DAWs application

Digital Audio Workstations (DAWs). DAWs are software applications designed for recording, editing, mixing, and manipulating audio and music.

Effects Implementation

The implementation of several audio effects was because the code uses p5.js. Here's a brief description of each:

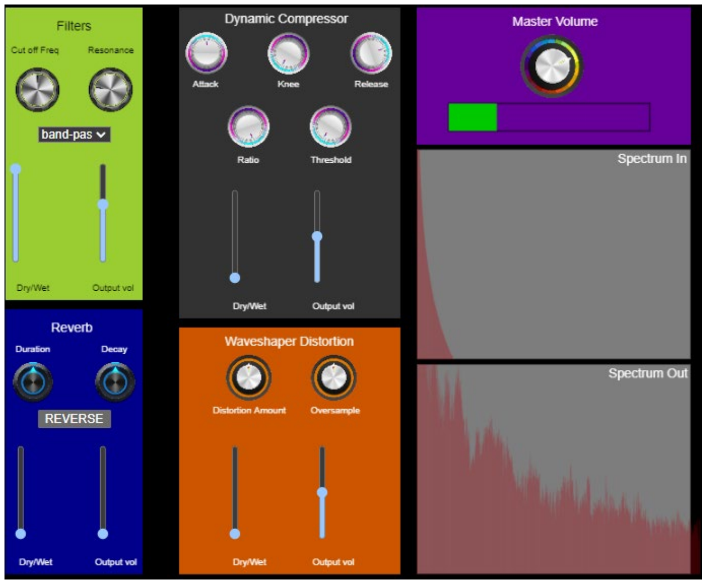
1. **Lowpass Filter:** This filter allows frequencies below a certain cutoff point to pass while attenuating frequencies above it. It's programmed to respond to changes from a dropdown menu and a knob for adjusting the cutoff frequency.
2. **Dynamic Compressor:** It reduces the dynamic range of the audio signal, making loud sounds quieter and quiet sounds louder. Attack, knee, release, ratio, and threshold parameters are adjustable via knobs and sliders.
3. **Distortion:** Adds harmonics to the sound, creating a 'fuzz' or 'crunch'. The amount of distortion and oversampling rate can be controlled with knobs.
4. **Reverb:** Gives the impression of space around the sound, simulating the sound reflections from surfaces. Duration and decay are adjustable, and it can be set to reverse.
5. **FFT (Fast Fourier Transform):** Analyzes and visualizes frequencies present in the audio signal. It's used here to create a dynamic visual representation of the sound spectrum.

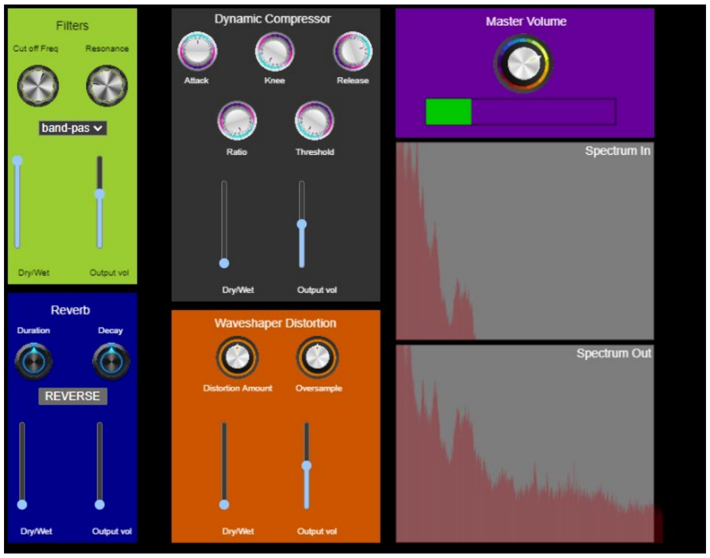
The effects are chained together in a signal path, and their parameters can be adjusted in real-time using the GUI controls.

