ALSound

FreePascal + OpenAL-Soft + LibSndFile

lulu - 2022

Table of content

Introduction

ALSound use OpenAL-Soft and LibSndFile libraries to provide an easy way to play, capture and mix sounds under Freepascal/Lazarus.

Only one single unit to include:

uses ALSound;

That's all !
Libraries are loaded automatically at startup and unloaded at application shutdown.

Put a copy of the libraries in your application folder executable.

ALSManager instance : the tools box

ALSManager is an unique instance of TALSManager class. It is created and freed automatically.

function ListOfPlaybackDeviceName: TStringArray;
Gives the list of playback device names found by OpenAL-Soft

function DefaultPlaybackDeviceName: string;
Gives the default playback device name. Empty string if none.

function CreateDefaultPlaybackContext: TALSPlaybackContext;
Opens the default playback device and creates a context on it.

function CreatePlaybackContext(aNameIndex: integer; const aAttribs: TALSContextAttributes): TALSPlaybackContext;
Opens the specified playback device and creates a context with custom attributes. ***aNameIndex*** is an index in the list provided by ALSManager.ListOfPlaybackDeviceName. You can set aNameIndex to -1 to refer to the default playback device. ***aAttribs*** see #0.0.4.1.Context attributes - record|outline context attributes.

function CreateDefaultLoopbackContext: TALSLoopbackContext;
Creates a default loopback context.

function ListOfCaptureDeviceName: TStringArray;
Gives the list of capture device names found by OpenAL-Soft

function DefaultCaptureDeviceName: string;
Gives the name of the default capture device. Empty string if none.

function CreateDefaultCaptureContext: TALSCaptureContext;
Opens the default capture device and creates a capture context on it.

function CreateCaptureContext(aNameIndex: integer; aFrequency: longword; aFormat: TALSCaptureFormat; aBufferSize: double): TALSCaptureContext;
Opens the specified capture device and creates a capture context on it with custom attributes.

aNameIndex is an index in the list provided by ListOfCaptureDeviceName. You can set aNameIndex to -1 to refer to the default capture device.

aFrequency: the frequency of the capture.

aFormat: the sample format

aBufferSize: the length in seconds of the internal buffer used by OpenAL-Soft

function ListOfAudioFileFormat_Complete: ArrayOfALSAudioFileFormat;
Gives the complete list of audio file format and sub-format supported by LibSndFile.

function ListOfAudioFileFormat_Simplified: ArrayOfALSSimplifiedAudioFileFormat;
Gives a simplified list of audio file format supported by libsndfile.

function DialogFileFilters(aIncludeAllFilesFilter: boolean): string;

Gives the supported list of audio file extension formatted to be used directly with TOpenDialog.Filter and TSaveDialog.Filter property. If ***aIncludeAllFilesFilter*** is True, adds also 'All files|*.*'

function ListOfPlaybackOutputMode: TStringArray;

Gives the list of available output mode expressed in string. Output mode can be applyed only on playback context. this is the way OpenAL-Soft will render the final mix on a playback context. Value can be stereo, surround 5.1, HRTF,... See TOALSPlaybackMixMode for the available mode.

function PlaybackOutputModeIndexToEnum(aIndex: integer): TALSPlaybackContextOutputMode;

Converts an index in the list of output mode to its corresponding enum value.

Context attributes – record

Context attributes is a record used to customize a playback or loopback context.
Fields are :

SampleRate : integer ;
the desired sample rate for the context. If the device is not capable to render audio at this sample rate, the nearest valid value is choosen. Range is from 8000 to 192000Hz.

MonoCount : integer ;
Number of mono sound the context can play - default 128.

StereoCount: integer;
Number of stereo sound the context can play - default 128.

ContextUseFloat: boolean;
TRUE asks the context to use buffers with float samples. FALSE asks the context to use buffers with 16bits signed int samples.

MaxAuxSend: integer;
Number of auxiliary SEND per sound - default 2 (max 6).

MixMode: TALSPlaybackContextOutputMode;
The mode to use for the final mix for a playback context. Default value is ALC_STEREO_BASIC

HRTFIndex: integer;
The index of the HRTF to use for the playback context. This index points to an HRTF name in the list provided by TALSPlaybackContext.HRTFList. Default value is -1, that means default HRTF will be used. Please, see HRTFDemo for an typical usage.

procedure InitDefault;
Initialize all fields to their default value. Don't forgot to call first !

procedure SetLoopbackMode(aSampleRate: integer; aChannels: TALSLoopbackChannel; aSampleType: TALSLoopbackSampleType);
Initialize parameters for a loopback context.

TALSPlaybackContext - class

Playback context allow to load sounds, create effects,

Before playing sounds we must creates a playback context with :

function ALSManager.CreateDefaultPlaybackContext : TALSPlaybackContext ;
or

function CreatePlaybackContext(aNameIndex: integer; const aAttribs: TALSContextAttributes): TALSPlaybackContext;

> NOTE :If your application need more than one playback context at the same time, you have to define a custom option : on Lazarus go to menu Project Option → Custom options → Defines... → enter : ALS_ENABLE_CONTEXT_SWITCHING → push Add → then check the option. This enable the use of multiple playback context at the same time.
> When multiple playback context are enabled, ALSound make an intensive use of critical section that can impact the application speed.

Context attributes - record

Loopback context

Capture context

Enum – Record

TOALSState

Possible state for sounds, playlist and capture context

```
TOALSState = (
        OALS_STOPPED,
        OALS_PLAYING,
        OALS_PAUSED,
        OALS_RECORDING );
```

TOALSPlaybackMixMode

Playback context output mode.

```
TOALSPlaybackMixMode = (
        ALC_SURROUND_5_1 = openalsoft.ALC_SURROUND_5_1_SOFT,
        ALC_SURROUND_6_1 = openalsoft.ALC_SURROUND_6_1_SOFT,
        ALC_SURROUND_7_1 = openalsoft.ALC_SURROUND_7_1_SOFT,
        ALC_ANY_SOFT     = openalsoft.ALC_ANY_SOFT,
        ALC_STEREO_BASIC = openalsoft.ALC_STEREO_BASIC_SOFT,
        ALC_STEREO_UHJ   = openalsoft.ALC_STEREO_UHJ_SOFT,
        ALC_STEREO_HRTF  = openalsoft.ALC_STEREO_HRTF_SOFT );
```

TOALSDistanceModel

Distance model
For more explanation see https://indiegamedev.net/2020/04/12/the-complete-guide-to-openal-with-c-part-3-positioning-sounds/

```
TOALSDistanceModel = (
        AL_NONE, // no distance attenuation
        AL_INVERSE_DISTANCE,
        AL_INVERSE_DISTANCE_CLAMPED,
        AL_LINEAR_DISTANCE,
        AL_LINEAR_DISTANCE_CLAMPED,
        AL_EXPONENT_DISTANCE,
        AL_EXPONENT_DISTANCE_CLAMPED );
```

TOALSEffect

```
 TOALSEffect = record
    Ready: boolean;
    procedure UpdateParameters( P: Pointer );
    // Redirects the effect output to the input of another.
    // This allow you to chains multiple effects on a single auxiliary send.
    // Returns True if success.
    function ChainWith( const aTargetEffect: TOALSEffect ): boolean;

    // the output gain of the effect. Range is [0..1]
    property OutputGain: single read FOutputGain write SetOutputGain;
    // Enable or disable automatic send adjustments based on the physical
    // positions of the sound and the listener
    property ApplyDistanceAttenuation: boolean read FApplyDistanceAttenuation write
SetApplyDistanceAttenuation;
  end;
```