CPSC 5910 Graphics/Game Project

Seattle University

SQ 2020

Ross Hoyt

**FMOD Audio Engine with Visual Studio 19: Overview & Installation**

FMOD is a powerful cross-platform audio middleware solution for games and audio applications. It provides a range of audio tools for C++ developers, from simple playback of nearly any audio file format, to 3D positional audio, and highly interactive, parameter-controlled soundscapes, music scores, and more. Additionally, it is free to use in any game project with a budget under $500,000. More licensing information is at <https://www.fmod.com/licensing>

For the purposes of this course, I have created an Audio Engine class in C++ which wraps around the FMOD API, providing easy access to FMOD’s functionality without having to deal with its details. Whether you’re using FMOD directly or via the Audio Engine class, you’ll need to add FMOD to your Visual Studio project:

**FMOD Setup Directions**

**1. Obtain FMOD’s .dll files, .lib files, and header files.** These are included in this repository, or can optionally be downloaded by creating an account at <https://www.fmod.com> and downloading/installing the ‘*FMOD Studio API*’ from <https://www.fmod.com/download>.

**2. Create a new sub-directory** in your VS Project’s *Include* folder, titled ‘*FMOD’* [e.g: */Include/FMOD/]*. Place the FMOD header files inside this new *FMOD* sub-directory.

**3.** **Determine the libraries** you need to add to your project, based on your VS Project’s existing build configuration. Choose either x64 or x86 (Win32) dependencies; you cannot use both in the same VS project. Then, place the chosen libraries directly into your project’s library directory (/Lib).

**4. Add the new dependencies to Debug and Release Configurations of VS ‘Project Properties’ window:** First, select the Platform you are using (x64 or Win32). Then, go to text box ‘*Additional Dependencies’*, found under *Configuration Properties/Linker/Input*. Set Configuration to Debug, and paste the Debug dependencies into the *Additional Dependencies* text box, (text to paste is provided in the chart below). Apply it, and then switch Configuration to Release. Paste the Release dependencies from below chart into the text box. Press apply again.

|  |  |  |
| --- | --- | --- |
| **Configuration:** | **Debug** | **Release** |
| **Paste into ‘Additional Dependencies’:** | fmodL\_vc.lib;fmodstudioL\_vc.lib; | fmod\_vc.lib;fmodstudio\_vc.lib; |

**5. Tell Visual Studio where to find your Debugging Environment.** In your project’s Properties window, set Configurations to ‘All Configurations’ and set Platform to ‘All Platforms’. Then, go to the *Environment*  text box, which can be found under *Configuration Properties/Debugging,* and define the path where Visual Studio can find your Library (/Lib) folder, relative to your project’s directory.

Here is how to determine what text to add into the *Configuration Properties/Debugging/Environment* text box:

If your ‘Lib’ directory is one directory back from your VS solution’s directory, like this directory structure:

*Game Project/Visual-Studio-Stuff/your-projects-VS-solution.sln*

*Game Project/Lib/fmod.dll, glfw.lib, etc*

…You would add *this* text to the *Environment* text box: **PATH=%PATH%;$(ProjectDir)\..\lib**

Or, if you have your Lib directory in the *same* directory as your Visual Studio Solution (.sln) file, you would add: **PATH=%PATH%;$(ProjectDir)\lib**

*(Note: This step (#5) will likely need to be repeated in each local repository of a Visual Studio project, if using Git. For some reason, this setting does not get tracked in Visual Studio Project when pushing to remote Git repository.)*

You’ve now setup FMOD for use in your project! You can directly include the FMOD source files in your code, e.g. *#include <FMOD/fmod.hpp>* .

**6. (Optional) Add the AudioEngine.h and AudioEngine.cpp files to your Visual Studio Project**, and then you can #include “AudioEngine.h”. You can construct an AudioEngine object in your main file, and use it to quickly bring a flexible sound implementation into your game. The Audio Engine class provides straightforward sound loading and playback capability, and supports 3D positional audio, variable listener position and sound source position. Support for FMOD Sound Banks is also in development.

Here is a simple example of how you could use AudioEngine.h to control in-game sound:

#include "Audio-Engine/AudioEngine.h"

AudioEngine audioEngine;

SoundInfo soundLoop2D (MUSIC\_2, true);

int main()

{

// Setup OpenGL, GLFW, etc...

// Initialize Audio Engine and Load sounds

audioEngine.init();

audioEngine.loadSound(soundLoop2D);

// Play looping music to start scene

//

audioEngine.playSound(soundLoop2D);

/\* OpenGL loop \*/

while (!glfwWindowShouldClose(window))

{

// Render your scene…

// set current player position each frame, if you are using 3D sounds

audioEngine.set3DListenerPosition(camera.Position.x, camera.Position.y, camera.Position.z,

camera.Front.y, camera.Front.x, camera.Front.z,

camera.Up.y, camera.Up.x, camera.Up.z );

audioEngine.update(); // per-frame FMOD update

// Other OpenGL code

}

}

If you have any questions or issues with installing FMOD or using the Audio Engine, feel free to send an email to [hoytross@seattleu.edu](mailto:hoytross@seattleu.edu).

Thanks for reading!