have the same audio format.

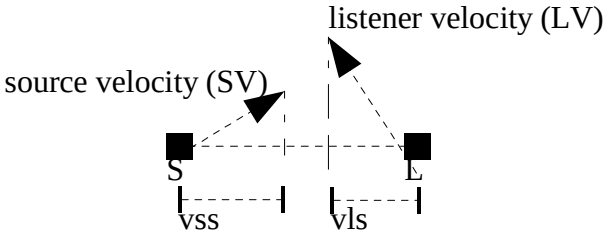
Doppler Shift

The doppler effect depends on the velocities of source and listener relative to the medium, and the propagation speed of sound in that medium. The application might want to emphasize or de-emphasize the doppler effect as physically accurate calculation might not give the desired results. The amount of frequency shift (pitch change) is proportional to the speed of listener and source along their line of sight.

The doppler effect as implemented by OpenAL is described by the formula below. Effects of the medium (air, water) moving with respect to listener and source are ignored.

SS: AL_SPEED_OF_SOUND = speed of sound (default value 343.3)
DF: AL_DOPPLER_FACTOR = Doppler factor (default 1.0)
vls: Listener velocity scalar (scalar, projected on source-to-listener vector)
vss: Source velocity scalar (scalar, projected on source-to-listener vector)
f: Frequency of sample
f': effective Doppler shifted frequency

Graphic representation of vls and vss:



3D Mathematical representation of vls and vss:

Mag(vector) = sqrt(vector.x * vector.x + vector.y * vector.y + vector.z * vector.z)
DotProduct(v1, v2) = (v1.x * v2.x + v1.y * v2.y + v1.z * v2.z)
SL = source to listener vector
SV = Source Velocity vector
LV = Listener Velocity vector

vls = DotProduct(SL, LV) / Mag(SL)
vss = DotProduct(SL, SV) / Mag(SL)

Dopper Calculation:

vss = min(vss, SS/DF)
vls = min(vls, SS/DF)
f' = f * (SS - DF*vls) / (SS - DF*vss)

There are two API calls global to the current context that provide control of the speed of sound and Doppler factor. AL_DOPPLER_FACTOR is a simple scaling of source and listener velocities to exaggerate or deemphasize the Doppler (pitch) shift resulting from the calculation.

void alDopplerFactor(ALfloat dopplerFactor);

A negative value will result in an AL_INVALID_VALUE error, the command is then ignored. The default value is 1. The current setting can be queried using *alGetFloat{v}* and AL_DOPPLER_FACTOR.

AL_SPEED_OF_SOUND allows the application to change the reference (propagation) speed used in the Doppler calculation. The source and listener velocities should be expressed in the same units as the speed of sound.

void alSpeedOfSound(ALfloat speed);

A negative or zero value will result in an AL_INVALID_VALUE error, and the command is ignored. The default value is 343.3 (appropriate for velocity units of meters and air as the propagation medium). The current setting can be queried using *alGetFloat{v}* and AL_SPEED_OF_SOUND.

Distance and velocity units are completely independent of one another (so you could use different units for each if desired). If an OpenAL application doesn't want to use Doppler effects, then leaving all velocities at zero will achieve that result.

Error Handling

The error state of OpenAL can be retrieved at any time using *alGetError*. *alGetError* clears the error state of OpenAL when it is called, so it is common for an OpenAL application to call *alGetError* at the beginning of a critical operation to clear the error state, perform the critical operation, and then use *alGetError* again to test whether or not an error occurred.

Error Codes:

Error Code	Description
AL_NO_ERROR	there is not currently an error
AL_INVALID_NAME	a bad name (ID) was passed to an OpenAL function
AL_INVALID_ENUM	an invalid enum value was passed to an OpenAL function
AL_INVALID_VALUE	an invalid value was passed to an OpenAL function
AL_INVALID_OPERATION	the requested operation is not valid
AL_OUT_OF_MEMORY	the requested operation resulted in OpenAL running out of memory

Example:

```
alGetError();  // Clear Error Code

// Generate Buffers
alGenBuffers(NUM_BUFFERS, g_Buffers);
if ((error = alGetError()) != AL_NO_ERROR)
{
	DisplayALError("alGenBuffers :", error);
	exit(-1);
}
```

Core OpenAL Functions

Buffer-Related

alGenBuffers

Description

This function generates one or more buffers, which contain audio data (see *alBufferData*). References to buffers are ALuint values, which are used wherever a buffer reference is needed (in calls such as *alDeleteBuffers*, *alSourcei*, *alSourceQueueBuffers*, and *alSourceUnqueueBuffers*).

```
void alGenBuffers(
        ALsizei n,
        ALuint *buffers
);
```

Parameters

n	the number of buffers to be generated
buffers	pointer to an array of ALuint values which will store the names of the new buffers

Possible Error States

State	***Description***
AL_INVALID_VALUE	The buffer array isn't large enough to hold the number of buffers requested.
AL_OUT_OF_MEMORY	There is not enough memory available to generate all the buffers requested.

Version Requirements

OpenAL 1.0 or higher

Remarks

If the requested number of buffers cannot be created, an error will be generated which can be detected with alGetError. If an error occurs, no buffers will be generated. If n equals zero, alGenBuffers does nothing and does not return an error.

alDeleteBuffers

Description

This function deletes one or more buffers, freeing the resources used by the buffer. Buffers which are attached to a source can not be deleted. See *alSourcei* and *alSourceUnqueueBuffers* for information on how to detach a buffer from a source.

```
void alDeleteBuffers(
        ALsizei n,
        ALuint *buffers
);
```

Parameters

n the number of buffers to be deleted

buffers pointer to an array of buffer names identifying the buffers to be deleted

Possible Error States

State	***Description***
AL_INVALID_OPERATION	The buffer is still in use and can not be deleted.
AL_INVALID_NAME	A buffer name is invalid.
AL_INVALID_VALUE	The requested number of buffers can not be deleted.

