

## 01 | 原来通过浏览器访问摄像头这么容易

2019-07-16 李超

从0打造音视频直播系统

[进入课程 >](#)



讲述：李超

时长 16:59 大小 15.57M



对于很多从事 JavaScript 开发的同学来说，基本都认为 JavaScript 是专门做页面控制的。如果用 JavaScript 做音视频处理，那真是很难想象的事儿。你可能首先想到的问题是：JavaScript 或者浏览器的性能跟得上吗？

而 Google 却不这么认为。Google 就是要做一些常人无法想象，又难以理解的事情，否则它就不是 Google 了。

“**浏览器 + WebRTC**”就是 Google 给出的答案。2011 年，Google 创立了 WebRTC 项目，其愿景就是可以在浏览器之间快速地实现音视频通信。

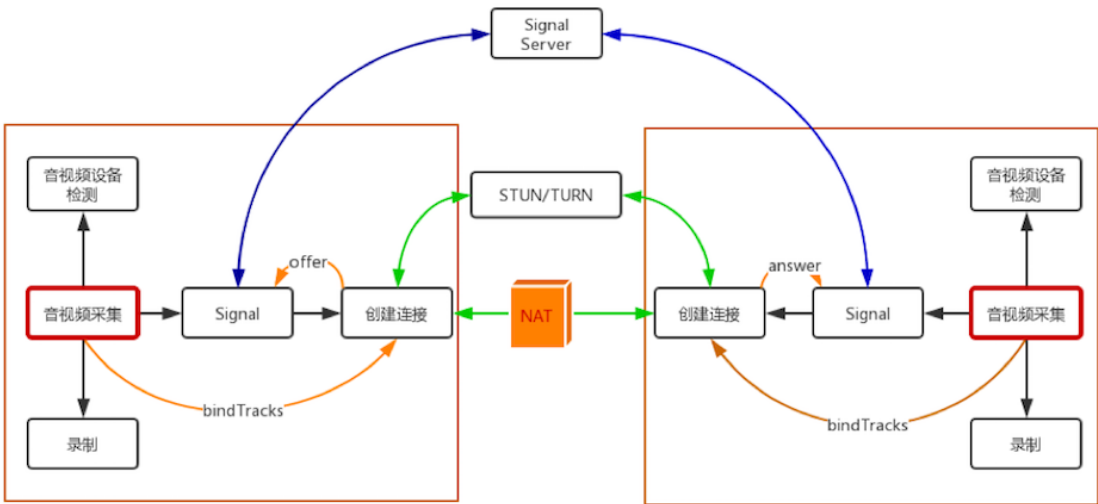
随着 WebRTC 1.0 规范的推出，现在主流浏览器 Chrome、Firefox、Safari 以及 Edge 都已经支持了 WebRTC 库。换句话说，**在这些浏览器之间进行实时音视频通信已经很成熟**

了。

下面我就通过讲解 JavaScript/ 浏览器访问电脑上的音视频设备，向你展示通过现代浏览器访问音视频设备是何其简单。

## WebRTC 处理过程

在正式讲解如何通过浏览器采集音视频数据之前，我先向你介绍一下 WebRTC 实现一对一音视频实时通话的整个处理过程。对这个过程的了解，可以帮助你在阅读文章时，能清楚地知道所阅读的这篇文章、所要学习的知识点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

上面这幅图是整个 WebRTC 1 对 1 音视频实时通话的过程图。通过这幅图，你可以看出要实现 1 对 1 音视频实时通话其过程还是蛮复杂的。

这幅图从大的方面可以分为 4 部分，即**两个 WebRTC 终端**（上图中的两个大方框）、**一个 Signal（信令）服务器**和**一个 STUN/TURN 服务器**。

WebRTC 终端，负责音视频采集、编解码、NAT 穿越、音视频数据传输。

Signal 服务器，负责信令处理，如加入房间、离开房间、媒体协商消息的传递等。

STUN/TURN 服务器，负责获取 WebRTC 终端在公网的 IP 地址，以及 NAT 穿越失败后的数据中转。

接下来，我就向你描述一下**WebRTC 进行音视频通话的大体过程**。

当一端（WebRTC 终端）进入房间之前，它首先会检测自己的设备是否可用。如果此时设备可用，则进行**音视频数据采集**，这也是本篇我们要介绍的重点内容。

采集到的数据一方面可以做预览，也就是让自己可以看到自己的视频；另一方面，可以将其录制下来保存成文件，等到视频通话结束后，上传到服务器让用户回看之前的内容。

在获取音视频数据就绪后，WebRTC 终端要发送 **“加入”** 信令到 Signal 服务器。Signal 服务器收到该消息后会创建房间。在另外一端，也要做同样的事情，只不过它不是创建房间，而是加入房间了。待第二个终端成功加入房间后，第一个用户会收到 **“另一个用户已经加入成功”** 的消息。

此时，第一个终端将创建 **“媒体连接” 对象**，即 **RTCPeerConnection**（该对象会在后面的文章中做详细介绍），并将采集到的音视频数据通过 RTCPeerConnection 对象进行编码，最终通过 P2P 传送给对端。