Effects affecting the sound

Low-pass filter

Allows frequencies below a specific cutoff frequency to pass through, reducing or eliminating frequencies above this threshold. In a sound's spectrum, applying a low-pass filter will visually manifest as a tapering off or complete absence of the higher frequency components beyond the cutoff point. Visually, the high-frequency 'tail' of the spectrum will be reduced or cut off in the FFT display. In the next 2 images you can see how I change the low pass to have less and less frequencies

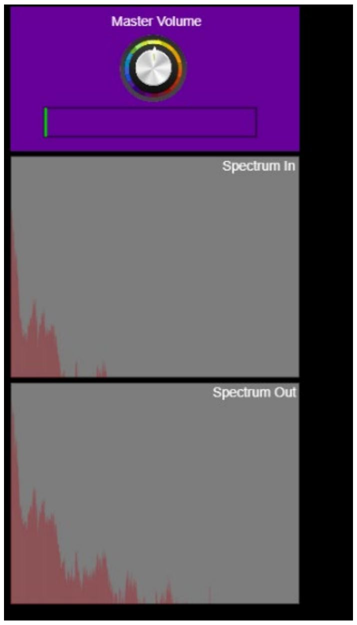


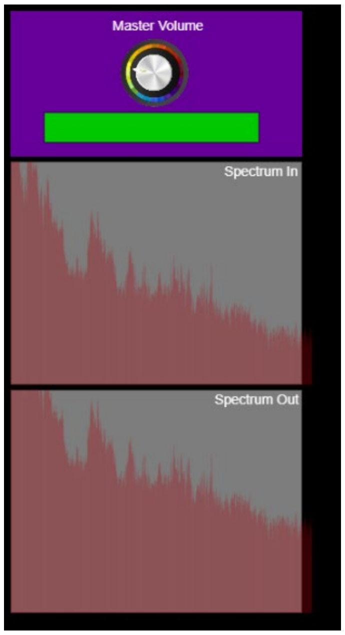


The master volume

The master volume affects the amplitude of the entire sound spectrum without altering the relative balance of frequencies. Increasing the master volume will raise the overall height of the spectrum display, indicating a louder sound, while decreasing the volume will lower the entire spectrum uniformly, making the sound quieter. The shape and balance of the frequency components remain the same; only their amplitude changes.

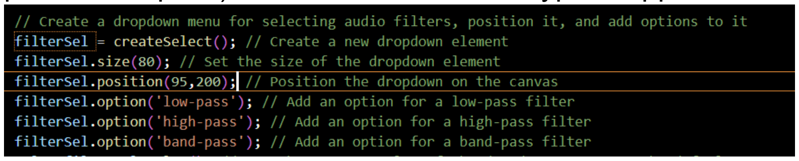
Example:





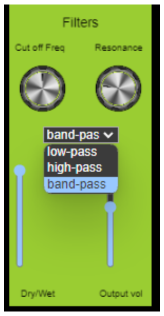
Selector of filters

The filterSel dropdown allows the user to select between different filter types (low-pass, high-pass, band-pass), and the selected filter type is applied in the filterChange() function.



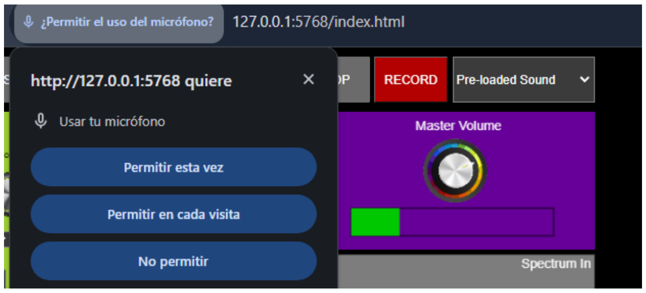
```
// Create a dropdown menu for selecting audio filters, position it, and add options to it
filterSel = createSelect(); // Create a new dropdown element
filterSel.size(80); // Set the size of the dropdown element
filterSel.position(95,200); // Position the dropdown on the canvas
filterSel.option('low-pass'); // Add an option for a low-pass filter
filterSel.option('high-pass'); // Add an option for a high-pass filter
filterSel.option('band-pass'); // Add an option for a band-pass filter
```

Then the function filterChange is changing the filter to the options displayed on the application



Microphone implementation

The **recInputChange** function allows for switching between a pre-loaded sound and live microphone input. The function stops the pre-loaded sound, start the microphone and set the effects into the live sound on the microphone.

