Project Report

Conor Holden

February 23, 2021

Abstract

Stuff about Pipes

Contents

1 Introduction

- Create a virtual instrument for the Uillean pipes
- not many there in advance
- Create a VST plug-in that can be used in digital audio work stations

2 Analysis

2.1 How Uillean Pipes Work

2.2 Simulating Instruments

2.2.1 Physical Modelling Synthesis

- Original Plan
- Mathematical Model of the Sound Wave (not simulation)
- Fairly easy on string instruments
- Uillean pipes relatively obscure
- out of scope for the project

2.2.2 Subtractive Synthesis

- Maynooth university thingy
- reasonable successful in pre-made synths
- would end up created a regular but limited subtractive syntheses

2.2.3 Additive Synthesis

- Sounds made up of many different sine waves
- Fast-Fourier transform
- need samples, too complex

2.2.4 Sampling

- simplest
- relies on quality of samples

2.3 Frameworks

There are a few different interfaces that allow plugins to communicate with digital audio workstations(DAW). The most popular is VST or Virtual Studio Technology, developed by Steinberg, a well know virtual instrument and audio application developer. It is a cross platform interface developed for Steinberg's DAW, Cubase and later licensed to other plugin and DAW creators. The vast majority of instrument and effect plugins you can buy today will either be in VST2 or VST3 format.

The second most popular is Audio Unit (AU), originally developed by Apple for their Logic Pro DAW and is now integrated into MacOS and IOS. Many DAWs on MacOS both support VST and AU, and they are, for the most part, interchangeable. It is possible to develop the instrument directly using these instruments, implementing everything yourself. However, for the purpose of this project, higher level frameworks will be used.

The first framework that was found is iPlug2. It is a free and open source framework based on the original iPlug released in 2008, improving on interface design and adding support for distributed plugin formats. IPlug provides many classes for digital signal processing used in audio applications and user interfaces which can be compiled to both VST and AU, as well as stand-alone applications and the new Web Audio Module, allowing you to host instruments online. However, being reactively new, released in 2018, there is not much documentation available and to understand what the code does, you would have to dig through the source code. For someone completely new to audio programming, the lack of documentation and tutorials makes creating a virtual instrument in a reasonable amount of time difficult.

The second framework is Juce and this is the one that was chosen. It is free to use for students, in open source projects and personally up to $50,000 in revenue and requires a license for other uses. As with iPlug, Juce can be compiled to many different plugin interfaces and a stand-alone application. What differs from iPlug is that Juce has much better documentation. There is an extensive number of classes that covers everything needed for audio programming and building user interfaces. Each class is documented and gives a good explanation of what each class does. On top of that, Juce is reactively popular in the audio programming community and there are numerous tutorials for this framework. The only downside is that these tutorials tend to be for the older Juce 5 when Juce 6 was released in June 2020 and the official tutorials used audio applications rather than plugins which have different formats.

3 Design

3.1 Plugin Layout

3.2 High Level

- Drone and chanter
- voices
- juce

3.3 Controls

- Enable, disable drone
- change key maybe

3.4 Samples

- 3 samplers, one for each components
- one sample can be pitched

4 Implementation

- classes
- juce plug-in layout
- Synthesiser and voices
- Param tree ==¿ parameter id all caps
- connection to parameter
- importing binary files
- buffers

4.1 Juce Plugin

2 sections

Plugin Editor

- UI
- buttons
- sliders
- Reference to processor

Plugin Editor

- UI
- buttons
- sliders
- Reference to processor

5 Evaluation

6 Conclusions

7 Appendix

8 Appendix 1 - How to use Juce

9 Appendix 1 - Music Example