Tutorial : Audio Plugin in C++ (JUCE)

ULG, Belgium, 2016

SCHMITZ Thomas

Department of Electrical Engineering and Computer Science,
University of Liège

10 juin 2016

Summary

1. Overview

2. Installation

3. A Multi Channels Delay Example

Section 1

Overview

After reading this tutorial, you should be able to :

- Install the development environment.
- Create new projects in **Projucer**.
- Export this project to your native IDE such as Xcode on OSX, or Visual Studio on Windows.
- Compile, run, and debug your app in the native IDE for the platform(s) you want to target.
- Create VST,VST3,AU,RTAS plugin for audio host.

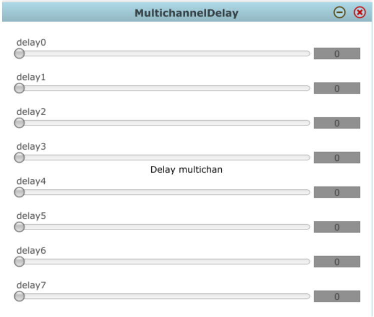


Figure: Multichannel delay example

Section 2

Installation

Software requirements

In order to create your own audio plugin, you need :

- An IDE : Visual Studio (free for student,Windows), Xcode (MAC), CodeBlock (Free, Windows, Linux)
- The library **JUCE**
- The **VST Audio Plugin SDK** (to create VST plugin
- The **ASIO SDK** (to run host debugger with ASIO drivers)
- The **source code** for this tutorial example

Note : in this tutorial i use **Virtual Studio 2015** which is free for student

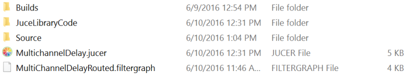
Directories

Directories location example :

- Install your IDE in $C:\backslash Program\ Files$
- Install Juce in $C:\backslash Program\ Files\backslash JUCE$
- Install the SDK ASIO and the SDK VST in $C:\backslash SDKs$
- Install the source code tutorial example where you want (i have put it in $\backslash E:\backslash Juce\backslash MultichannelDelay$)

Section 3

A Multi Channels Delay Example



Builds	6/9/2016 12:54 PM	File folder	
JuceLibraryCode	6/10/2016 12:31 PM	File folder	
Source	6/10/2016 1:04 PM	File folder	
MultichannelDelay.jucer	6/10/2016 12:31 PM	JUCER File	5 KB
MultiChannelDelayRouted.filtergraph	6/10/2016 11:46 A...	FILTERGRAPH File	4 KB

FIGURE: Multichannel delay folder example

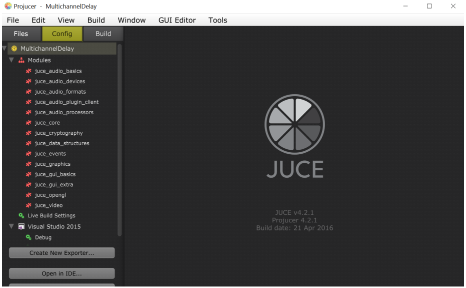


FIGURE: Projucer

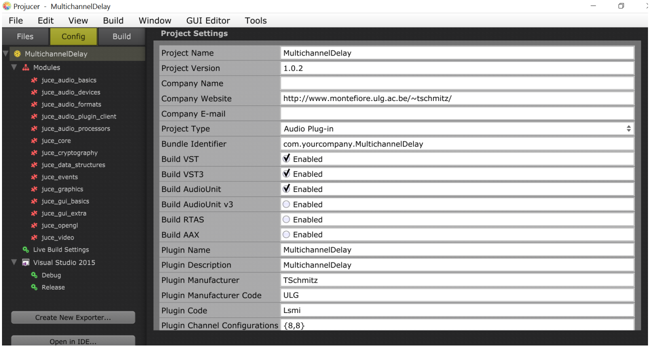
Multichannel delay example

- Open the multichannel delay example by clicking the *MultichannelDelay.jucer* file.
- It opens Projucer who make your setting project for your IDE.
- if you don't use *Visual Studio 2015*, you have to start a new project (audio plugin) following **this tutorial** and replacing the file generated in the source directory by those provided in my example.

