

音频组件使用指南

前言

背景音乐是专题常见的组成部分，将其抽离成组件，方便开发者进行开发。

同时，因为浏览器的原因，音乐播放 in 移动端会抛出一些奇奇怪怪的错误，抽离为组件也可以统一隔离掉这些由于浏览器差异造成错误，让 fundebug 能够更加集中于处理核心的问题。

目前 fundebug 上由于背景音乐播放出现的问题大概有两类：

第一种是由于部分 webkit 内核浏览器，把 HTMLAudioElement 中的 play() 方法改成了异步执行，返回 promise，当 promise resolved 后才是真正的音乐播放。

假如在音乐加载过程中（未播放，promise 还没被 resolved），用户点击了停止（或者程序调用了 pause() 方法），会直接抛出 DomException，说明浏览器阻断加载过程。

- (ios)AbortError:
The operation was aborted.
- (android)AbortError:
The play() request was interrupted by a call to pause().

第二种是由于浏览器禁止程序播放音乐造成的错误，必须要经过用户操作才能够首次播放。

- (ios) NotAllowedError:
The request is not allowed by the user agent or the platform in the current context, possibly because the user denied permission.
- (android) NotAllowedError:
play() can only be initiated by a user gesture.

为了统一解决以上出现的问题，我们开发了音乐播放的组件。

DEMO


http://static-src.mama.cn/demo/component/index.html#/music-object

用法

基本用法

插入一个音乐，并设置 “自动播放” 。

```
import "mfex-core/lib/music/music.scss"; // 如果没有特别要求，可以使用预定义的一套样式。
import Music from "mfex-core/lib/music";
Music.inject(assetsURL("music/bgm.mp3"));
```

 在大多数专题类项目里面，用以上这一句就可以完成自动音乐播放的需求了。

Vue 使用（BETA）

该组件还专门以 Vue 制作的专题做了一个插件，可以以以下方式引入

暂时仅保证基本使用，如果发现问题可以随时找我反馈。

```
import "mfex-core/lib/music/music.scss"; // 如果没有特别要求，可以使用预定义的一套样式。
import musicObject from "mfex-core/lib/music.vue";
import Vue from "vue";
// 注入全局方法 Vue.musicObject(详见见 API) 以及一个全局组件 g-music-component
Vue.use(musicObject);
```


之后你可以在 vue 的单文件组件中加入这一行，就可以得到一个自动播放的背景音乐

```
<g-music-component msrc="//static-files.mama.cn/act/xxx/assets/music/a.mp3"></g-music-component>

<g-music-component v-bind:msrc="musicSrc"></g-music-component> <!-- 用 v-bind 绑定也可以 -->
```

关于 vue 插件使用的问题，会汇总到子文档中编写，暂不在此处展开。

其他 API 使用

 如果没有其他需要，之后的部分可以跳过。

1. 插入一个音乐，并设置 “自动播放” 。

```
// 以下配置都是默认值
Music.inject(assetsURL("music/bgm.mp3"), {
  className: "g-music",
  activeClassName: "g-music--active",
  loop: true,
  autoplay: true,
  injectAt: null // 插入元素的 id 名称，假如是 falsy 值，就默认插入到 document.body
});

// 利用程序调用 “自动播放”。
Music.inject(assetsURL("music/bgm.mp3"), {
  autoplay: false
})
.then((mo) => {
  // authorized.
  return mo.play();
})
.then(() => {
  console.log("play successfully.");
});
```

2. 插入一个音乐，过几秒后停止

```
import Music from "mfex-core/lib/music";

Music.inject(assetsURL("music/bgm.mp3"))
.then((mo) => {
  // authorized.
  setTimeout(() => mo.pause(), 5000);
});
```

3. 插入一个音乐，过几秒后换为另外一个音乐再播放

```
import Music from "mfex-core/lib/music";

function promiseTimeout(ms) {
  return new Promise((res) => {
    setTimeout(() => res(), ms);
  });
}

let musicObject = null;

Music.inject(gUtils.assetsURL("music/bgm.mp3")).then((mo) => {
  alert("ready");
  musicObject = mo;
  return promiseTimeout(5000);
}).then(() => {
  return musicObject.load(gUtils.assetsURL("music/music-1.mp3"));
}).then(() => {
  // play music-1.mp3;
});

// 带有配置项
Music.inject(gUtils.assetsURL("music/bgm.mp3")).then((mo) => {
  alert("ready");
  musicObject = mo;
  return promiseTimeout(5000);
}).then(() => {
  return musicObject.load(gUtils.assetsURL("music/music-1.mp3"), {
    autoplay: false, // 默认是 true
    loop: false // 默认是 true
  });
}).then(() => {
  return musicObject.play();
}).then(() => {
  // play music-1.mp3;
});
```

4. 实现一个糟糕的用户体验

```
import Music from "mfex-core/lib/music";

let musicObject = null;

Music.inject(gUtils.assetsURL("music/music.mp3")).then((mo) => {
  musicObject = mo;
  return mo.waitForPause();
}).then(() => {
  return musicObject.play();
}).then(() => {
  // continue to play music.mp3;
});
```

API

详细的 API 说明，请到子页面中查看。

Q&A

new MusicObject(src, conf) 返回 promise 的含义？

ios 移动端音乐播放的一个老问题就是，第一次播放音乐获取用户操作才能够正常播放，否则会报错。（其目的是为了保护用户的流量。👊）

有部分安卓机（如华为，或者 oppo，vivo）估计也因为类似的原因，与 ios 一样做了这种限制。

聪明的开发者想到，通过绑定 body 的 touchstart 事件，可以绕过这一个限制，实现“自动播放”的效果。

另外，在微信 webview 可以通过对 weixinJSBridgeReady 这一事件进行监听，获得类似“自动播放”的效果。

考虑到获取用户操作与 weixinJSBridge 的注入过程过程，其实是一段异步的获取权限过程，只要获取了权限，剩下就任意操作了。

因此本组件综合以上两种情况，首先丢出一个 promise，待判断好权限问题了，resolve 一下。使 then() 后的操作是可以得到保证的操作。

参考

https://sites.google.com/a/chromium.org/dev/audio-video/autoplay

https://webkit.org/blog/7734/auto-play-policy-changes-for-macos/

https://docs.google.com/presentation/d/1DhW29bTLkD06JSqp_wLUyByo00nI4kr09IaGQYQEJLU/edit#slide=id.g24637dd1e3_0_11