Version Requirements

OpenAL 1.0 or higher

Remarks

If the requested number of buffers cannot be deleted, an error will be generated which can be detected with alGetError. If an error occurs, no buffers will be deleted. If n equals zero, alDeleteBuffers does nothing and will not return an error.

alIsBuffer

Description

This function tests if a buffer name is valid, returning AL_TRUE if valid, AL_FALSE if not.

```
ALboolean alIsBuffer(
        ALuint buffer
);
```

Parameters

buffer	a buffer name to be tested for validity

Possible Error States

None

Version Requirements

OpenAL 1.0 or higher

Remarks

The NULL buffer is always valid (see *alSourcei* for information on how the NULL buffer is used).

alBufferData

Description

This function fills a buffer with audio data. All the pre-defined formats are PCM data, but this function may be used by extensions to load other data types as well.

```
void alBufferData(
        ALuint buffer,
        ALenum format,
        const ALvoid *data,
        ALsizei size,
        ALsizei freq
);
```

Parameters:

buffer	buffer name to be filled with data
format	format type from among the following: AL_FORMAT_MONO8 AL_FORMAT_MONO16 AL_FORMAT_STEREO8 AL_FORMAT_STEREO16
data	pointer to the audio data
size	the size of the audio data in bytes
freq	the frequency of the audio data

Possible Error States

State	***Description***
AL_OUT_OF_MEMORY	There is not enough memory available to create this buffer.
AL_INVALID_VALUE	The size parameter is not valid for the format specified, the buffer is in use, or the data is a NULL pointer.
AL_INVALID_ENUM	The specified format does not exist.

Version Requirements

OpenAL 1.0 or higher

Remarks

8-bit PCM data is expressed as an unsigned value over the range 0 to 255, 128 being an audio output level of zero. 16-bit PCM data is expressed as a signed value over the range -32768 to 32767, 0 being an audio output level of zero. Stereo data is expressed in interleaved format, left channel first. Buffers containing more than one channel of data will be played without 3D spatialization.

alBufferf

Description

This function sets a floating point property of a buffer.

```
void alBufferf(
        ALuint buffer,
        ALenum param,
        ALfloat value
);
```

Parameters

buffer	buffer name whose attribute is being retrieved
param	the name of the attribute to be set
value	the ALfloat value to be set

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified buffer doesn't have parameters (the NULL buffer), or doesn't exist.

Version Requirements

OpenAL 1.1 or higher

Remarks

There are no relevant buffer properties defined in OpenAL 1.1 which can be affected by this call, but this function may be used by OpenAL extensions.

alBuffer3f

Description

This function sets a floating point property of a buffer.

```
void alBuffer3f(
        ALuint buffer,
        ALenum param,
        ALfloat v1,
        ALfloat v2,
        ALfloat v3
);
```

Parameters

buffer	buffer name whose attribute is being retrieved
param	the name of the attribute to be set
v1, v2, v3	the ALfloat values to be set

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified buffer doesn't have parameters (the NULL buffer), or doesn't exist.

Version Requirements

OpenAL 1.1 or higher

Remarks

There are no relevant buffer properties defined in OpenAL 1.1 which can be affected by this call, but this function may be used by OpenAL extensions.

alBufferfv

Description

This function sets a floating point property of a buffer.

```
void alBufferfv(
        ALuint buffer,
        ALenum param,
        ALfloat *values
);
```

Parameters

buffer	buffer name whose attribute is being retrieved
param	the name of the attribute to be set
values	a pointer to the ALfloat values to be set

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified buffer doesn't have parameters (the NULL buffer), or doesn't exist.

Version Requirements

OpenAL 1.1 or higher

Remarks

There are no relevant buffer properties defined in OpenAL 1.1 which can be affected by this call, but this function may be used by OpenAL extensions.

alBufferi

Description

This function retrieves an integer property of a buffer.

```
void alBufferi(
        ALuint buffer,
        ALenum param,
        ALint value
);
```

Parameters

buffer	buffer name whose attribute is being retrieved
param	the name of the attribute to be set
value	a pointer to an ALint to hold the retrieved data

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified buffer doesn't have parameters (the NULL buffer), or doesn't exist.

Version Requirements

OpenAL 1.1 or higher

Remarks

There are no relevant buffer properties defined in OpenAL 1.1 which can be affected by this call, but this function may be used by OpenAL extensions.v

alBuffer3i

Description

This function sets a floating point property of a buffer.

```
void alBuffer3i(
        ALuint buffer,
        ALenum param,
        ALint v1,
        ALint v2,
        ALint v3
);
```

Parameters

buffer buffer name whose attribute is being retrieved

param the name of the attribute to be set

v1, v2, v3 the ALint values to be set

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified buffer doesn't have parameters (the NULL buffer), or doesn't exist.

Version Requirements

OpenAL 1.1 or higher

Remarks

There are no relevant buffer properties defined in OpenAL 1.1 which can be affected by this call, but this function may be used by OpenAL extensions.

alBufferiv

Description

This function sets a floating point property of a buffer.

```
void alBufferiv(
        ALuint buffer,
        ALenum param,
        ALint *values
);
```

Parameters

buffer	buffer name whose attribute is being retrieved
param	the name of the attribute to be set
values	a pointer to the ALint values to be set

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified buffer doesn't have parameters (the NULL buffer), or doesn't exist.

Version Requirements

OpenAL 1.1 or higher

Remarks

There are no relevant buffer properties defined in OpenAL 1.1 which can be affected by this call, but this function may be used by OpenAL extensions.

alGetBufferf

Description

This function retrieves a floating point property of a buffer.

```
void alGetBufferf(
ALuint buffer,
ALenum pname,
ALfloat *value
);
```

Parameters

buffer	buffer name whose attribute is being retrieved
pname	the name of the attribute to be retrieved
value	a pointer to an ALfloat to hold the retrieved data

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified buffer doesn't have parameters (the NULL buffer), or doesn't exist.
AL_INVALID_VALUE	The specified value pointer is not valid.

Version Requirements

OpenAL 1.0 or higher

Remarks

There are no relevant buffer properties defined in OpenAL 1.1 which can be retrieved by this call, but this function may be used by OpenAL extensions.

alGetBuffer3f

Description

This function retrieves a floating point property of a buffer.

```
void alGetBuffer3f(
        ALuint buffer,
        ALenum pname,
        ALfloat *v1,
        ALfloat *v2,
        ALfloat *v3
);
```

Parameters

buffer buffer name whose attribute is being retrieved

pname the name of the attribute to be retrieved

v1, v2, v3 pointers to a ALfloat values to hold the retrieved data

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified buffer doesn't have parameters (the NULL buffer), or doesn't exist.
AL_INVALID_VALUE	The specified value pointer is not valid.

Version Requirements

OpenAL 1.1 or higher

Remarks

There are no relevant buffer properties defined in OpenAL 1.1 which can be retrieved by this call, but this function may be used by OpenAL extensions.

alGetBufferfv

Description

This function retrieves a floating point property of a buffer.

```
void alGetBufferfv(
        ALuint buffer,
        ALenum pname,
        ALfloat *values
);
```

Parameters

buffer	buffer name whose attribute is being retrieved
pname	the name of the attribute to be retrieved
values	pointer to an ALfloat vector to hold the retrieved data

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified buffer doesn't have parameters (the NULL buffer), or doesn't exist.
AL_INVALID_VALUE	The specified value pointer is not valid.