Setting in Projucer

Multichannel delay example

- On the *Config* tab, click on the name of the project and verify the configuration as depicted in the following figure
- The Plugin channel Configuration is set to 8,8 (8 inputs and 8 outputs), put the number of entry that your sound card have, for example 2,2 for a standard sound card.



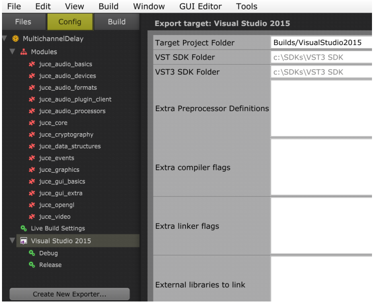
Project Name	MultichannelDelay
Project Version	1.0.2
Company Name	
Company Website	http://www.montefiore.ulg.ac.be/~tschmitz/
Company E-mail	
Project Type	Audio Plug-in
Bundle Identifier	com.yourcompany.MultichannelDelay
Build VST	☑ Enabled
Build VST3	☑ Enabled
Build AudioUnit	☑ Enabled
Build AudioUnit v3	○ Enabled
Build RTAS	○ Enabled
Build AAX	○ Enabled
Plugin Name	MultichannelDelay
Plugin Description	MultichannelDelay
Plugin Manufacturer	TSchmitz
Plugin Manufacturer Code	ULG
Plugin Code	Lsmi
Plugin Channel Configurations	{8,8}

FIGURE: Projucer config

Setting in Projucer 2

Multichannel delay example

- In Projucer, click on your IDE and set the location of the VST SDK folder.
- Then click on Module and verify that all the module have the right path $C:\backslash Program\ Files\backslash JUCE\backslash modules$ if you have followed my advice on directories location.
- Now you can open the project in your IDE by clicking the button *Open in IDE...*



MultichannelDelay
- Modules
  - juce_audio_basics
  - juce_audio_devices
  - juce_audio_formats
  - juce_audio_plugin_client
  - juce_audio_processors
  - juce_core
  - juce_cryptography
  - juce_data_structures
  - juce_events
  - juce_graphics
  - juce_gui_basics
  - juce_gui_extra
  - juce_opengl
  - juce_video
- Live Build Settings
- Visual Studio 2015
  - Debug
  - Release

Export target: Visual Studio 2015

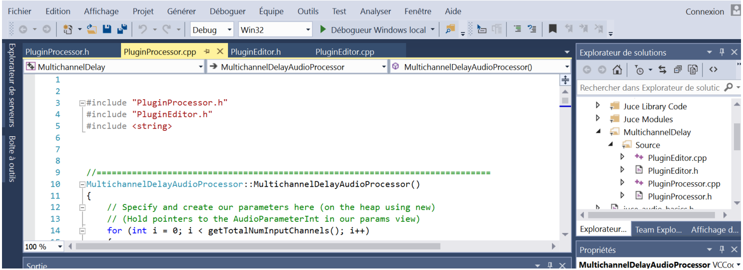
Target Project Folder	Builds/VisualStudio2015
VST SDK Folder	c:\SDKs\VST3 SDK
VST3 SDK Folder	c:\SDKs\VST3 SDK
Extra Preprocessor Definitions	
Extra compiler flags	
Extra linker flags	
External libraries to link	

FIGURE: IDE SDK location

IDE

Source files

- You need 4 files to start with your plugin
- PluginEditor.h PluginEditor.cpp, where you set the graphic interface of the plugin.
- PluginProcessor.h, PluginProcessor.cpp where you set the operation to do with the samples buffer coming from the host.



