Onenob Audio Processing Plugins with Csound and Cabbage

Joshua Daniel M C

March 2023

Abstract

Audio mixing is an essential and crucial part of music production, and the use of plugins has become increasingly essential to aid in the process. This project explores the development of a number of audio mixing Audio Unit (AU) plugins using the Csound programming language and the Cabbage framework. The aim of this project is to explore the various opcodes in Csound and gui capabilities of Cabbage to create visually appealing and user-friendly plugins that can be easily integrated into a digital audio workstation (DAW) workflow. Inspired by Waves OneKnob series plugins, One of the main aspect of these plugins is that they have a single knob to control the effect, making them beginner-friendly.

Keywords: CSound, Cabbage Audio, Plugins

Contents

1. Introduction

The audio mixing AU plugins developed in this project each serve a unique purpose in the process of audio mixing and processing. There are ten plugins in total (more to be added in the future), including an high-frequency booster, bass booster, compressor, distortion, filters, and ambient, among others. These plugins were created using the Csound programming language and Cabbage, an IDE for Csound , which provided an intuitive and easy-to-use interface. The plugins were designed to have a simple control interface with just a single knob to control the effect, making them beginner-friendly. The different opcodes available in Csound were explored and implemented in the plugins, allowing for a diverse range of effects to be created. By developing these plugins and evaluating their performance and usability, this project contributes to the development of high-quality and accessible audio mixing tools for musicians and producers.

2. Csound and Opcodes

Csound is a powerful and versatile programming language designed for sound synthesis and Audio processing(1)(2) It provides a wide range of built-in signal processing functions, known as opcodes, that allow users to create complex and intricate audio effects and sounds. These opcodes are the building blocks of Csound and can be used in combination with one another to create unique and customized audio effects.

Csound's opcodes cover a wide range of audio processing and synthesis techniques, including filtering, oscillation, modulation, and envelope shaping, among others. Each opcode has a specific function and set of parameters that can be adjusted to produce different sounds or effects.

The flexibility and versatility of Csound make it an ideal tool for audio production and experimentation. It is widely used in the music industry, film scoring, and sound design. The ability to develop custom opcodes also allows users to create unique and innovative audio processing techniques.

In this project, Csound and its opcodes were used to develop a series of audio mixing AU plugins. The exploration of different opcodes allowed for a diverse range of audio effects to be created and implemented in the plugins, showcasing the power and flexibility of Csound in audio processing and synthesis.

3. Cabbage Audio - IDE

Cabbage is an open-source audio programming environment that is built on top of the Csound audio synthesis language. It was created by Rory Walsh and is maintained by a group of developers and contributors from around the world(3)(4)

One of the key features of Cabbage is its ability to create cross-platform audio applications and plugins that can be used on multiple operating systems, including Windows, macOS, and Linux. This makes it an attractive option for audio developers who want to create applications that can be used across multiple platforms.

Cabbage's GUI toolkit is also a major strength of the framework. It provides a range of customizable widgets, including buttons, sliders, knobs, and graphs, which can be used to create visually appealing and interactive user interfaces. These widgets can be customized with different colors, shapes, and sizes, and can be scripted using Lua or Python, providing even more flexibility and customization options.

In this project, the Cabbage framework was used to develop a series of audio mixing AU plugins. The intuitive and user-friendly interface provided by Cabbage allowed for the creation of visually appealing plugins with a simple control interface. The GUI capabilities of Cabbage were leveraged to provide a seamless and intuitive user experience, making the plugins beginner-friendly and easy to use.

4. Onenob Audio Plugins

The Onenob series of plugins is a unique collection of 10 audio processing tools that aim to provide a simple and easy-to-use experience for users. These plugins are available in both mono and stereo versions, making them versatile and suitable for a wide range of audio projects. While many plugins offer multiple controls for different parameters, the Onenob series stands out by using only one knob for each plugin, allowing users to achieve quick and intuitive results.

The 10 plugins in the Onenob series are:

- Onenob Sparkle
- Onenob Deep Bass
- Onenob Mid
- Onenob HFilter
- Onenob LFilter
- Onenob Compress
- Onenob Delay
- Onenob Ambient
- Onenob Distort
- Onenob Weirdo

Each plugin offers a unique audio processing effect, including parametric equalization, high-pass and low-pass filters, compression, delay, reverb, distortion, and more. The simplicity of the Onenob series allows users to easily add depth, dimension, and character to their audio projects with just a twist of a knob. Although currently only available as Audio Units (AU) plugins for use in Apple Logic Pro X, these plugins can be conveniently exported as Virtual Studio Technology (VST) plugins to operate seamlessly on both Mac and Windows systems, providing maximum flexibility and compatibility to all audio producers and engineers. The components and the code can be downloaded from https://github.com/joshuadanielmc/Onenob-Mix-Plugins. The different opcodes used for each plugins are listed below.