Version Requirements

OpenAL 1.1 or higher

Remarks

There are no relevant buffer properties defined in OpenAL 1.1 which can be retrieved by this call, but this function may be used by OpenAL extensions.

alGetBufferi

Description

This function retrieves an integer property of a buffer. The relevant properties are listed in the table *Buffer Properties.*

```
void alGetBufferi(
        ALuint buffer,
        ALenum pname,
        ALint *value
);
```

Parameters

buffer	buffer name whose attribute is being retrieved
pname	the name of the attribute to be retrieved: AL_FREQUENCY AL_BITS AL_CHANNELS AL_SIZE AL_DATA
value	a pointer to an ALint to hold the retrieved data

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified buffer doesn't have parameters (the NULL buffer), or doesn't exist.
AL_INVALID_VALUE	The specified value pointer is not valid.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alGetBuffer3i

Description

This function retrieves a floating point property of a buffer.

```
void alGetBuffer3i(
        ALuint buffer,
        ALenum pname,
        ALint *v1,
        ALint *v2,
        ALint *v3
);
```

Parameters

buffer buffer name whose attribute is being retrieved

pname the name of the attribute to be retrieved

v1, v2, v3 pointers to ALint values to hold the retrieved data

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified buffer doesn't have parameters (the NULL buffer), or doesn't exist.
AL_INVALID_VALUE	The specified value pointer is not valid.

Version Requirements

OpenAL 1.1 or higher

Remarks

There are no relevant buffer properties defined in OpenAL 1.1 which can be retrieved by this call, but this function may be used by OpenAL extensions.

alGetBufferiv

Description

This function retrieves a floating point property of a buffer. The relevant properties are listed in the table *Buffer Properties.*

```
void alGetBufferiv(
        ALuint buffer,
        ALenum pname,
        ALint *values
);
```

Parameters

buffer	buffer name whose attribute is being retrieved
pname	the name of the attribute to be retrieved
values	pointer to an ALint vector to hold the retrieved data

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified buffer doesn't have parameters (the NULL buffer), or doesn't exist.
AL_INVALID_VALUE	The specified value pointer is not valid.

Version Requirements

OpenAL 1.1 or higher

Remarks

None

Source-Related

alGenSources

Description:

This function generates one or more sources. References to sources are ALuint values, which are used wherever a source reference is needed (in calls such as *alDeleteSources* and *alSourcei*).

```
void alGenSources(
        ALsizei n,
        ALuint *sources
);
```

Parameters:

n	the number of sources to be generated
sources	pointer to an array of ALuint values which will store the names of the new sources

Possible Error States

State	***Description***
AL_OUT_OF_MEMORY	There is not enough memory to generate all the requested sources.
AL_INVALID_VALUE	There are not enough non-memory resources to create all the requested sources, or the array pointer is not valid.
AL_INVALID_OPERATION	There is no context to create sources in.

Version Requirements

OpenAL 1.0 or higher

Remarks

If the requested number of sources cannot be created, an error will be generated which can be detected with alGetError. If an error occurs, no sources will be generated. If n equals zero, alGenSources does nothing and does not return an error.

alDeleteSources

Description

This function deletes one or more sources.

```
void alDeleteSources(
        ALsizei n,
        ALuint *sources
);
```

Parameters

n	the number of sources to be deleted
sources	pointer to an array of source names identifying the sources to be deleted

Possible Error States

State	***Description***
AL_INVALID_NAME	At least one specified source is not valid, or an attempt is being made to delete more sources than exist.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

If the requested number of sources cannot be deleted, an error will be generated which can be detected with alGetError. If an error occurs, no sources will be deleted. If n equals zero, alDeleteSources does nothing and will not return an error.

A playing source can be deleted – the source will be stopped and then deleted.

alIsSource

Description

This function tests if a source name is valid, returning AL_TRUE if valid and AL_FALSE if not.

```
boolean alIsSource(
        ALuint source
);
```

Parameters

source a source name to be tested for validity

Possible Error States

State	***Description***
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alSourcef

Description

This function sets a floating point property of a source. The relevant properties are listed in the table *Source Properties.*

```
void alSourcef(
        ALuint source,
        ALenum param,
        ALfloat value
);
```

Parameters

source	source name whose attribute is being set
param	the name of the attribute to set: AL_PITCH AL_GAIN AL_MAX_DISTANCE AL_ROLLOFF_FACTOR AL_REFERENCE_DISTANCE AL_MIN_GAIN AL_MAX_GAIN AL_CONE_OUTER_GAIN
value	the value to set the attribute to

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value given is out of range.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alSource3f

Description

This function sets a source property requiring three floating point values. The relevant properties are listed in the table *Source Properties.*

```
void alSource3f(
        ALuint source,
        ALenum param,
        ALfloat v1,
        ALfloat v2,
        ALfloat v3
);
```

Parameters

source source name whose attribute is being set

param the name of the attribute to set:
AL_POSITION
AL_VELOCITY
AL_DIRECTION

v1, v2, v3 the three ALfloat values which the attribute will be set to

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value given is out of range.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

This function is an alternative to alSourcefv.

alSourcefv

Description

This function sets a floating point-vector property of a source. The relevant properties are listed in the table *Source Properties.*

```
void alSourcefv(
        ALuint source,
        ALenum param,
        ALfloat *values
);
```

Parameters

source	source name whose attribute is being set
param	the name of the attribute being set: AL_POSITION AL_VELOCITY AL_DIRECTION
values	a pointer to the vector to set the attribute to

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value given is out of range.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

This function is an alternative to alSource3f.

alSourcei

Description

This function sets an integer property of a source. The relevant properties are listed in the table *Source Properties.*

```
void alSourcei(
        ALuint source,
        ALenum param,
        ALint value
);
```

Parameters

source	source name whose attribute is being set
param	the name of the attribute to set: AL_SOURCE_RELATIVE AL_CONE_INNER_ANGLE AL_CONE_OUTER_ANGLE AL_LOOPING AL_BUFFER AL_SOURCE_STATE
value	the value to set the attribute to

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value given is out of range.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The buffer name zero is reserved as a “NULL Buffer" and is accepted by *alSourcei(…, AL_BUFFER, …)* as a valid buffer of zero length. The NULL Buffer is extremely useful for detaching buffers from a source which were attached using this call or with *alSourceQueueBuffers*.

alSource3i

Description

This function sets an integer property of a source. The relevant properties are listed in the table *Source Properties.*

```
void alSourcei(
        ALuint source,
        ALenum param,
        ALint v1,
        ALint v2,
        ALint v3
);
```

Parameters

source	source name whose attribute is being set
param	the name of the attribute to set: AL_POSITION AL_VELOCITY AL_DIRECTION
v1, v2, v3	the values to set the attribute to