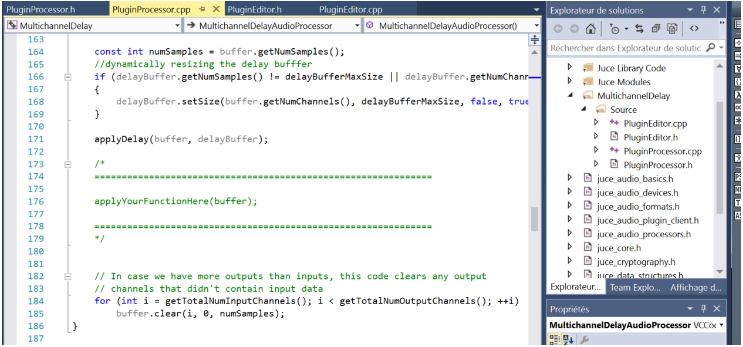
```


#include "PluginProcessor.h"
#include "PluginEditor.h"
#include <string>



//===============================================================================
MultichannelDelayAudioProcessor::MultichannelDelayAudioProcessor()
{
    // Specify and create our parameters here (on the heap using new)
    // (Hold pointers to the AudioParameterInt in our params view)
    for (int i = 0; i < getTotalNumInputChannels(); i++)
```

Figure: IDE source files

Adding operation on the input buffer

Source files

- The host send you a buffer of sample that you can process, here i have apply a delay function on each channel but you can add other function.
- the buffer can have multichannel and its size is set by your host (for example 256 samples)
- you can build and find a MultichannelDelay.dll (for vst) plugin in the build directory.
- follow **more tutorials** on the Juce website to understand the library.



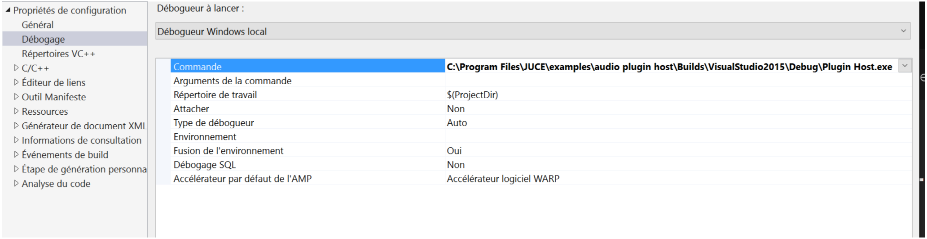
```
        const int numSamples = buffer.getNumSamples();
        //dynamically resizing the delay bufffer
        if (delayBuffer.getNumSamples() != delayBufferMaxSize || delayBuffer.getNumChann
        {
            delayBuffer.setSize(buffer.getNumChannels(), delayBufferMaxSize, false, true
        }

        applyDelay(buffer, delayBuffer);

        /*
        ==============================================================

        applyYourFunctionHere(buffer);

        ==============================================================
        */


        // In case we have more outputs than inputs, this code clears any output
        // channels that didn't contain input data
        for (int i = getTotalNumInputChannels(); i < getTotalNumOutputChannels(); ++i)
            buffer.clear(i, 0, numSamples);
    }
```

FIGURE: Add your own function here

Debugging your plugin

Use the little host of Juce

- Juce provide a host example that you can call during the debug mode
- In Visual Studio click on *debug* ,*property*, in the command option specify that you want to use the PluginHost.exe as in the figure below.



Débogueur à lancer :

Débogueur Windows local

Commande	**C:\Program Files\JUCE\examples\audio plugin host\Builds\VisualStudio2015\Debug\Plugin Host.exe**
Arguments de la commande	
Répertoire de travail	$(ProjectDir)
Attacher	Non
Type de débogueur	Auto
Environnement	
Fusion de l'environnement	Oui
Débogage SQL	Non
Accélérateur par défaut de l'AMP	Accélérateur logiciel WARP

FIGURE: Launch your plugin with the host example in debug mode

Debugging your plugin 2

Use the little host of Juce

- Now when you build it launch the host, you can import your plugin by clicking the right mouse button and select it. if you don't see it , click on *option* and then on *edit the list of the available plugins*.

