

02 | 如何通过WebRTC进行音视频设备检测呢？

2019-07-18 李超

从0打造音视频直播系统

[进入课程 >](#)



讲述：李超

时长 15:00 大小 13.75M



使用过音视频会议或在线教育等实时互动软件的同学都知道，在打开摄像头（Camera）或麦克风（Micphone）的时候，首先要对其进行检测，检测的内容包括：

电脑 / 手机上都有那些音视频设备？

我们选中的音视频设备是否可用？

以手机为例，它一般会包括前置摄像头和后置摄像头。我们可以根据自己的需要，选择打开不同的摄像头。当然，手机上不单有多个摄像头，麦克风的种类就更多了，如：

系统内置麦克

外插的耳机

蓝牙耳机

• • • • •

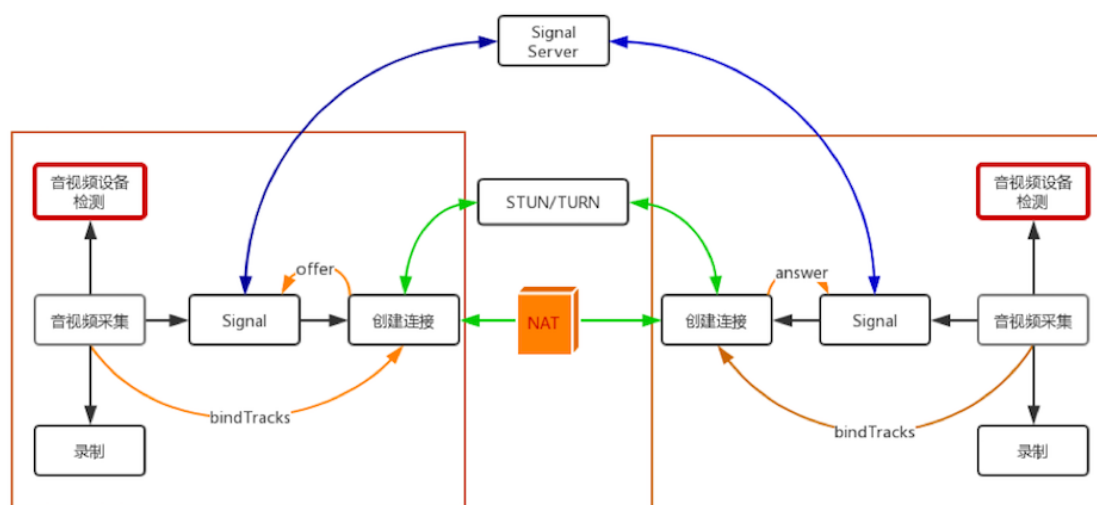
以上这些设备，我们都可以通过手动或自动的方式打开它们。

那么，WebRTC 是否提供了相关的接口，以便我们查询自己机子上都有哪些音视频设备呢？答案是肯定的。

下面我们就来看看如何使用浏览器下 WebRTC API 来显示我们的音视频设备吧。

WebRTC 处理过程

在正式讲解之前，咱们先回顾一下 WebRTC 的整体处理过程图，以便让你清楚地知道咱们这篇文章在整个处理过程中的位置。



WebRTC 1 对 1 音视频实时通话过程示意图

上面这幅图与第一篇文章中的图几乎是一模一样的。其差别在于，我将图中两个**音视频设备检测模块**置红了。

这样，我们本文所讲的内容在整个 WebRTC 处理过程中的位置就一目了然了！

音视频设备的基本原理

既然说到音视频设备，那么我们再顺便介绍一下音视频设备的基本工作原理，对于这些设备工作原理的了解，会为你后续学习音视频相关知识提供很大的帮助。

1. 音频设备

音频有**采样率**和**采样大小**的概念，实际上这两个概念就与音频设备密不可分。

音频输入设备的主要工作是采集音频数据，而采集音频数据的本质就是模数转换（A/D），即将模拟信号转换成数字信号。

模数转换使用的采集定理称为**奈奎斯特定理**，其内容如下：

在进行模拟 / 数字信号的转换过程中，当采样率大于信号中最高频率的 2 倍时，采样之后的数字信号就完整地保留了原始信号中的信息。

你也知道，人类听觉范围的频率是 20Hz ~ 20kHz 之间。对于日常语音交流（像电话），8kHz 采样率就可以满足人们的需求。但为了追求高品质、高保真，你需要将音频输入设备的采样率设置在 40kHz 以上，这样才能完整地将原始信号保留下来。例如我们平时听的数字音乐，一般其采样率都是 44.1k、48k 等，以确保其音质的无损。