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value given is out of range.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.1 or higher

Remarks

None

alSourceiv

Description

This function sets an integer property of a source. The relevant properties are listed in the table *Source Properties.*

```
void alSourcei(
        ALuint source,
        ALenum param,
        ALint *values
);
```

Parameters

source	source name whose attribute is being set
param	the name of the attribute to set: AL_POSITION AL_VELOCITY AL_DIRECTION
values	the values to set the attribute to

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value given is out of range.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.1 or higher

Remarks

None

alGetSourcef

Description

This function retrieves a floating point property of a source. The relevant properties are listed in the table *Source Properties.*

```
void alGetSourcef(
        ALuint source,
        ALenum param,
        ALfloat *value
);
```

Parameters

source	source name whose attribute is being retrieved
param	the name of the attribute to retrieve: AL_PITCH AL_GAIN AL_MIN_GAIN AL_MAX_GAIN AL_MAX_DISTANCE AL_ROLLOFF_FACTOR AL_CONE_OUTER_GAIN AL_CONE_INNER_ANGLE AL_CONE_OUTER_ANGLE AL_REFERENCE_DISTANCE
value	a pointer to the floating point value being retrieved

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alGetSource3f

Description

This function retrieves three floating point values representing a property of a source. The relevant properties are listed in the table *Source Properties.*

```
void alGetSource3f(
        ALuint source,
        ALenum param,
        ALfloat *v1,
        ALfloat *v2,
        ALfloat *v3
);
```

Parameters

source	source name whose attribute is being retrieved
param	the name of the attribute being retrieved: AL_POSITION AL_VELOCITY AL_DIRECTION
v1, v2, v3	pointers to the values to retrieve

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.1 or higher

Remarks

None

alGetSourcefv

Description

This function retrieves a floating point-vector property of a source. The relevant properties are listed in the table *Source Properties.*

```
void alGetSourcefv(
        ALuint source,
        ALenum param,
        ALfloat *values
);
```

Parameters

source	source name whose attribute is being retrieved
param	the name of the attribute being retrieved: AL_POSITION AL_VELOCITY AL_DIRECTION
values	a pointer to the vector to retrieve

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alGetSourcei

Description

This function retrieves an integer property of a source. The relevant properties are listed in the table *Source Properties.*

```
void alGetSourcei(
        ALuint source,
        ALenum pname,
        ALint *value
);
```

Parameters

source	source name whose attribute is being retrieved
pname	the name of the attribute to retrieve: AL_SOURCE_RELATIVE AL_BUFFER AL_SOURCE_STATE AL_BUFFERS_QUEUED AL_BUFFERS_PROCESSED
value	a pointer to the integer value being retrieved

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alGetSource3i

Description

This function retrieves an integer property of a source. The relevant properties are listed in the table *Source Properties.*

```
void alGetSourcei(
        ALuint source,
        ALenum param,
        ALint *v1,
        ALint *v2,
        ALint *v3
);
```

Parameters

source	source name whose attribute is being retrieved
pname	the name of the attribute to retrieve: AL_POSITION AL_VELOCITY AL_DIRECTION
v1, v2, v3	pointers to the integer values being retrieved

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.1 or higher

Remarks

None

alGetSourceiv

Description

This function retrieves an integer property of a source. The relevant properties are listed in the table *Source Properties.*

```
void alGetSourceiv(
        ALuint source,
        ALenum param,
        ALint *values
);
```

Parameters

source	source name whose attribute is being retrieved
param	the name of the attribute to retrieve: AL_POSITION AL_VELOCITY AL_DIRECTION
values	pointer to the integer values being retrieved

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.1 or higher

Remarks

None

alSourcePlay

Description

This function plays a source.

```
void alSourcePlay(
        ALuint source
);
```

Parameters

source the name of the source to be played

Possible Error States

State	***Description***
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The playing source will have its state changed to AL_PLAYING. When called on a source which is already playing, the source will restart at the beginning. When the attached buffer(s) are done playing, the source will progress to the AL_STOPPED state.

alSourcePlayv

Description

This function plays a set of sources.

```
void alSourcePlayv(
        ALsizei n,
        ALuint *sources
);
```

Parameters

n	the number of sources to be played
sources	a pointer to an array of sources to be played

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The playing sources will have their state changed to AL_PLAYING. When called on a source which is already playing, the source will restart at the beginning. When the attached buffer(s) are done playing, the source will progress to the AL_STOPPED state.

alSourcePause

Description

This function pauses a source.

```
void alSourcePause(
        ALuint source
);
```

Parameters

source the name of the source to be paused

Possible Error States

State	***Description***
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The paused source will have its state changed to AL_PAUSED.

alSourcePausev

Description

This function pauses a set of sources.

```
void alSourcePausev(
        ALsizei n,
        ALuint *sources
);
```

Parameters

n	the number of sources to be paused
sources	a pointer to an array of sources to be paused

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The paused sources will have their state changed to AL_PAUSED.

alSourceStop

Description

This function stops a source.

```
void alSourceStop(
        ALuint source
);
```

Parameters

source the name of the source to be stopped

Possible Error States

State	***Description***
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The stopped source will have its state changed to AL_STOPPED.

alSourceStopv

Description

This function stops a set of sources.

```
void alSourceStopv(
        ALsizei n,
        ALuint *sources
);
```

Parameters

n	the number of sources to stop
sources	a pointer to an array of sources to be stopped

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The stopped sources will have their state changed to AL_STOPPED.

alSourceRewind

Description

This function stops the source and sets its state to AL_INITIAL.

```
void alSourceRewind(
        ALuint source
);
```

Parameters

source the name of the source to be rewound

Possible Error States

State	***Description***
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alSourceRewindv

Description

This function stops a set of sources and sets all their states to AL_INITIAL.

```
void alSourceRewindv(
        ALsizei n,
        ALuint *sources
);
```

Parameters

n	the number of sources to be rewound
sources	*a* pointer to an array of sources to be rewound

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alSourceQueueBuffers

Description

This function queues a set of buffers on a source. All buffers attached to a source will be played in sequence, and the number of processed buffers can be detected using an *alSourcei* call to retrieve AL_BUFFERS_PROCESSED.

```
void alSourceQueueBuffers(
        ALuint source,
        ALsizei n,
        ALuint* buffers
);
```

Parameters

source	the name of the source to queue buffers onto
n	the number of buffers to be queued
buffers	a pointer to an array of buffer names to be queued

Possible Error States

State	***Description***
AL_INVALID_NAME	At least one specified buffer name is not valid, or the specified source name is not valid.
AL_INVALID_OPERATION	There is no current context, an attempt was made to add a new buffer which is not the same format as the buffers already in the queue, or the source already has a static buffer attached.

