Cassettes – An Online Audio Editor

Xingxing Yang
CCRMA
digdongaa@gmail.com

Jatin Chowdhury
CCRMA
Jatinchowdhury18@gmail.com

ABSTRACT

Cassettes is a project that started in 2018 summer. The motivation is to make an online audio editor in the browser, like an online version of Audacity. Through several months' development, now we can use it to record, edit, play and download audio. It is currently working well in Chrome.

1. INTRODUCTION

In this demo, we present Cassettes, which is an online audio editor. Compared to other online DAWs, it is mainly for audio editing. Users can easily adjust gain (fade in/out/buffer gain), drag and drop audio files cross tracks, cut buffers, glue buffers, adding audio effects, choose area to download, etc.

Its interface is as follows:

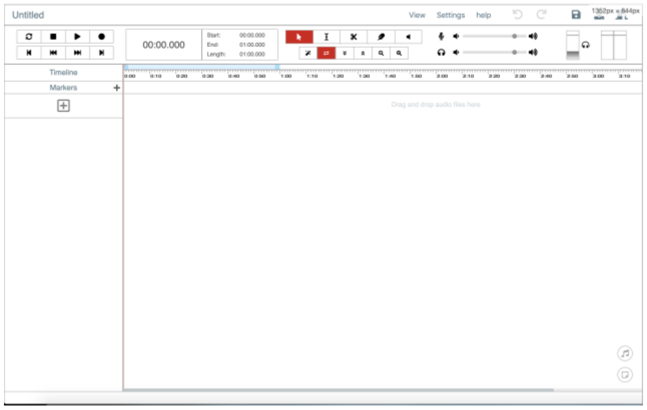

FIGURE 1: The Cassettes Interface

2. RELATED WORK

There have already been many online DAWs.

The open-sourced ones include opendaw, waveform-playlist, etc.

The commercial ones include Soundtrap, Bandlab, Soundation, etc. Most of them are mainly used for music production, while they may also be used for podcast/audio production.

The project is originally inspired by waveform-playlist. The limitation of waveform-playlist is that the interactivity is not so mature.

3. IMPLEMENTATION

3.1 Software design and architecture

The architecture mainly consists of 2 parts, the user interface and the audio engine. To make them communicate, some essential data is stored.

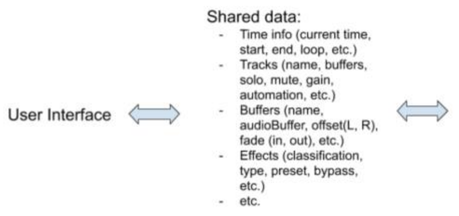
User Interface
Shared data:
- Time info (current time, start, end, loop, etc.)
- Tracks (name, buffers, solo, mute, gain, automation, etc.)
- Buffers (name, audioBuffer, offset(L, R), fade (in, out), etc.)
- Effects (classification, type, preset, bypass, etc.)
- etc.
Audio engine

FIGURE 2: Architecture

The audio engine responds to user interaction by shared data. (We use React, so our shared data is mainly stored in a Redux store.)

3.2 Parts

The audio editor can be divided into several parts, including timeline, waveform, tracks, buffers, audio effects, audio libraries and download, etc. In the following sections, we will describe the implementation of waveform, audio effects and download.

3.2.1 Waveform

Previous work for browser-based audio waveform visualization includes waves-ui by IRCAM and peaks.js by bbc R&D. Our implementation of waveform visualization partially refers to their work.

For drawing the waveform, the goal is to display the waveform instantly as the audio buffer is ready and to respond quickly as the user zooms in and out while keeping the waveform looking comfortable to users. To achieve this, we used technologies including canvas, web workers and html img.

As users initially import the audio file, a canvas is drawn by extracting the max and min values for every finite length audio samples depending on the zoom level (e.g. 1024 samples for zoom level 3). In the meantime, a web worker is working to compute the canvas data and image for other zoom levels (11 is used in our case). For browsers supporting OffscreenCanvas (e.g. Chrome), img is used for further display as user zooms in and out, which is much faster than canvas when loading.

