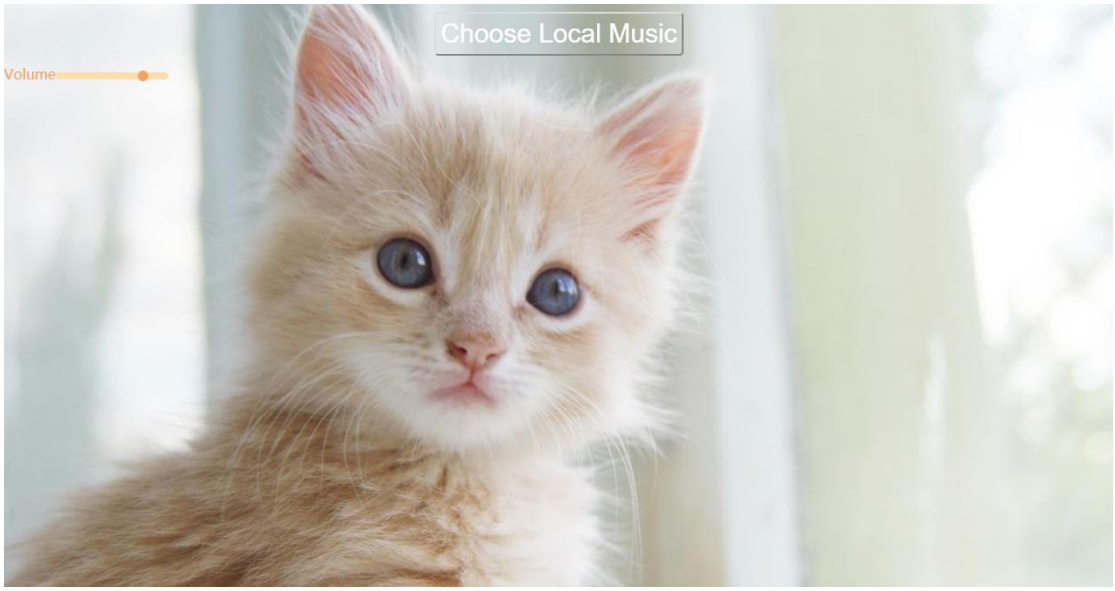
PJ1 音乐可视化报告

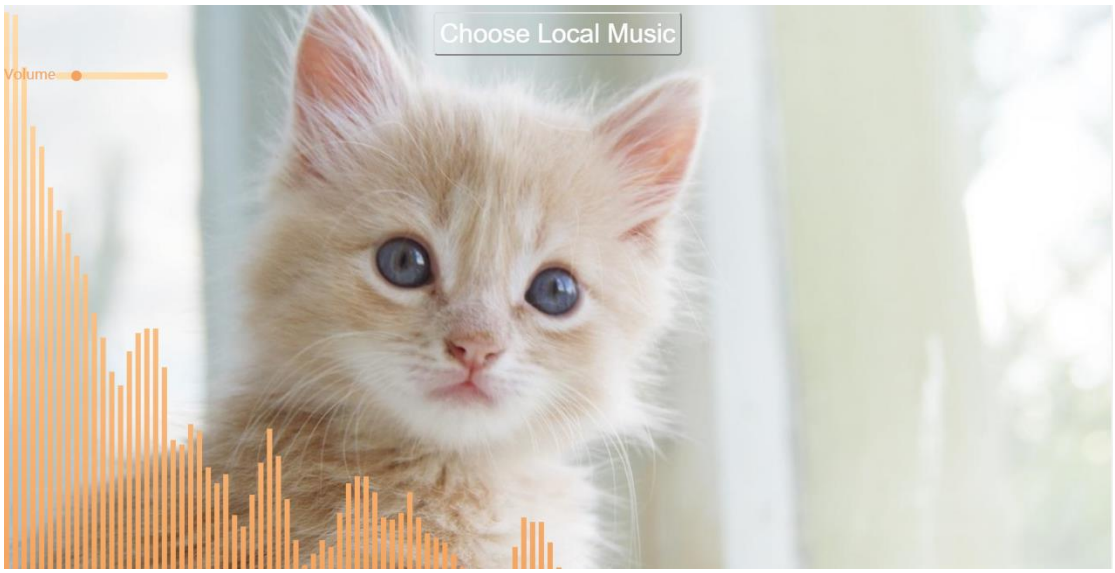
程序说明：

编程部分结合 html，JavaScript 和 css 进行。
可视化部分通过 Web Audio 和 Canvas（2D）实现。

点击 index.html 即可运行，运行主界面如图（选用了一张网络图片做背景）。



点击按钮上次本地音乐文件，即可播放，效果如图（具有调节音量功能）。



算法原理：

创建 AudioContext，作用是关联音频输入，对音频进行解码、控制音频的播放暂停等基础操作。

```
MusicVisualization.ac = new (window.AudioContext || window.webkitAudioContext)();
```

AnalyserNode 用于获取音频的频率数据（FrequencyData）和时域数据（TimeDomainData）。从而实现音频的可视化。

fftSize 决定了 frequencyData 的长度，具体为 fftSize 的一半。

```
this.analyser = MusicVisualization.ac.createAnalyser();
this.size = obj.size;
this.analyser.fftSize = this.size*2;
```

将音频节点，关联到 AudioContext 上，作为整个音频分析过程的输入。

```
        bufferSource.connect(self.analyser);
```

播放音频

```
var bufferSource = MusicVisualization.ac.createBufferSource();
// 解码成功后的buffer赋值给bufferSource的buffer属性
bufferSource.buffer = buffer;
```

获取频率数据。

```
function fn(){
    self.analyser.getByteFrequencyData(arr);
    self.draw(arr);
    requestAnimationFrame(fn);
}
```

最后，把频率数据映射为图形参数，这里简单地改变了每一个小矩形的高度和颜色。

```
function draw(arr){
    ctx.clearRect( x: 0, y: 0,width,height);  //清空上次画布内容
    ctx.fillStyle = line;
    var rectWidth = width/size;
    var cw = rectWidth*0.6;  // rectWidth*0.6的目的是保证矩形之间有间隙
    for(var i=0;i<size;i++){
        var o = Dots[i];
        var rectHeight = arr[i]/256*height;  //数据最大值为256
        // 小矩形（x,y,width,height）;
        ctx.fillRect( x: rectWidth*i, y: height-rectHeight,cw,rectHeight);
    }
```

参考文献：

1. 慕课- HTML5 音乐可视化
   https://www.imooc.com/learn/299
2. Web Audio 在音频可视化中的应用
   https://juejin.im/post/5d8c122be51d4578176b4b2b