Version Requirements

OpenAL 1.0 or higher

Remarks

When first created, a source will be of type AL_UNDETERMINED. A successful *alSourceQueueBuffers* call will change the source type to AL_STREAMING.

alSourceUnqueueBuffers

Description

This function unqueues a set of buffers attached to a source. The number of processed buffers can be detected using an *alSourcei* call to retrieve AL_BUFFERS_PROCESSED, which is the maximum number of buffers that can be unqueued using this call.

```
void alSourceUnqueueBuffers(
        ALuint source,
        ALsizei n,
        ALuint* buffers
);
```

Parameters

source	the name of the source to unqueue buffers from
n	the number of buffers to be unqueued
buffers	a pointer to an array of buffer names that were removed

Possible Error States

State	***Description***
AL_INVALID_VALUE	At least one buffer can not be unqueued because it has not been processed yet.
AL_INVALID_NAME	The specified source name is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The unqueue operation will only take place if all n buffers can be removed from the queue.

Listener-Related

alListenerf

Description

This function sets a floating point property for the listener. The relevant properties are listed in the table *Listener Properties.*

```
void alListenerf(
        ALenum param,
        ALfloat value
);
```

Parameters:

param	the name of the attribute to be set: AL_GAIN
value	the ALfloat value to set the attribute to

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alListener3f

Description

This function sets a floating point property for the listener. The relevant properties are listed in the table *Listener Properties.*

```
void alListener3f(
        ALenum param,
        ALfloat v1,
        ALfloat v2,
        ALfloat v3
);
```

Parameters

param	the name of the attribute to set: AL_POSITION AL_VELOCITY
v1, v2, v3	the value to set the attribute to

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alListenerfv

Description

This function sets a floating point-vector property of the listener. The relevant properties are listed in the table *Listener Properties.*

```
void alListenerfv(
        ALenum param,
        ALfloat *values
);
```

Parameters

param	the name of the attribute to be set: AL_POSITION AL_VELOCITY AL_ORIENTATION
values	pointer to floating point-vector values

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alListeneri

Description

This function sets an integer property of the listener.

```
void alListeneri(
        ALenum param,
        ALint value
);
```

Parameters

param	the name of the attribute to be set
value	the integer value to set the attribute to

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

There are no integer listener attributes defined for OpenAL 1.1, but this function may be used by an extension.

alListener3i

Description

This function sets an integer property of the listener. The relevant properties are listed in the table *Listener Properties.*

```
void alListener3i(
        ALenum param,
        ALint v1,
        ALint v2,
        ALint v3
);
```

Parameters

param	the name of the attribute to be set: AL_POSITION AL_VELOCITY
v1, v2, v3	the integer values to set the attribute to

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.1 or higher

Remarks

None

alListeneriv

Description

This function sets an integer property of the listener. The relevant properties are listed in the table *Listener Properties.*

```
void alListeneriv(
        ALenum param,
        ALint *values
);
```

Parameters

param the name of the attribute to be set
AL_POSITION
AL_VELOCITY
AL_ORIENTATION

values pointer to the integer values to set the attribute to

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.1 or higher

Remarks

None

alGetListenerf

Description

This function retrieves a floating point property of the listener. The relevant properties are listed in the table *Listener Properties.*

```
void alGetListenerf(
        ALenum param,
        ALfloat *value
);
```

Parameters

param	the name of the attribute to be retrieved: AL_GAIN
value	a pointer to the floating point value being retrieved

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alGetListener3f

Description

This function retrieves a set of three floating point values from a property of the listener. The relevant properties are listed in the table *Listener Properties.*

```
void alGetListener3f(
        ALenum param,
        ALfloat *v1,
        ALfloat *v2,
        ALfloat *v3
);
```

Parameters

param	the name of the attribute to be retrieved AL_POSITION AL_VELOCITY
v1, v2, v3	pointers to the three floating point being retrieved

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alGetListenerfv

Description

This function retrieves a floating point-vector property of the listener. The relevant properties are listed in the table *Listener Properties.*

```
void alGetListenerfv(
        ALenum param,
        ALfloat *values
);
```

Parameters

param the name of the attribute to be retrieved
AL_POSITION
AL_VELOCITY
AL_ORIENTATION

values a pointer to the floating point-vector value being retrieved

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alGetListeneri

Description

This function retrieves an integer property of the listener.

```
void alGetListeneri(
        ALenum param,
        ALint *value
);
```

Parameters

param	the name of the attribute to be retrieved
value	a pointer to the integer value being retrieved

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

There are no integer listener attributes defined for OpenAL 1.1, but this function may be used by an extension.

alGetListener3i

Description

This function retrieves an integer property of the listener. The relevant properties are listed in the table *Listener Properties.*

```
void alGetListeneri(
        ALenum param,
        ALint *v1,
        ALint *v2,
        ALint *v3
);
```

Parameters

param	the name of the attribute to be retrieved AL_POSITION AL_VELOCITY
v1, v2, v3	pointers to the integer values being retrieved

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.1 or higher

Remarks

None

alGetListeneriv

Description

This function retrieves an integer property of the listener. The relevant properties are listed in the table *Listener Properties.*

```
void alGetListeneriv(
        ALenum param,
        ALint *values
);
```

Parameters

param	the name of the attribute to be retrieved AL_POSITION AL_VELOCITY AL_ORIENTATION
values	a pointer to the integer values being retrieved

Possible Error States

State	***Description***
AL_INVALID_VALUE	The value pointer given is not valid.
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.1 or higher

Remarks

None

State-Related

alEnable

Description

This function enables a feature of the OpenAL driver.

```
void alEnable(
        ALenum capability
);
```

Parameters

capability the name of a capability to enable

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified capability is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

There are no capabilities defined in OpenAL 1.1 to be used with this function, but it may be used by an extension.

alDisable

Description

This function disables a feature of the OpenAL driver.

```
void alDisable(
        ALenum capability
);
```

Parameters

capability the name of a capability to disable

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified capability is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

There are no capabilities defined in OpenAL 1.1 to be used with this function, but it may be used by an extension.

alIsEnabled

Description

This function returns a boolean indicating if a specific feature is enabled in the OpenAL driver.

```
ALboolean alIsEnabled(
        ALenum capability
);
```

Parameters

capability the name of a capability to enable

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified capability is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns AL_TRUE if the capability is enabled, AL_FALSE if the capability is disabled. There are no capabilities defined in OpenAL 1.1 to be used with this function, but it may be used by an extension.

alGetBoolean

Description:

This function returns a boolean OpenAL state.

```
ALboolean alGetBoolean(
        ALenum  param
);
```

