WebAudio API:

- **Application in Web App:**
- Basic audio operations performed with audio nodes (interface)
- Audio operations run within audio contexts (interface)
- First we need to create an audio context
- Inside the audio context, we can create audio sources
- Then we choose the final destination of the audio - system speakers
- OscillatorNode
    - A periodic waveform acts as audio source based on the frequency we set for it (a constant tone)
    - Has one output - no input
    - Can de-tune (in cents units)
    - Can specify type of waveform to play (sine, square, sawtooth, triangle, custom, etc)
    - Can specify the time and length of how long to play tone
- Can easily create piano visualization: https://codepen.io/kbrammer/pen/Wdjday
- Source:
    - https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API

MediaStream Recording API:

- **Application in Web App:**
- Makes it possible to capture data for analysis, processing, or saving to disk
- Very easy to work with
- Major interface: MediaRecorder
- MediaStream represents stream of media content - audio tracks
- MediaRecorder object takes data from MediaStream and delivers to us
- MediaRecorder has start()/stop()/pause()/resume() methods for recording of audio
- When media is done being recorded, delivered in readable blob format
- The recorder gives events, which contain chunks of the recording - we need to push these chunks into an array
- Gather all chunks when recording stops to Blob to play with <audio> element
- Sources:
    - https://developer.mozilla.org/en-US/docs/Web/API/MediaStream_Recording_API

Django Highcharts:

- Charting Library (Javascript)
- Charting library works with any back-end database or server-stack
- Data can be provided to it to chart in any form (CSV, JSON, Python, R)
- Compatible with any browser + device
- Similar to the kind of charts we want to display:
  - https://www.highcharts.com/demo/pie-donut

Vexflow API:

- Music note rendering API written in Javascript
- Sample Code I've tested out:

```
const VF = Vex.Flow;

// Create an SVG renderer and attach it to the DIV element named "boo".
var vf = new VF.Factory({renderer: {elementId: 'boo', height: 400}});
var score = vf.EasyScore();
var system = vf.System();

system.addStave({
  voices: [
    score.voice(
      score.notes('D4/q, D4')
        .concat(score.beam(score.notes('D4/8, D4, D4, D4')))
    )
  ]
}).addClef('treble').addTimeSignature('4/4');


vf.draw();
```

- Generates: