Audio Waveform Visualizer

Brian Hoepelman

Jonel Roman

GitHub: <https://github.com/jrlx/Visualizer>

# Overview

## Introduction

This project aims to create an interactive application that takes an audio track and converts it into visual waveforms. It was developed using C++ with the help of several libraries, namely ImGui (using ImGui-SFML to integrate with SFML) for the user interface, SFML for audio processing, playback, and visualization, and NativeFileDialog-Extended for file selection. The primary objective of this project was to build a user-friendly tool that enables users to visualize audio data in a graphical manner, using C++ concepts such as object-oriented programming (OOP).

## Background

The visualization of audio data as waveforms can provide deeper insights into audio characteristics such as volume, frequency, and timing, not to mention it looks cool. ImGui is a well-suited library for building graphical interfaces in a simple manner while SFML offers an efficient way to handle multimedia elements, such as audio playback and processing. NativeFileDialog-Extended was integrated to provide an easy method for users to load audio files into the application.

# Technical Details

## Implementation of OOP Concepts

* **Encapsulation**: The logic of the application was encapsulated into several classes. The main process that handles all other processes is in the Application class, the creation of the controls is handled by the UIManager class, the class AudioPlayer handles audio playback, and the visualization itself is handled by the WaveformVisualizer class.
* **Inheritance**: The WaveformVisualizer class inherits from a base Visualizer class, which could be extended in the future to handle different types of visualizations.
* **Polymorphism**: Virtual functions were used in the Visualizer class, allowing different types of visualizations (like waveform vs spectrogram) to be implemented and chosen at runtime.

## Other C++ Topics

* **Standard Library**: The project uses the C++ standard library, using std::vector to store waveform vertices, and iostream to output to the console.
* **Exception Handling**: The project uses std::exception and std::runtime\_error alongside try – catch statements, in order to throw and catch potential errors.
* **Constructors and Deconstructors**: The Application class has a constructor that initializes the window and ImGui, and a deconstructor to shutdown ImGui. Some of the other classes use constructors and deconstructors as well.

# Analysis and Reflection

## Challenges and Solutions

* Initially, we struggled setting up ImGui-SFML, ImGui, and SFML to work together. We ended up implementing a CMake configuration to automatically fetch the files and link them together.
* We struggled a bit implementing the waveform visualization. Problems such as miscalculated indices that accessed incorrect samples and inconsistent scaling factors that resulted in a distorted waveform. We overcame these issues by consulting external resources, such as online videos and documentation, which helped us better understand the nuances of audio processing and waveform scaling. Additionally, constant debugging and testing were performed to ensure correctness.

## Future Improvements

* **Display More Information**: Enhance the visualization by displaying additional information such as the number of samples, sample rate, bit depth, and audio duration. This could provide users with a deeper understanding of the audio file’s properties.
* **User Customization**: Another potential extension is to allow users to customize the waveform, such as changing colors, scaling, or applying different filters to visualize specific frequency ranges.

# References

## Libraries

* **SFML**: <https://www.sfml-dev.org/>
* **ImGui**: <https://github.com/ocornut/imgui>
* **ImGui-SFML**: <https://github.com/SFML/imgui-sfml>
* **NativeFileDialog-Extended**: <https://github.com/btzy/nativefiledialog-extended>

## References

* <https://www.sfml-dev.org/documentation/2.6.2/classsf_1_1VertexArray.php>
* <https://www.sfml-dev.org/tutorials/2.6/audio-sounds.php>
* <https://www.youtube.com/watch?v=Xdbk1Pr5WXU>
* <https://www.youtube.com/watch?v=bqGjh67x7IU>
* <https://cplusplus.com/reference/memory/unique_ptr/>
* <https://edw.is/using-cmake/>