Parameters

param the state to be queried:
AL_DOPPLER_FACTOR
AL_SPEED_OF_SOUND
AL_DISTANCE_MODEL

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The boolean state described by param will be returned.

alGetDouble

Description

This function returns a double precision floating point OpenAL state.

```
Aldouble alGetDouble(
        ALenum  param
);
```

Parameters

param	the state to be queried: AL_DOPPLER_FACTOR AL_SPEED_OF_SOUND AL_DISTANCE_MODEL

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The double value described by param will be returned.

alGetFloat

Description

This function returns a floating point OpenAL state.

```
ALfloat  alGetFloat(
        ALenum  param
);
```

Parameters

param the state to be queried:
AL_DOPPLER_FACTOR
AL_SPEED_OF_SOUND
AL_DISTANCE_MODEL

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The floating point state described by param will be returned.

alGetInteger

Description

This function returns an integer OpenAL state.

```
Alint alGetInteger(
        ALenum  param
);
```

Parameters

param	the state to be queried: AL_DOPPLER_FACTOR AL_SPEED_OF_SOUND AL_DISTANCE_MODEL

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The integer state described by param will be returned.

alGetBooleanv

Description

This function retrieves a boolean OpenAL state.

```
void alGetBooleanv(
        ALenum param,
        ALboolean *data
);
```

Parameters

param	the state to be returned: AL_DOPPLER_FACTOR AL_SPEED_OF_SOUND AL_DISTANCE_MODEL
data	a pointer to the location where the state will be stored

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_VALUE	The specified data pointer is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alGetDoublev

Description

This function retrieves a double precision floating point OpenAL state.

```
void alGetDoublev(
        ALenum param,
        ALdouble *data
);
```

Parameters

param	the state to be returned: AL_DOPPLER_FACTOR AL_SPEED_OF_SOUND AL_DISTANCE_MODEL
data	a pointer to the location where the state will be stored

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_VALUE	The specified data pointer is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alGetFloatv

Description

This function retrieves a floating point OpenAL state.

```
void alGetFloatv(
        ALenum param,
        ALfloat *data
);
```

Parameters

param	the state to be returned: AL_DOPPLER_FACTOR AL_SPEED_OF_SOUND AL_DISTANCE_MODEL
data	a pointer to the location where the state will be stored

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_VALUE	The specified data pointer is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alGetIntegerv

Description

This function retrieves an integer OpenAL state.

```
void alGetIntegerv(
        ALenum param,
        ALint *data
);
```

Parameters

param	The state to be returned: AL_DOPPLER_FACTOR AL_SPEED_OF_SOUND AL_DISTANCE_MODEL
data	a pointer to the location where the state will be stored

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.
AL_INVALID_VALUE	The specified data pointer is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

None

alGetString

Description

This function retrieves an OpenAL string property.

```
const ALchar * alGetString(
        ALenum param
);
```

Parameters

param	The property to be returned	
		AL_VENDOR
		AL_VERSION
		AL_RENDERER
		AL_EXTENSIONS

Possible Error States

State	***Description***
AL_INVALID_ENUM	The specified parameter is not valid.

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns a pointer to a null-terminated string.

alDistanceModel

Description

This function selects the OpenAL distance model.

The AL_INVERSE_DISTANCE model works according to the following formula:

G_dB = AL_GAIN – 20log10(1 + AL_ROLLOFF_FACTOR*(distance – AL_REFERENCE_DISTANCE)/AL_REFERENCE_DISTANCE));
G_dB = min(G_dB, AL_MAX_GAIN);
G_dB = max(G_dB, AL_MIN_GAIN);

The AL_INVERSE_DISTANCE_CLAMPED model works according to the following formula:

distance = max(distance, AL_REFERENCE_DISTANCE);
distance = min(distance, AL_MAX_DISTANCE);
G_dB = AL_GAIN – 20log10(1 + AL_ROLLOFF_FACTOR*(distance – AL_REFERENCE_DISTANCE)/AL_REFERENCE_DISTANCE));
G_dB = min(G_dB, AL_MAX_GAIN);
G_dB = max(G_dB, AL_MIN_GAIN);

The AL_NONE model works according to the following formula:

G_db = AL_GAIN;

```
void alDistanceModel(
        ALenum value
);
```

Parameters

value	the distance model to be set:	AL_NONE AL_INVERSE_DISTANCE AL_INVERSE_DISTANCE_CLAMPED

Possible Error States

State	***Description***
AL_INVALID_VALUE	The specified distance model is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The default distance model in OpenAL is AL_INVERSE_DISTANCE_CLAMPED.

alDopplerFactor

Description

This function selects the OpenAL Doppler factor value.

```
void alDopplerFactor(
        ALfloat value
);
```

Parameters

value the Doppler scale value to set

Possible Error States

State	***Description***
AL_INVALID_VALUE	The specified value is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.0 or higher

Remarks

The default Doppler factor value is 1.0.

alSpeedOfSound

Description

This function selects the speed of sound for use in Doppler calculations.

```
void alSpeedOfSound(
        ALfloat value
);
```

Parameters

value the speed of sound value to set

Possible Error States

State	***Description***
AL_INVALID_VALUE	The specified value is not valid.
AL_INVALID_OPERATION	There is no current context.

Version Requirements

OpenAL 1.1 or higher

Remarks

The default speed of sound value is 343.3.

Error-Related

alGetError

Description

This function returns the current error state and then clears the error state.

```
ALenum alGetError(ALvoid);
```

Parameters

None

Possible Error States

None

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns an Alenum representing the error state. When an OpenAL error occurs, the error state is set and will not be changed until the error state is retrieved using alGetError. Whenever alGetError is called, the error state is cleared and the last state (the current state when the call was made) is returned. To isolate error detection to a specific portion of code, alGetError should be called before the isolated section to clear the current error state.

Extension-Related

alIsExtensionPresent

Description

This function tests if a specific extension is available for the OpenAL driver.

```
ALboolean alIsExtensionPresent(
        const ALchar *extname
);
```

Parameters

extname a null-terminated string describing the desired extension

Possible Error States

State	***Description***
AL_INVALID_VALUE	The specified extension string is not a valid pointer.

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns AL_TRUE if the extension is available, AL_FALSE if the extension is not available.

alGetProcAddress

Description

This function returns the address of an OpenAL extension function.

```
void * alGetProcAddress(
        const ALchar *fname
);
```

Parameters

fname	a null-terminated string containing the function name

Possible Error States

None

Version Requirements

OpenAL 1.0 or higher

Remarks

The return value is a pointer to the specified function. The return value will be NULL if the function is not found.

alGetEnumValue

Description

This function returns the enumeration value of an OpenAL enum described by a string.

```
ALenum alGetEnumValue(
        const ALchar *ename
);
```

Parameters

ename a null-terminated string describing an OpenAL enum

Possible Error States

None

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns the actual ALenum described by a string. Returns NULL if the string doesn't describe a valid OpenAL enum.