Equalizer:

- **pareq:** This opcode is used in Onenob Sparkle, bass, and mid plugins to implement equalizer. It allows for boosting or cutting specific frequency ranges by adjusting the amplitude and width of the filters.

Filters:

- **bqrez:** This opcode is used in Onenob Hfilters and Lfilters plugins to implement a biquad resonant filter. It allows for cutting or boosting frequencies above or below a specified cutoff frequency and adding resonance to the filter.

Compression:

- **compress2:** This opcode is used in Onenob compress to implement a dynamic range compressor. It allows for reducing the dynamic range of an audio signal by attenuating the peaks while boosting the quieter sections of the signal.

Delay:

- **vdelay:** This opcode is used in Onenob delay to implement a variable delay line. It allows for delaying the input signal by a specified amount of time, with adjustable feedback and cross-feedback parameters.

Reverb:

- **freeverb:** This opcode is used in Onenob ambient to implement a freeverb algorithmic reverb. It allows for simulating the acoustic properties of different spaces by modeling the behavior of sound reflections and absorption.

Distortion:

- **distort:** This opcode is used in Onenob distort to implement a distortion effect. It allows for adding harmonics to the input signal by increasing the amplitude of its nonlinear components.

Weirdo:

- **vdelay:** This opcode is used in Onenob weirdo to create a unique and interesting sound by applying extreme settings to the variable delay line. It allows for creating echoes and resonances that can be blended with the dry signal using a mix knob.

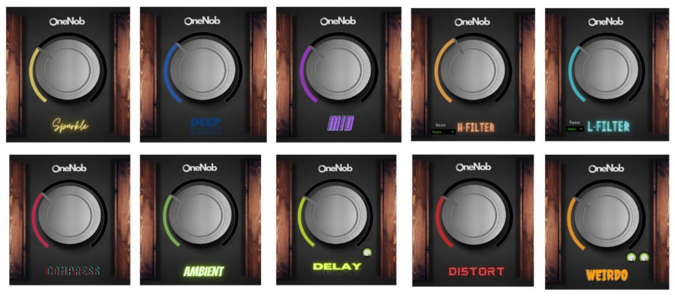

Figure 1: working screenshot of the onenob plugins developed using CSound and Cabbage framework

5. Summary and conclusions

The development of ten audio processing plugins using Csound and the Cabbage framework has demonstrated the simplicity and the potential of the Csound programming language and Cabbage framework in creating high-quality and visually appealing audio plugins. The project successfully utilized a single knob control interface, which makes the plugins user-friendly and accessible to beginners, allowing them to quickly and easily add effects to their audio tracks.

Throughout the development process, the exploration of various opcodes has provided a deeper understanding of their potential applications in audio mixing. The opcodes allows users to create efficient processing with simple programming. This has allowed the plugins to be created with unique features that can achieve different sound effects and enhance the overall quality of the audio.

Overall, the project has highlighted the power and versatility of Csound as a programming language for audio processing and Cabbage framework for GUI and demonstrated its potential in the development of audio plugins. In the future, it would be interesting to further exploration of integration of the developed plugins into different digital audio workstation (DAW) workflows to determine their compatibility and effectiveness in a professional audio production setting. Additionally, exploring more complex control interfaces could offer new and exciting ways to manipulate audio, expanding the capabilities of Csound-based audio plugins even further.

6. Acknowledgements

I would like to express my sincere gratitude to my professors, Dr. Iain McCurdy and Prof. Victor Lazzarini, for their invaluable guidance, support, and mentorship throughout the course of this project. I am grateful to Dr. Iain McCurdy for inspiring me with his examples in Cabbage and for providing me with the necessary resources and feedback to complete this project successfully. His encouragement and willingness to share his expertise have been crucial to my growth as a csound programmer. I would also like to extend my thanks to Prof. Victor Lazzarini for his guidance and support, as well as his passion for teaching and his dedication to ensuring that his students receive the best education possible.

7. References

[1] Victor Lazzarini, Steven Yi, Joachim Heintz, Øyvind Brandtsegg, Iain McCurdy, et al. *Csound: A sound and music computing system*. Springer, 2016.

[2] Csound Community. Csound - sound and music computing system, 2023. Accessed: April 6, 2023.

[3] Rory Walsh. Audio plugin development with cabbage. In *Proceedings of the Linux Audio Conference, Maynooth, Ireland*, pages 47–53, 2011.

[4] Rory Walsh. Developing csound plugins with cabbage. In *Ways Ahead: Proceedings of the First International Csound Conference*, pages 64–82.

[5] ALEX HOFMANN and IAIN MCCURDY. Collaborative documentation of open source music software: The csound floss manual. In *Ways Ahead: Proceedings of the First International Csound Conference*, page 282. Cambridge Scholars Publishing, 2013.