采集到的数据再经过量化、编码，最终形成数字信号，这就是音频设备所要完成的工作。在量化和编码的过程中，采样大小（保存每个采样的二进制位个数）决定了每个采样最大可以表示的范围。如果采样大小是 8 位，则它表示的最大值就是 $2^8 - 1$ ，即 255；如果是 16 位，则其表示的最大数值是 65535。

2. 视频设备

至于视频设备，则与音频输入设备很类似。当实物光通过镜头进行到摄像机后，它会通过视频设备的模数转换（A/D）模块，即光学传感器，将光转换成数字信号，即 RGB（Red、Green、Blue）数据。

获得 RGB 数据后，还要通过 DSP（Digital Signal Processor）进行优化处理，如自动增强、白平衡、色彩饱和等都属于这一阶段要做的事情。

通过 DSP 优化处理后，你就得到了 24 位的真彩色图片。因为每一种颜色由 8 位组成，而一个像素由 RGB 三种颜色构成，所以一个像素就需要用 24 位表示，故称之为**24 位真彩**

色。

另外，此时获得的 RGB 图像只是临时数据。因最终的图像数据还要进行压缩、传输，而编码器一般使用的输入格式为 YUV I420，所以在摄像头内部还有一个专门的模块用于将 RGB 图像转为 YUV 格式的图像。

那什么是 YUV 呢？YUV 也是一种色彩编码方法，主要用于电视系统以及模拟视频领域。它将亮度信息（Y）与色彩信息（UV）分离，即使没有 UV 信息一样可以显示完整的图像，只不过是黑白的，这样的设计很好地解决了彩色电视机与黑白电视的兼容问题。

YUV 格式还是蛮复杂的，它有好几种存储方式，需要用一整篇的文章来详述才行。所以，在这里我就不一一描述了，如果你想进一步了解其相关知识可以到网上搜索相关资料自行学习。

通过上面的讲解，现在你应该对音频设备与视频设备都有一个基本的认知了。

WebRTC 设备管理的基本概念

在讲解如何通过浏览器的 WebRTC API 获取音视频设备之前，咱们先了解几个 WebRTC 关于设备的基本概念。如果这些基本概念不清楚的话，就很难理解后面的知识。

[MediaDevices](#)，该接口提供了访问（连接到计算机上的）媒体设备（如摄像头、麦克风）以及截取屏幕的方法。实际上，它允许你访问任何硬件媒体设备。而**咱们要获取可用的音视频设备列表，就是通过该接口中的方法来实现的。**

[MediaDeviceInfo](#)，它表示的是每个输入 / 输出设备的信息。包含以下三个重要的属性：

deviceId，设备的**唯一标识**；

label，设备**名称**；

kind，设备**种类**，可用于识别出是音频设备还是视频设备，是输入设备还是输出设备。

需要注意的是，出于安全原因，**除非用户已被授予访问媒体设备的权限（要想授予权限需要使用 HTTPS 请求），否则 label 字段始终为空。**

另外，label 可以用作指纹识别机制的一部分，以识别是否是合法用户。对于这一点我们以后再专门讨论。

Promise，它是一种 JavaScript 异步处理机制。其思想是，首先执行指定的业务逻辑，而不管逻辑的对错，然后再根据结果做具体的操作：如果成功了做些什么，失败了做些什么。结合下面的例子，可以让你对 **Promise** 有个清楚的认识，生成 **Promise** 对象时，首先会执行 **function** 函数中的逻辑，该函数会根据随机数生成 timeOut，然后定时地对 timeOut 做出判断：

如果 timeOut 小于 1，则调用 resolve 方法。resolve 又会调用 **Promise** 中 **then** 部分传入的函数。

如果 timeOut 大于等于 1，则调用 reject 方法。reject 则会调用 **Promise** 中 **catch** 部分传入的函数。

复制代码

```
1      new Promise(function (resolve, reject) {
2          console.log('start new Promise...');
3
4          // 产生随机值
5          var timeOut = Math.random() * 2;
6          console.log('set timeout to: ' + timeOut + ' seconds.');
```