ALC Functions

Context-Related

alcCreateContext

Description

This function creates a context using a specified device.

```
ALCcontext * alcCreateContext(
        ALCdevice *device,
        ALCint* attrlist
);
```

Parameters

device	a pointer to a device
attrlist	a pointer to a set of attributes: ALC_FREQUENCY ALC_REFRESH ALC_SYNC

Possible Error States

State	***Description***
ALC_INVALID_VALUE	An additional context can not be created for this device.
ALC_INVALID_DEVICE	The specified device is not a valid output device.

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns a pointer to the new context (NULL on failure).

alcMakeContextCurrent

Description

This function makes a specified context the current context.

```
ALCboolean alcMakeContextCurrent(
        ALCcontext *context
);
```

Parameters

context a pointer to the new context

Possible Error States

State	***Description***
ALC_INVALID_CONTEXT	The specified context is invalid.

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns ALC_TRUE on success, or ALC_FALSE on failure.

alcProcessContext

Description

This function tells a context to begin processing.

```
void alcProcessContext(
        ALCcontext *context
);
```

Parameters

context a pointer to the new context

Possible Error States

State	***Description***
ALC_INVALID_CONTEXT	The specified context is invalid.

Version Requirements

OpenAL 1.0 or higher

Remarks

When a context is suspended, changes in OpenAL state will be accepted but will not be processed. *alcSuspendContext* can be used to suspend a context, and then all the OpenAL state changes can be applied at once, followed by a call to *alcProcessContext* to apply all the state changes immediately. In some cases, this procedure may be more efficient than application of properties in a non-suspended state. In some implementations, process and suspend calls are each a NOP.

alcSuspendContext

Description

This function suspends processing on a specified context.

```
void alcSuspendContext(
        ALCcontext *context
);
```

Parameters

context a pointer to the context to be suspended

Possible Error States

State	***Description***
ALC_INVALID_CONTEXT	The specified context is invalid.

Version Requirements

OpenAL 1.0 or higher

Remarks

When a context is suspended, changes in OpenAL state will be accepted but will not be processed. A typical use of *alcSuspendContext* would be to suspend a context, apply all the OpenAL state changes at once, and then call *alcProcessContext* to apply all the state changes at once. In some cases, this procedure may be more efficient than application of properties in a non-suspended state. In some implementations, process and suspend calls are each a NOP.

alcDestroyContext

Description

This function destroys a context.

```
void alcDestroyContext(
        ALCcontext *context
);
```

Parameters

context a pointer to the new context

Possible Error States

State	***Description***
ALC_INVALID_CONTEXT	The specified context is invalid.

Version Requirements

OpenAL 1.0 or higher

Remarks

A context which is not current can be destroyed at any time (all sources within that context will also be deleted – buffers are not affected). *alcMakeContextCurrent* should be used to make sure the context to be destroyed is not current (NULL is valid for *alcMakeContextCurrent*).

alcGetCurrentContext

Description

This function retrieves the current context.

```
ALCcontext * alcGetCurrentContext( ALCvoid );
```

Parameters:

None

Possible Error States

None

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns a pointer to the current context.

alcGetContextsDevice

Description

This function retrieves a context's device pointer.

```
ALCdevice * alcGetContextsDevice( ALCcontext *context );
```

Parameters:

context a pointer to a context

Possible Error States

State	***Description***
ALC_INVALID_CONTEXT	The specified context is invalid.

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns a pointer to the specified context's device.

Error-Related

alcGetError

Description

This function retrieves the current context error state.

```
ALCenum alcGetError( ALCdevice *device );
```

Parameters

device a pointer to the device to retrieve the error state from

Possible Error States

None

Version Requirements

OpenAL 1.0 or higher

Remarks

None

Device-Related

alcOpenDevice

Description

This function opens a device by name.

```
ALCdevice *alcOpenDevice(
        const ALCchar *devicename
);
```

Parameters

devicename a null-terminated string describing a device

Possible Error States

The return value will be NULL if there is an error.

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns a pointer to the opened device. Will return NULL if a device can not be opened.

alcCloseDevice

Description

This function closes a device by name.

```
ALCboolean alcCloseDevice(
        ALCdevice *device
);
```

Parameters:

device a pointer to an opened device

Possible Error States

State	***Description***
ALC_INVALID_DEVICE	The specified device name doesn't exist.

Version Requirements

OpenAL 1.0 or higher

Remarks

Any contexts and buffers within the device will be destroyed when the device is closed, and ALC_TRUE will be returned on success or ALC_FALSE on failure.

Extension-Related

alcIsExtensionPresent

Description

This function queries if a specified context extension is available.

```
ALCboolean alcIsExtensionPresent(
        ALCdevice *device,
        const ALCchar *extName
);
```

Parameters

device	a pointer to the device to be queried for an extension
extName	a null-terminated string describing the extension

Possible Error States

State	***Description***
ALC_INVALID_VALUE	The string pointer is not valid.

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns ALC_TRUE if the extension is available, ALC_FALSE if the extension is not available.

alcGetProcAddress

Description

This function retrieves the address of a specified context extension function.

```
void  * alcGetProcAddress(
            ALCdevice *device,
            const ALCchar *funcName
);
```

Parameters

device	a pointer to the device to be queried for the function
funcName	a null-terminated string describing the function

Possible Error States

State	***Description***
ALC_INVALID_VALUE	The string pointer is not valid.

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns the address of the function, or NULL if it is not found.

alcGetEnumValue

Description

This function retrieves the enum value for a specified enumeration name.

```
ALCenum alcGetEnumValue(
        ALCdevice *device,
        const ALCchar *enumName
);
```

Parameters

device	a pointer to the device to be queried
enumName	a null terminated string describing the enum value

Possible Error States

State	***Description***
ALC_INVALID_VALUE	The string pointer is not valid.

Version Requirements

OpenAL 1.0 or higher

Remarks

Returns the enum value described by the enumName string. This is most often used for querying an enum value for an ALC extension.

State-Related

alcGetString

Description

This function returns pointers to strings related to the context.

```
const ALCchar * alcGetString(
        ALCdevice *device,
        ALenum param
);
```

Parameters

device a pointer to the device to be queried

param an attribute to be retrieved:
ALC_DEFAULT_DEVICE_SPECIFIER
ALC_CAPTURE_DEFAULT_DEVICE_SPECIFIER
ALC_DEVICE_SPECIFIER
ALC_CAPTURE_DEVICE_SPECIFIER
ALC_EXTENSIONS

Possible Error States

State	***Description***
ALC_INVALID_ENUM	The specified parameter is not valid.