当然，在进行 P2P 穿越时很有可能失败。所以，当 P2P 穿越失败时，为了保障音视频数据仍然可以互通，则需要通过 TURN 服务器（TURN 服务会在后面文章中专门介绍）进行音视频数据中转。

这样，当音视频数据“历尽千辛万苦”来到对端后，对端首先将收到的音视频数据进行解码，最后再将其展示出来，这样就完成了一端到另一端的单通。如果双方要互通，那么，双方都要通过 RTCPeerConnection 对象传输自己一端的数据，并从另一端接收数据。

以上，就是这幅图大体所描述的含义。而本文要重点介绍的内容就是 WebRTC 终端中的音视频采集部分。

## 音视频采集基本概念

在正式介绍 JavaScript 采集音视频数据的 API 之前，你还需要了解一些基本概念。这些概念虽然都不难理解，但在后面讲解 API 时都会用到它们，很是重要，所以在这里我还是给你着重汇总和强调下。

**摄像头。**用于捕捉（采集）图像和视频。

**帧率。**现在的摄像头功能已非常强大，一般情况下，一秒钟可以采集 30 张以上的图像，一些好的摄像头甚至可以采集 100 张以上。我们把**摄像头一秒钟采集图像的次數称为帧率**。帧率越高，视频就越平滑流畅。然而，在直播系统中一般不会设置太高的帧率，因为帧率越高，占的网络带宽就越多。

**分辨率。**摄像头除了可以设置帧率之外，还可以调整分辨率。我们常见的分辨率有 2K、1080P、720P、420P 等。分辨率越高图像就越清晰，但同时也带来一个问题，即占用的带宽也就越多。所以，在直播系统中，分辨率的高低与网络带宽有紧密的联系。也就是说，分辨率会跟据你的网络带宽进行动态调整。

**宽高比。**分辨率一般分为两种宽高比，即 16:9 或 4:3。4:3 的宽高比是从黑白电视而来，而 16:9 的宽高比是从显示器而来。现在一般情况下都采用 16:9 的比例。

**麦克风。**用于采集音频数据。它与视频一样，可以指定一秒内采样的次数，称为**采样率**。每个采样用几个 bit 表示，称为**采样位深或采样大小**。

**轨 ( Track ) 。**WebRTC 中的“轨”借鉴了多媒体的概念。火车轨道的特性你应该非常清楚，两条轨永远不会相交。“轨”在多媒体中表达的就是**每条轨数据都是独立的，不会与其他轨相交**，如 MP4 中的音频轨、视频轨，它们在 MP4 文件中是被分别存储的。

**流 ( Stream ) 。**可以理解为容器。在 WebRTC 中，“流”可以分为媒体流 ( MediaStream ) 和数据流 ( DataStream )。其中，**媒体流**可以存放 0 个或多个音频轨或视频轨；数据流可以存 0 个或多个数据轨。

## 音视频采集

有了上面这些基本概念，你就可以很容易理解后面所要讲的内容了。接下来，就让我们来具体看看在浏览器下采集音视频的 API 格式以及如何控制音视频的采集吧。

### 1. getUserMedia 方法

在浏览器中访问音视频设备非常简单，只要调用**getUserMedia**这个 API 即可。该 API 的基本格式如下：

```
1 var promise = navigator.mediaDevices.getUserMedia(constraints);
```

它返回一个**Promise**对象。

如果**getUserMedia**调用成功，则可以通过 Promise 获得**MediaStream**对象，也就是说我们现在已经从音视频设备中获取到音视频数据了。

如果调用失败，比如用户拒绝该 API 访问媒体设备（音频设备、视频设备），或者要访问的媒体设备不可用，则返回的 Promise 会得到 **PermissionDeniedError** 或 **NotFoundError** 等错误对象。

## 2. MediaStreamConstraints 参数

从上面的调用格式中可以看到，**getUserMedia**方法有一个输入参数**constraints**，其类型为 **MediaStreamConstraints**。它可以指定**MediaStream**中包含哪些类型的媒体轨（音频轨、视频轨），并且可为这些媒体轨设置一些限制。

下面我们就来详细看一下它包括哪些限制，这里我引用一下 WebRTC 1.0 规范对 **MediaStreamConstraints**的定义，其格式如下：

```
1 dictionary MediaStreamConstraints {  
2     (boolean or MediaTrackConstraints) video = false,  
3     (boolean or MediaTrackConstraints) audio = false  
4 };
```


从上面的代码中可以看出，该结构可以指定采集音频还是视频，或是同时对两者进行采集。

举个例子，比如你只想采集视频，则可以像下面这样定义 constraints：

```
1  
2 const mediaStreamContrains = {  
3     video: true  
4 };
```




或者，同时采集音视和视频：

 复制代码

```
1
2 const mediaStreamConstraints = {
3   video: true,
4   audio: true
5 };
6
```