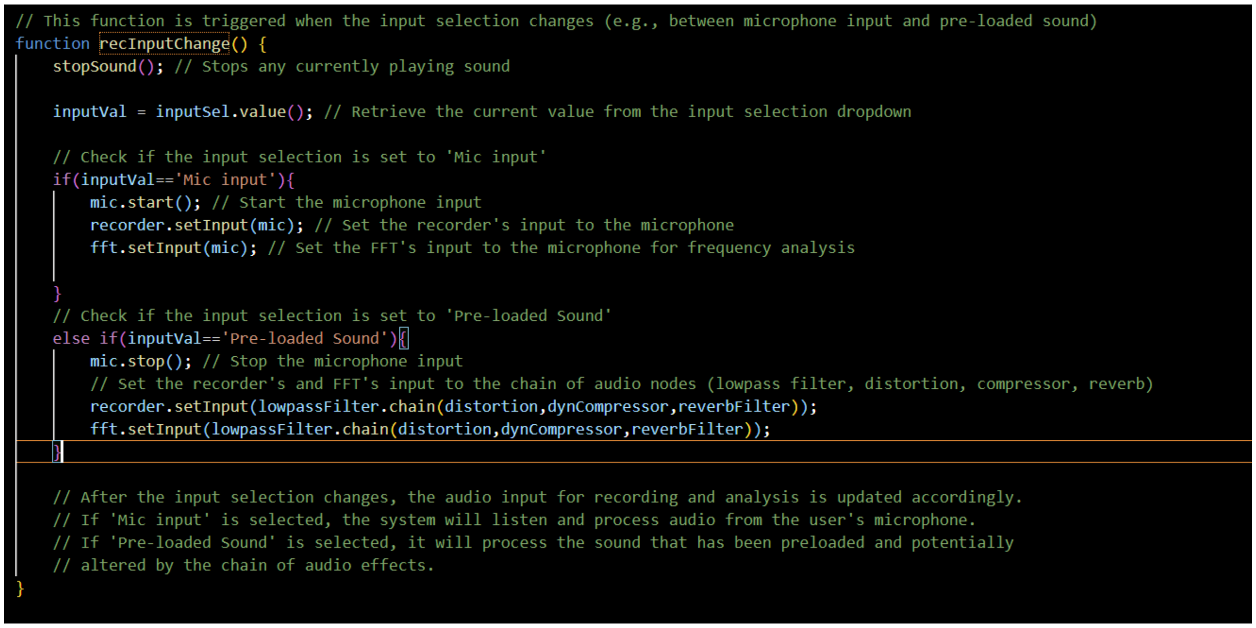


¿Permitir el uso del micrófono? 127.0.0.1:5768/index.html

http://127.0.0.1:5768 quiere

Usar tu micrófono

After the input selection changes, the audio input for recording and analysis is updated accordingly and will automatically download the file with the changes the application make.



```
// This function is triggered when the input selection changes (e.g., between microphone input and pre-loaded sound)
function recInputChange() {
    stopSound(); // Stops any currently playing sound

    inputVal = inputSel.value(); // Retrieve the current value from the input selection dropdown

    // Check if the input selection is set to 'Mic input'
    if(inputVal=='Mic input'){
        mic.start(); // Start the microphone input
        recorder.setInput(mic); // Set the recorder's input to the microphone
        fft.setInput(mic); // Set the FFT's input to the microphone for frequency analysis

    }
    // Check if the input selection is set to 'Pre-loaded Sound'
    else if(inputVal=='Pre-loaded Sound'){
        mic.stop(); // Stop the microphone input
        // Set the recorder's and FFT's input to the chain of audio nodes (lowpass filter, distortion, compressor, reverb)
        recorder.setInput(lowpassFilter.chain(distortion,dynCompressor,reverbFilter));
        fft.setInput(lowpassFilter.chain(distortion,dynCompressor,reverbFilter));
    }

    // After the input selection changes, the audio input for recording and analysis is updated accordingly.
    // If 'Mic input' is selected, the system will listen and process audio from the user's microphone.
    // If 'Pre-loaded Sound' is selected, it will process the sound that has been preloaded and potentially
    // altered by the chain of audio effects.
}
```