# 【大结局】《从案例中学习JavaScript》之酷炫音乐播放器（四）

这是之前写的用H5制作的音乐播放器，前三节其实已经做得差不多了，音轨的制作原理已经在上一节说明，不过一直还没有和音乐对接。

本章作为该系列的一个完结篇，我会专门把动态音轨的实现代码贴出来，demo地址会在文章最后给出。

为了尽可能保持条理清晰，我就重新开一个页面来说明吧。你把本文的代码拷过去，应该是可以直接运行的。（当然，音乐文件需要换成你本地的）

## 1. 画一个demo页

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>音轨和音乐对接</title>
    </head>
    <body>

    </body>
</html>
```

在这个页面的head部分添加一个style块，我就不单独整css文件了，把所有的样式都写在一个页面吧。

给body添加一个线性背景

```
body {
    background:-webkit-linear-gradient(top,skyblue 0%,#fff 100%) no-repeat;
    background:linear-gradient(top,skyblue 0%,#fff 100%) no-repeat;
}
```

这个表示从上往下，上面是天蓝色，渐变到下面的纯白色。有一个过渡的效果。

界面：

---

从图中可以看到，页面上只有顶部有一条蓝色的线，这是因为目前的页面还没有任何东西。这个属性需要你body区域里有实实在在的东西，才能把高度给撑开。

当然，如果你用纯色，不使用 linear-gradient ，就没关系。

比如你直接写：

```
body {
    background:lightskyblue;
}
```

效果：

这样就没关系。

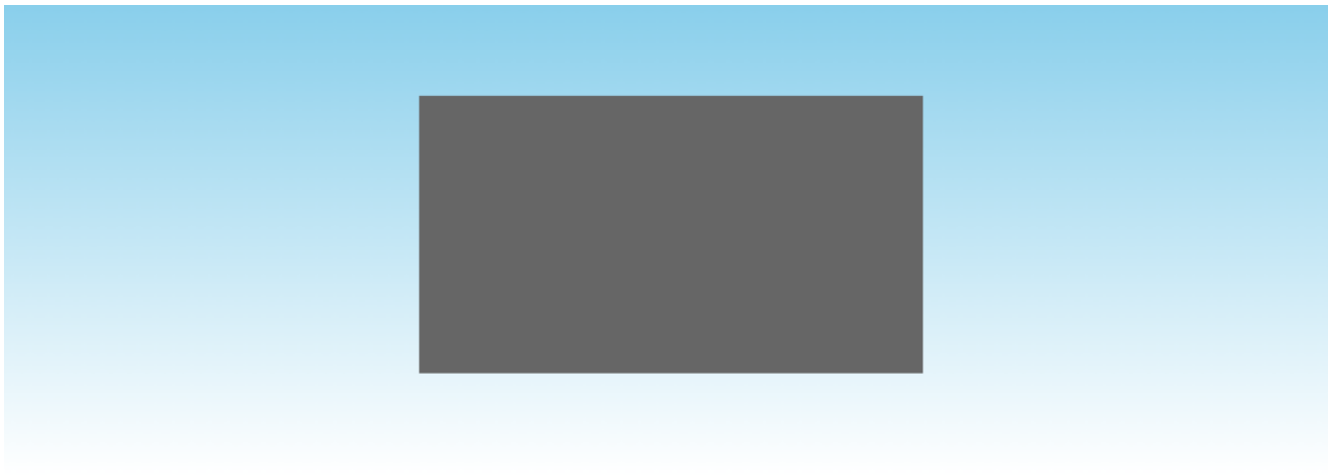
OK，继续，我们给body区域添加一个div，作为我们的音轨盒子。

```
<div class='box'></div>
```

设置一下他的css样式

```
.box {
    width: 500px;
    height: 275px;
    background: #666;
    position: relative;
    margin: 100px auto;
}
```

效果：



这样是不是就有了。

## 2. 音轨制作

我们还是采用span来制作音轨，具体原理和绘制方法已经在上一节中说明，这里不再赘述，我直接上代码了。

在body下方添加一个script块，里面就写我们的js代码：

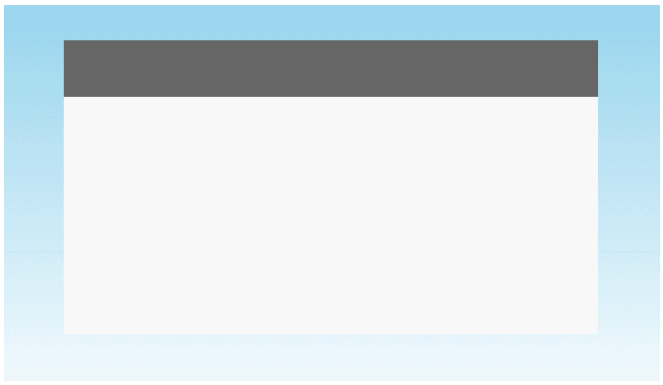
```
<script>