Version Requirements

OpenAL 1.0 or higher

Remarks

ALC_DEFAULT_DEVICE_SPECIFIER will return the name of the default output device.

ALC_CAPTURE_DEFAULT_DEVICE_SPECIFIER will return the name of the default capture device.

ALC_DEVICE_SPECIFIER will return the name of the specified output device if a pointer is supplied, or will return a list of all available devices if a NULL device pointer is supplied. A list is a pointer to a series of strings separated by NULL characters, with the list terminated by two NULL characters.

ALC_CAPTURE_DEVICE_SPECIFIER will return the name of the specified capture device if a pointer is supplied, or will return a list of all available devices if a NULL device pointer is supplied.

ALC_EXTENSIONS returns a list of available context extensions, with each extension separated by a space and the list terminated by a NULL character.

alcGetIntegerv

Description

This function returns integers related to the context.

```
void alcGetIntegerv(
        ALCdevice *device,
        ALCenum param,
        ALCsizei size,
        ALCint *data
);
```

Parameters

device	a pointer to the device to be queried
param	an attribute to be retrieved: ALC_MAJOR_VERSION ALC_MINOR_VERSION ALC_ATTRIBUTES_SIZE ALC_ALL_ATTRIBUTES
size	the size of the destination buffer provided
data	a pointer to the data to be returned

Possible Error States

State	***Description***
ALC_INVALID_VALUE	The specified data pointer or size is not valid.
ALC_INVALID_ENUM	The specified parameter is not valid.
ALC_INVALID_DEVICE	The specified device is not valid.
ALC_INVALID_CONTEXT	The specified context is not valid.

Version Requirements

OpenAL 1.0 or higher

Remarks

The versions returned refer to the specification version that the implementation meets.

Capture-Related

alcCaptureOpenDevice

Description

This function opens a capture device by name.

```
ALCdevice * alcCaptureOpenDevice(
        const ALCchar *devicename,
        ALCuint frequency,
        ALCenum format,
        ALCsizei buffersize
);
```

Parameters

devicename	a pointer to a device name string
frequency	the frequency that the data should be captured at
format	the requested capture buffer format
buffersize	the size of the capture buffer

Possible Error States

State	***Description***
ALC_INVALID_VALUE	One of the parameters has an invalid value.
ALC_OUT_OF_MEMORY	The specified device is invalid, or can not capture audio.

Version Requirements

OpenAL 1.1 or higher

Remarks

Returns the capture device pointer, or NULL on failure.

alcCaptureCloseDevice

Description

This function closes the specified capture device.

```
ALCboolean alcCaptureCloseDevice(
        ALCdevice *device
);
```

Parameters

device a pointer to a capture device

Possible Error States

State	***Description***
ALC_INVALID_DEVICE	The specified device is not a valid capture device.

Version Requirements

OpenAL 1.1 or higher

Remarks

Returns ALC_TRUE if the close operation was successful, ALC_FALSE on failure.

alcCaptureStart

Description

This function begins a capture operation.

```
void alcCaptureStart(
        ALCdevice *device
);
```

Parameters

device a pointer to a capture device

Possible Error States

State	***Description***
ALC_INVALID_DEVICE	The specified device is not a valid capture device.

Version Requirements

OpenAL 1.1 or higher

Remarks

alcCaptureStart will begin recording to an internal ring buffer of the size specified when opening the capture device. The application can then retrieve the number of samples currently available using the ALC_CAPTURE_SAPMPLES token with *alcGetIntegerv*. When the application determines that enough samples are available for processing, then it can obtain them with a call to *alcCaptureSamples*.

alcCaptureStop

Description

This function stops a capture operation.

```
void alcCaptureStop(
        ALCdevice *device
);
```

Parameters

device a pointer to a capture device

Possible Error States

State	***Description***
ALC_INVALID_DEVICE	The specified device is not a valid capture device.

Version Requirements

OpenAL 1.1 or higher

Remarks

None

alcCaptureSamples

Description

This function completes a capture operation.

```
void alcCaptureSamples(
        ALCdevice *device,
        ALCvoid *buffer,
        ALCsizei samples
);
```

Parameters

device a pointer to a capture device

buffer a pointer to a data buffer, which must be large enough to accommodate *samples* number of samples

samples the number of samples to be retrieved

Possible Error States

State	***Description***
ALC_INVALID_VALUE	The specified number of samples is larger than the number of available samples.
ALC_INVALID_DEVICE	The specified device is not a valid capture device.

Version Requirements

OpenAL 1.1 or higher

Remarks

None

ALC and AL Function Lists

Functions new to OpenAL 1.1 are *italicized* and **boldface**.

ALC Functions

alcCreateContext
alcMakeContextCurrent
alcProcessContext
alcSuspendContext
alcDestroyContext
alcGetCurrentContext
alcGetContextsDevice
alcOpenDevice
alcCloseDevice
alcGetError
alcIsExtensionPresent
alcGetProcAddress
alcGetEnumValue
alcGetString
alcGetIntegerv
alcCaptureOpenDevice
alcCaptureCloseDevice
alcCaptureStart
alcCaptureStop
alcCaptureSamples

AL Functions

alEnable
alDisable
alIsEnabled
alGetString
alGetBooleanv
alGetIntegerv
alGetFloatv
alGetDoublev
alGetBoolean
alGetInteger
alGetFloat
alGetDouble
alGetError
alIsExtensionPresent
alGetProcAddress
alGetEnumValue
alListenerf
alListener3f
alListenerfv
alListeneri
alListener3i
alListeneriv
alGetListenerf
alGetListener3f
alGetListenerfv
alGetListeneri
alGetListener3i
alGetListeneriv
alGenSources

alDeleteSources
alIsSource
alSourcef
alSource3f
alSourcefv
alSourcei
alSource3i
alSourceiv
alGetSourcef
alGetSource3f
alGetSourcefv
alGetSourcei
alGetSource3i
alGetSourceiv
alSourcePlayv
alSourceStopv
alSourceRewindv
alSourcePausev
alSourcePlay
alSourceStop
alSourceRewind
alSourcePause
alSourceQueueBuffers
alSourceUnqueueBuffers
alGenBuffers
alDeleteBuffers
alIsBuffer
alBufferf
alBuffer3f
alBufferfv
alBufferi
alBuffer3i
alBufferiv
alGetBufferf
alGetBuffer3f
alGetBufferfv
alGetBufferi
alGetBuffer3i
alGetBufferiv
alDopplerFactor
alDopplerVelocity
alSpeedOfSound
alDistanceModel