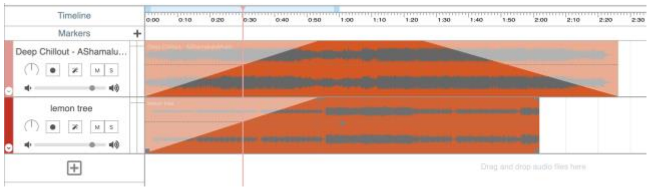

FIGURE 3: Waveform Overview

3.2.2 *Audio Effect Plugins*

In our design, each track has a port to link audio effect plugins. The master track also has a port to link audio effect plugins. This design is very common in desktop DAWs.

The main process of the audio engine for handling audio effects when the user hits play is as follows: creating effect => adding effect => setting effect params and bypass => setting effect automation. When the effect state changes during playing, the audio engine correspondingly responds.

The web audio API, and in particular the new Audio Worklet have made possible a whole world of web-based audio effects. Along with using built-in web audio effects, the low-level nature of the Audio Worklet have allowed us to fine-tune our own DSP algorithms to run in the browser, as well as providing outstanding performance even for the most complex of DSP algorithms.

Along with basic EQ, compressor, and channel strip effects that can be found in most DAWs, we have developed, a limiter, distortion, delay, chorus, flanger, and more. We also offer several reverb effects, including convolution and algorithmic reverb, plus a physical model of reverb inside a tube. Each parameter of these effects can be fully automated inside the audio editor.

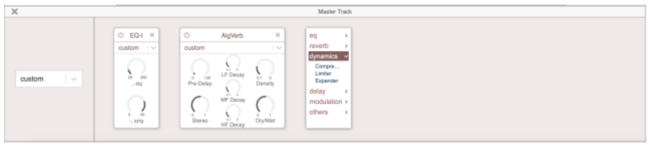

FIGURE 4: Audio Effect Plugins

Following is one of our plugins, which is written using web Audioworklet:

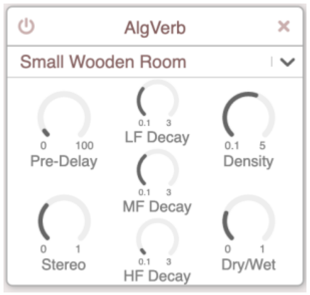

FIGURE 5: Algorithmic Reverb (Audio Effect Plugins)

For future advancements, we would like to continue expanding our effects library, adding effects including noise suppression, as well as potentially supporting third party plugins, allowing anyone to developing web audio effects to be able to use their effects inside our editor.

3.2.3 *Download*

In our design, users should choose the area they want to download first, then choose which format they want to use.

The download part is mainly achieved by OfflineAudioContext, which allows us to render the buffer quickly and finally to be available for encoding to other audio formats.

The audio encoding libraries we used include lamejs, libvorbis and fdkaac.

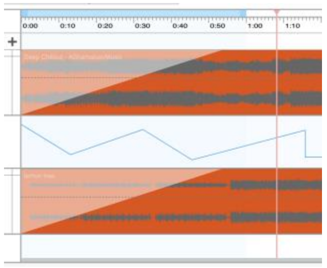

FIGURE 7: Choose The Area first

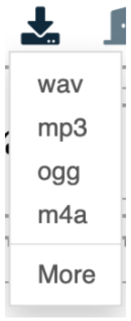

Figure 8: Click To Download

4. Future work

During our development, we mainly do test manually by listening to it. In the future, we'd like to add automated tests to save time.

We also want to add more features to it, such as video integration, third-party web audio plugins, text to speech generator, etc.

5. Links

The website can be accessible at https://cassettes.herokuapp.com.

6. REFERENCES

[1] Naomiaro - Waveform Playlist: https://github.com/naomiaro/waveform-playlist

[2] Peaks.js https://waveform.prototyping.bbc.co.uk/

[3] Web Audio API: https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API