</script>
```

我们设置每一条音轨的宽度为10px。

```
var box = document.getElementsByTagName('div')[0];  //获取承载音轨的父盒子
var allWidth = box.clientWidth;//获取承载音轨盒子的宽度
var len = (allWidth / 10 );     //计算一共出现多少条音轨
var html = '';                  //动态拼接音轨
for(var i = 0;i < len ; i ++){
    html+="<span class='item' style='left:"+(i*10)+"px;background:#fff;'></span>";
}

box.innerHTML = html;           //添加音轨
```

我们先给每一个span添加一个高度，看看效果
css：

```
.item {
    position:absolute;
    width:10px;
    height:222px;
    left:0px;
    bottom:0px;
    opacity: 0.96;
}
```



然后，把span的高度设置为0。

好的，音轨盒子差不多就这样了。

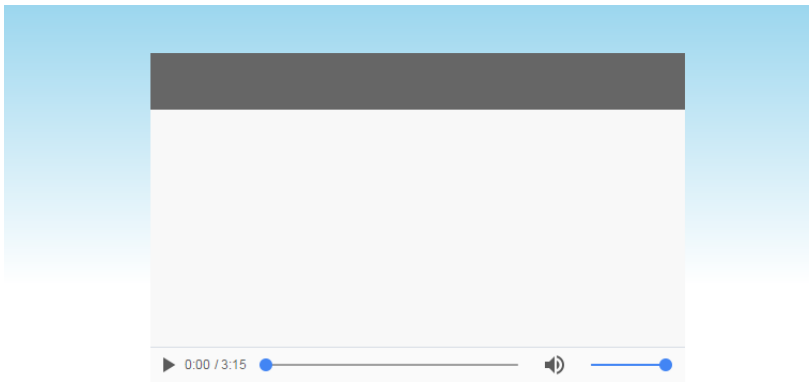
## 3. 用audio标签播放音乐

生成默认的audio对象，然后添加到body区域

```
var audio = document.createElement('audio');
audio.src = 'mp3/01.mp3'
audio.controls = 'controls';
document.body.appendChild(audio);
```

css:

```
audio {
    display: block;
    margin: -100px auto;
    width: 500px;
    border-top: 1px solid;
    border-top-color: #d9dee6;
}
```



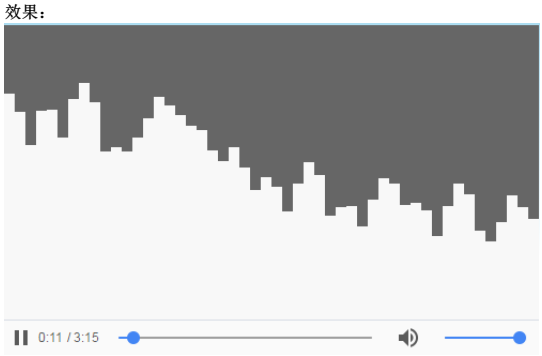

我引入了一首音乐，现在点击播放按钮，就已经可以播放音乐了。

## 4. 音轨盒子与音乐对接

这部分涉及到很多H5的属性，主要就是一个解析的过程，我就直接贴代码了。

```
//1:音频上下文
window.AudioContext = window.AudioContext || window.webkitAudioContext || window.mozAudioContext || window.msAudioContext;
/*动画执行的兼容写法*/
window.requestAnimationFrame = window.requestAnimationFrame || window.webkitRequestAnimationFrame || window.mozRequestAnimationFrame || window.msRequestAnimationFrame;


//2:初始化音轨对象
var audioContext = new window.AudioContext();

var flag = null; //控制是否解析的开关变量

//拿到播放器去解析音乐文件
var audioBufferSouceNode = audioContext.createMediaElementSource(audio);

audio.onplay = function(){
    flag = true;
    //创建解析对象
    var analyser = audioContext.createAnalyser();
    parse(analyser,function(array){
        console.log(array); //打印解析出来的音轨节点
        for(var i = 0;i < len ; i ++){
            document.getElementsByClassName('item')[i].style.height = array[i] + 'px';
        }
    });
}

audio.onpause = function(){
    for(var i = 0;i < len ; i ++){
        document.getElementsByClassName('item')[i].style.height = 1 + 'px';
    }
    flag = false;
}


function parse(analyser,callback){
    if(!flag){
        return;
    }
    audioBufferSouceNode.connect(analyser);
    analyser.connect(audioContext.destination);
    var array = new Uint8Array(analyser.frequencyBinCount);
    analyser.getByteFrequencyData(array);
    if(callback) callback(array);
    requestAnimationFrame(function(){
        parse(analyser,callback);
    });
}
```

效果：




OK，对接完成了。

我把背景色改了一下，不用渐变色了，还是用纯色吧。

这样就完成了，这算是一个简单版本的。

不过原理都是一样的，我已经把这个音轨对接的代码整合到之前的版本了。

演示站点
http://jacksky.d113.163ns.cn/music/index.html

嗯，音乐播放器系列到此为止就算是结束了，感谢各位的捧场。