拼课微信：1716143661

```
7
8          // 设置一个定时器函数，根据随机值触发该函数执行
9          setTimeout(function () {
10              if (timeOut < 1) {
11                  console.log('call resolve()...');
12                  resolve('200 OK');
13              }
14              else {
15                  console.log('call reject()...');
16                  reject('timeout in ' + timeOut + ' seconds.');
```

拼课微信：1716143661

```
17              }
18          }, timeOut * 1000);
19      }).then(function (r) {
20          console.log('Done: ' + r);
21      }).catch(function (reason) {
22          console.log('Failed: ' + reason);
23      });
```

有了上面这些基础知识，你就很容易理解下面的内容了。首先，我们来看浏览器上 WebRTC 获取音视频设备列表的接口，其格式如下：

 复制代码

```
1 MediaDevices.enumerateDevices()
```

通过调用 **MediaDevices** 的 [enumerateDevices\(\)](#) 方法就可以获取到媒体输入和输出设备列表，例如：麦克风、相机、耳机等。是不是非常简单？

该函数返回的是一个 **Promise** 对象。我们只需要向它的 **then** 部分传入一个函数，就可以通过该函数获得所有的音视频设备信息了。

传入的函数有一个参数，它是一个 **MediaDeviceInfo** 类型的数组，用来存放 WebRTC 获取到的每一个音视频设备信息。

这样说可能有点抽象，还是让我们结合下面代码看一个具体的例子吧。

 复制代码

```
1 ...
2
3 // 判断浏览器是否支持这些 API
4 if (!navigator.mediaDevices || !navigator.mediaDevices.enumerateDevices) {
5     console.log("enumerateDevices() not supported.");
6     return;
7 }
8
9 // 枚举 cameras and microphones.
10 navigator.mediaDevices.enumerateDevices()
11 .then(function(deviceInfos) {
12
13     // 打印出每一个设备的信息
14     deviceInfos.forEach(function(deviceInfo) {
15         console.log(deviceInfo.kind + ": " + deviceInfo.label +
16                     " id = " + deviceInfo.deviceId);
17     });
18 })
19 .catch(function(err) {
20     console.log(err.name + ": " + err.message);
21 });
```


总结起来，上面的代码中做了以下几件事儿：

首先，判断浏览器是否支持 `MediaDevice` 接口（老版本浏览器可能不支持）。

如果支持，则调用 `navigator.mediaDevices.enumerateDevices()` 方法获取音视频设备列表，该方法会返回一个 **Promise** 对象。

如果返回 **Promise** 对象成功，则执行 `then` 中的函数。而 **then** 分支中的函数非常简单，它遍历每一个 `MediaDeviceInfo`，并将每个 `MediaDeviceInfo` 中的基本信息打印出来，也就是我们想要的每个音视频设备的基本信息。

但如果失败的话，则执行 `catch` 中的函数。

通过上面的介绍，你是不是觉得在浏览器上检测音视频设备非常简单呢？

设备检测

在获取到电脑 / 手机上的所有设备信息后，我们就可以对设备的可用性做真正的检测了。在我们的设备列表中，可以通过 [MediaDeviceInfo](#) 结构中的 **kind** 字段，将设备分类为音频设备或视频设备。

如果再细分的话，还可以通过 `kind` 字段再将视听设备分为**输入设备**和**输出设备**。如我们平时使用的耳机，从大的方面说它是一个音频设备，但它同时兼有音频输入设备和音频输出设备的功能。

对于区分出的音频设备和视频设备，每种不同种类的设备还会设置各自的**默认设备**。还是以耳机这个音频设备为例，将耳机插入电脑后，耳机就变成了音频的默认设备；将耳机拔出后，默认设备又切换成了系统的音频设备。

因此，在获取到所有的设备列表后，如果我们不指定某个具体设备，直接调用 [《01 | 原来通过浏览器访问摄像头这么容易》](#) 一文中所介绍的 `getUserMedia` API 来采集音视频数据时，它就会从设备列表中的默认设备上采集数据。当然，我们是可以通过 `MediaDeviceInfo` 中的 `deviceId` 字段来指定从哪个具体设备采集数据的，不过这就是后话了。