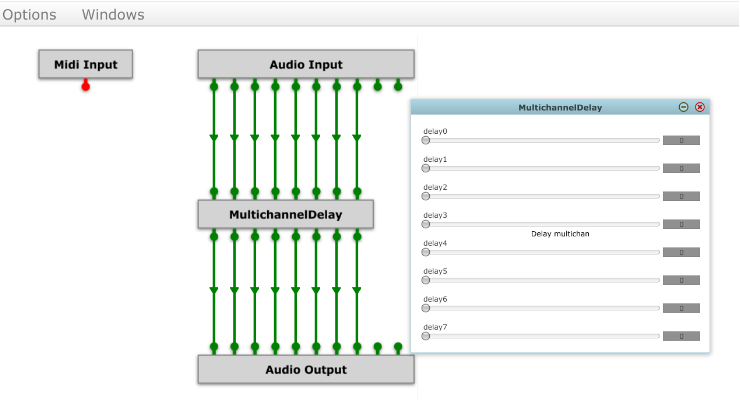


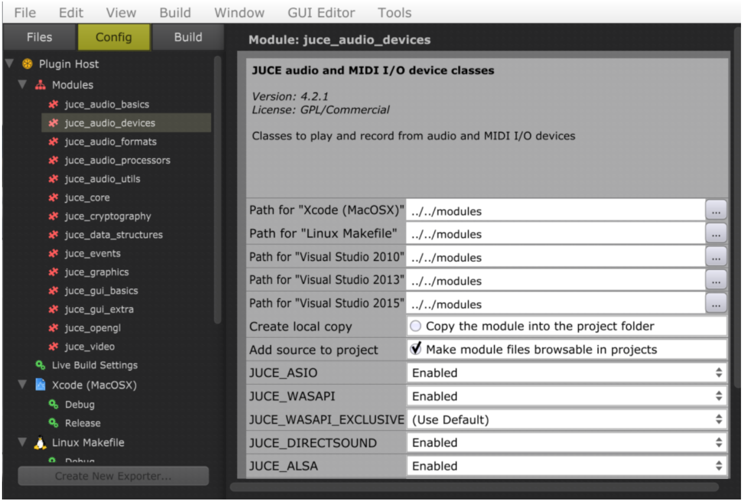
FIGURE: Connect your plugin to the inputs/outputs audio device

Host with ASIO driver

Build the host with ASIO enable

- By default, the host cannot use ASIO, you have to re-build the hostPlugin in the example directory by yourself.
- Open the PluginHost.jucer file in JUCE examples directory
- click on *juce_audio_devices*, and select *enabled* for *JUCE_ASIO*.
- Open the project in your IDE and find the line $\#include < iasiodrv.h >$ in the file *juce_audio_devices.cpp*, replaces it by the path to this *iasiodrv.h* in the ASIO SDK, (if you have followed my advice on directory location you should have :
  $\#include < C : \backslash SDKs \backslash ASIOSDK2.3 \backslash common \backslash iasiodrv.h >$

Host with ASIO driver 2



Module: juce_audio_devices

JUCE audio and MIDI I/O device classes

Version: 4.2.1
License: GPL/Commercial

Classes to play and record from audio and MIDI I/O devices

Path for "Xcode (MacOSX)"	../../modules
Path for "Linux Makefile"	../../modules
Path for "Visual Studio 2010"	../../modules
Path for "Visual Studio 2013"	../../modules
Path for "Visual Studio 2015"	../../modules
Create local copy	○ Copy the module into the project folder
Add source to project	☑ Make module files browsable in projects
JUCE_ASIO	Enabled
JUCE_WASAPI	Enabled
JUCE_WASAPI_EXCLUSIVE	(Use Default)
JUCE_DIRECTSOUND	Enabled
JUCE_ALSA	Enabled

FIGURE: Connect your plugin to the inputs/outputs audio device

Bibliography

- Juce Tutorials
  Juce documentation
- Steinberg VST SDK, ASIO SDK
- Audio Effects : Theory, Implementation and Application,Joshua D. Reiss