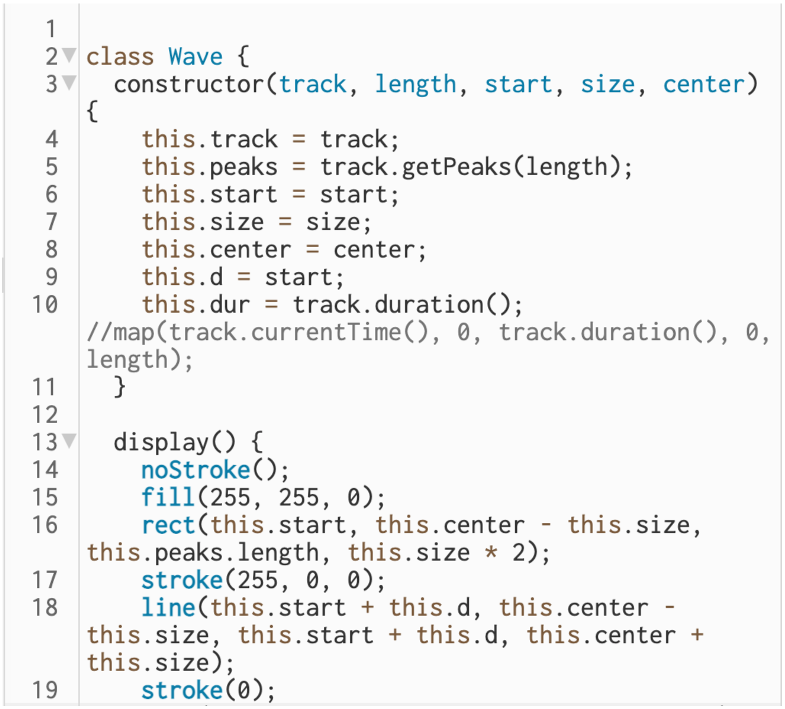
Final Project: Documentation

Link to Project: https://editor.p5js.org/alexacalder/sketches/6Bzes8j2d

For my final project, I rendered two spinning turntables, a crossfader, and waveforms that follow the two songs I uploaded into P5.js. Below each turntable is a Play/Stop button that stops each song individually. To run the turntable, it is quite simple. The music doesn't start automatically– the user has to click play for each (or either) of the records to begin the songs. Then, the user can use the crossfader to play around with the two songs, adjusting sounds and the flow between to two as much as they want. At the top of each record, a waveform will immediately begin to follow along with each song, showing at what point each song is at. On the records, a spinning line indicates to the user if the song is actively playing or if the song is paused. While seemingly simple, this project was a huge challenge for me as coding has always been quite challenging for me to learn. Rendering the waveforms and the crossfader were by far the most difficult for me but I was super proud of how they turned out in the end.

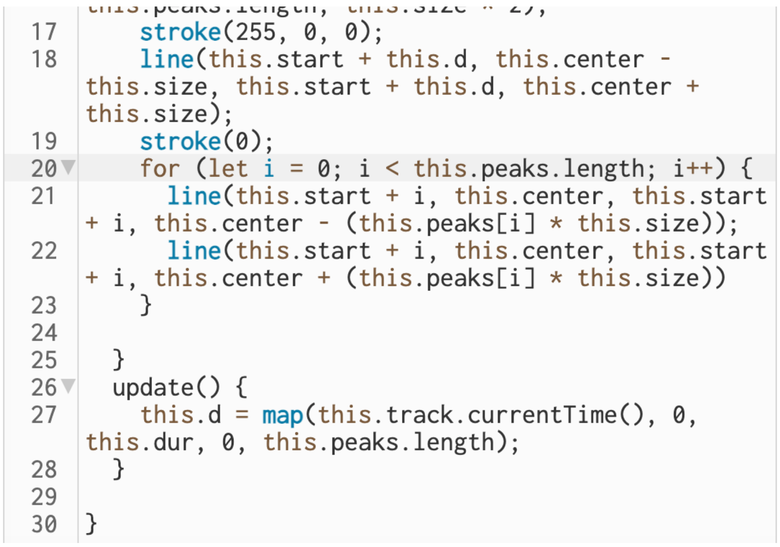
For what I consider the best part of my project, I'd have to say I'm most proud of my waveform class. Essentially, I had the waveform track and mimic each song's peaks, size, and length, in order to effectively render the waveforms. This took me a while to figure out, but this is the code.



```

class Wave {
  constructor(track, length, start, size, center) {
    this.track = track;
    this.peaks = track.getPeaks(length);
    this.start = start;
    this.size = size;
    this.center = center;
    this.d = start;
    this.dur = track.duration(); //map(track.currentTime(), 0, track.duration(), 0, length);
  }

  display() {
    noStroke();
    fill(255, 255, 0);
    rect(this.start, this.center - this.size, this.peaks.length, this.size * 2);
    stroke(255, 0, 0);
    line(this.start + this.d, this.center - this.size, this.start + this.d, this.center + this.size);
    stroke(0);
```



```
this.peaks.length, this.size * 2);
    stroke(255, 0, 0);
    line(this.start + this.d, this.center - this.size, this.start + this.d, this.center + this.size);
    stroke(0);
    for (let i = 0; i < this.peaks.length; i++) {
      line(this.start + i, this.center, this.start + i, this.center - (this.peaks[i] * this.size));
      line(this.start + i, this.center, this.start + i, this.center + (this.peaks[i] * this.size))
    }

  }
  update() {
    this.d = map(this.track.currentTime(), 0, this.dur, 0, this.peaks.length);
  }

}
```