如果我们能从指定的设备上采集到音视频数据，那说明这个设备就是有效的设备。我们在排查设备问题的时候，就可以利用上面的方法，对每个设备都一项一项进行检测，即**先排查视频设备，然后再排查音频设备**。因此，需要调用两次 `getUserMedia` API 进行设备检测。

第一次，调用 `getUserMedia` API 只采集视频数据并将其展示出来。如果用户能看到自己的视频，说明视频设备是有效的；否则，设备无效，可以再次选择不同的视频设备进行重新检测。

第二次，如果用户视频检测通过了，再次调用 `getUserMedia` API 时，则只采集音频数据。由于音频数据不能直接展示，所以需要使用 JavaScript 中的 `AudioContext` 对象，将采集到的音频计算后，再将其绘制到页面上。这样，当用户看到音频数值的变化后，说明音频设备也是有效的。

通过以上步骤，我们就完成了对指定音视频设备的检测工作。

小结

在本文中，我主要向你介绍了如何通过浏览器中的 WebRTC 接口获取自己机子上的音视频设备列表。通过上面的描述你应该可以了解到，在浏览器下只需要调用 `enumerateDevices` 方法，就可以轻松获取设备信息了。

这个看似简单的接口，其实在 WebRTC 底层做了大量的工作。你可以思考这样一个 case，当我们在获取设备列表时，用户将设备从电脑上拔下来了，此时设备列表会发生怎样的变化呢？如果设备被拔出后，设备列表不发生变化，那么用户正好选择使用被拔下的设备，又会怎样呢？

你还可以顺着这个思路想下去，当我们正在使用某个视频或音频设备的时候，由于某种原因，设备失效了（设备坏了或被拔出了），这种情况下又该怎样处理呢？一般情况下，我们想到的可能是提示设备出错的消息。但如果电脑中还有其他设备可用呢？是不是可以自动切换到另外一个可用的设备上呢？另外，假如有多个设备同时可用，此时该如何切换设备呢？这听上去是不是就很棘手？

但不用担心，我说的这些问题，WebRTC 已经处理得非常好，有兴趣的同学可以自行验证一下 WebRTC 的具体行为是怎样的。所以，从这一点我们也可以看出 WebRTC 所做出的努力。

思考时间

在获取音视频设备时，我们可以得到机子上（电脑 / 手机等）的所有音视频设备。每个设备都有 `deviceId` 字段，那么你可以想一想这个字段在真实场景中是如何通过 `deviceId` 进

行设备切换的呢？

欢迎在留言区与我分享你的想法，也欢迎你在留言区记录你的思考过程。感谢阅读，如果你觉得这篇文章对你有帮助的话，也欢迎把它分享给更多的朋友。



从 0 打造音视频直播系统

手把手教你打造实时互动音视频直播系统

李超

新东方音视频直播技术专家
前沪江音视频架构师



新版升级：点击「 请朋友读」，20位好友免费读，邀请订阅更有**现金**奖励。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 01 | 原来通过浏览器访问摄像头这么容易

下一篇 03 | 如何使用浏览器给自己拍照呢？

精选留言 (12)

写留言



李跃爱学习

2019-07-18

老师音频采样率解释得很清楚，可是采样大小没有深入解释，能否补充一下，不同采样大小意味着什么差别？

作者回复: 如果采样大小太小的话，你的声音的振幅又很大，这时就会对声音产生损失呀！

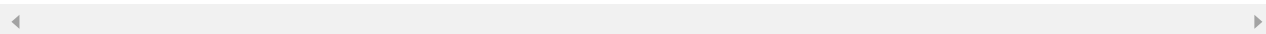
**Keep-Moving**

2019-07-18

想问一下，后续的实例代码也都是js语言的吗？

展开 ∨

作者回复: 基本都是，服务端会有一点 C/C++ 的



1

**一步**

2019-07-18

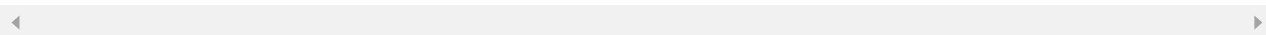
我这通过 navigator.mediaDevices.enumerateDevices() 获取到的设备列表想请教下老师：

1: 对于每个设备 还有个 groupId，这个字段是干什么用的？每一组的设备还有特殊的含义吗？

2: 获取的设备分为 InputDeviceInfo 和 MediaDeviceInfo，这里 InputDeviceInfo 是...

展开 ∨

作者回复: 举个例子，对于耳机，它即是音频输入设备又是音频输出设备，因此它们属于同一个 groupId，你可以自己做个实验验证一下哈。对于第二个问题不同的浏览器表现不一样，其中 chrome 与 webrtc 规范是最接近的，你可以用 chrome 实验一下，是可以区分出音频的输入与输出设置的。



1

**赵健**

2019-07-21

需要注意的是，出于安全原因，除非用户已被授予访问媒体设备的权限，否则 label 字段始终为空 --- 老师，您好，不太明白何为用户已被授予访问媒体设备的权限？比如，第一节课里面的电脑的摄像头已经被打开了，这个还不算有访问设备的权限吗？

展开 ∨

**熊出没**

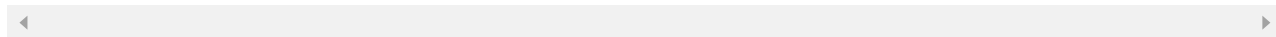


2019-07-20

怎么把这里的代码用起来 JS还是太陌生了

展开 ▾

作者回复: 首先代码部分我会放到github上, 下周一你就可以看到了。其次我想你是对如何发布写的H5页面不熟悉, 这块我还会再写一篇简短的文章放出来。耐心等待!



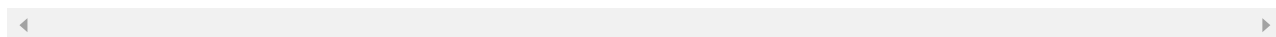
tommy_zhang

2019-07-20

怎么实现摄像头的关闭操作?

展开 ▾

作者回复: 从stream获取的 track后, 调stop方面, 可以去试一试。我想你应该可以自己解决这个问题哈!



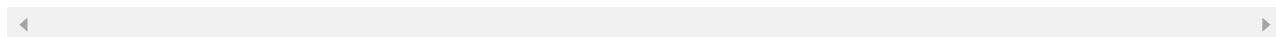
恋着歌

2019-07-18

在调用 getUserMedia 的时候, 可以通过 MediaStreamConstraints 参数可以指定采集设备的deviceId。但是这样只能切换输入设备。

展开 ▾

作者回复: 为啥? 如果你有多个输出设备切换不了?

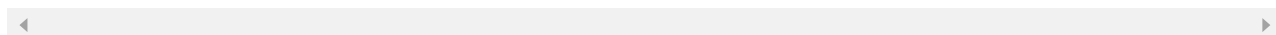


君

2019-07-18

移动端收到web端的视频显示不完全, 居中截取, 请问怎么把web端的视频全部显示出来

作者回复: 没有这种情况, 一定是你自己写的逻辑有问题, 再仔细找找原因!





一步

2019-07-18

老师 我这里就是用 Chrome 进行实验的，
navigator.mediaDevices.enumerateDevices() 返回的设备信息，有时候是
MediaDeviceInfo，InputDeviceInfo 这两个有什么区别呢？

作者回复: MediaDeviceInfo 表是所有设备信息，InputDeviceInfo表式输入设备信息，
MediaDeviceInfo 包含 InputDeviceInfo。



Geek_miao

2019-07-18

排查设备这部分，能不能加上代码说明一下

展开 ▾

作者回复: 原理已经说的很清楚了，至于排查的细节逻辑属于业务逻辑，这不是本专栏的重点，这块要你自己解决哈。实在抱歉！



恋着歌

2019-07-18

deviceId 切换，通过观察 enumerateDevices() 返回的数据，目的是要向 default 切换。
对目标设备信息做一次拷贝，修改 deviceId 为 default，修改 label 加上前缀“默认 -”。
不知道老师的题目是不是这样解答？

作者回复: 你加上“默认 -” 它能真的变成默认吗哈？你可以自己亲自试一下



LongXiaJun

2019-07-18

打卡！

展开

作者回复: 打卡，哈哈