其实，你还可以通过 `MediaTrackConstraints` 进一步对每一条媒体轨进行限制，比如下面的代码示例：

 复制代码

```
1 const mediaStreamConstraints = {
2   video: {
3     frameRate: {min: 20},
4     width: {min: 640, ideal: 1280},
5     height: {min: 360, ideal: 720},
6     aspectRatio: 16/9
7   },
8   audio: {
9     echoCancellation: true,
10    noiseSuppression: true,
11    autoGainControl: true
12  }
13 };
```

上面这个例子表示：视频的帧率最小 20 帧每秒；宽度最小是 640，理想的宽度是 1280；同样的，高度最小是 360，最理想高度是 720；此外宽高比是 16:9；对于音频则是开启回音消除、降噪以及自动增益功能。

除了上面介绍的这些参数来控制摄像头和麦克风外，当然还有其他一些参数可以设置，更详细的参数信息，可以跳到下面的[参考部分](#)。

通过上面的这些方式就可以很方便地控制音视频的设备了，是不是非常简单？

## 如何使用 getUserMedia API


接下来，我们看一下如何使用上面介绍的 API 来采集视频数据吧。

下面的 HTML 代码非常简单，它引入一段 JavaScript 代码用于捕获音视频数据，然后将采集到的音视频数据通过 video 标签播放出来。

 复制代码

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Realtime communication with WebRTC</title>
5     <link rel="stylesheet", href="css/client.css" />
6   </head>
7   <body>
8     <h1>Realtime communication with WebRTC </h1>
9     <video autoplay playsinline></video>
10    <script src="js/client.js"></script>
11  </body>
12 </html>
```

为便于你更好地理解该部分的知识，上面这段代码中有两条代码我需要解释一下，一句是：

 复制代码

```
1 <video autoplay playsinline></video>
```

它是 HTML5 的视频标签，不仅可以播放多媒体文件，还可以用于播放采集到的数据。其参数含义如下：

**autoplay**，表示当页面加载时可以自动播放视频；

**playsinline**，表示在 HTML5 页面内播放视频，而不是使用系统播放器播放视频。

另一句是：

```
1 <script src="js/client.js"></script>
```

它引入了外部的 JavaScript 代码，起到的作用就是获取视频数据。具体代码如下：

```
1 'use strict';
2
3 const mediaStreamConstraints = {
4   video: true
5 };
6
7 const localVideo = document.querySelector('video');
8
9 function gotLocalMediaStream(mediaStream){
10   localVideo.srcObject = mediaStream;
11 }
12
13 function handleLocalMediaStreamError(error){
14   console.log('navigator.getUserMedia error: ', error);
15 }
16
17 navigator.mediaDevices.getUserMedia(mediaStreamConstraints).then(
18   gotLocalMediaStream
19 ).catch(
20   handleLocalMediaStreamError
21 );
```

通过上面的代码，我们就可以采集到视频数据并将它展示在页面上了，很简单吧！接下来，我们来大体看一下它的逻辑。

JavaScript 代码中首先执行 **getUserMedia()** 方法，该方法会请求访问 Camera。如果是第一次请求 Camera，浏览器会向用户弹出提示窗口，让用户决定是否可以访问摄像头。如果用户允许访问，且设备可用，则调用 **gotLocalMediaStream** 方法。

在 **gotLocalMediaStream** 方法中，其输入参数为 **MediaStream** 对象，该对象中存放着 **getUserMedia** 方法采集到的音视频轨。我们将它作为视频源赋值给 HTML5 的 video 标签的 **srcObject** 属性。这样在 HTML 页面加载之后，就可以在该页面中看到摄像头采集到的视频数据了。



在这个例子中，**getUserMedia**方法的输入参数**mediaStreamConstraints**限定了只采集视频数据。同样的，你也可以采集音频数据或同时采集音频和视频数据。

## 小结

在 WebRTC 中，**MediaTrack**和**MediaStream**这两个概念特别重要，后续学习 WebRTC 的过程中，我们会反复用到，所以在这最开始你就要理解透这两个概念。举个例子，如果你想在一个房间里，同时共享视频、共享音频、共享桌面，该怎么做呢？如果你对 **MediaTrack** 和 **MediaStream** 真正理解了，就会觉得 WebRTC 处理这种情况太简单了。

另外，在本文中我重点介绍了**getUserMedia**这个 API，它是 **WebRTC 几个核心 API 之一，你必须熟练掌握它**。因为通过它，你可以对音视频设备做各种各样的控制，例如，是采集音频，还是采集视频？视频的分辨率是多少？帧率是多少？音频的采样率是多少？

当然，特别关键的一点是可以通过该 API 开启**回音消除**。回音消除问题是所有做实时互动直播系统最难解决的问题之一。对于 JavaScript 开发同学来说，现在只需要调用该 API 时，将回音消除选项打开就可以了，一下子解决了世界难题。

最后，我还通过一个例子向你具体展示了视频采集后的效果。相信通过这些讲解和展示，你应该已经感受到目前浏览器的强大，以及它可以做更多、更有意思的音视频相关的事情了。

这里你也可以看一下我做出来的效果图（没有美颜）：



```
{
  "aspectRatio": 1.3333333333333333,
  "deviceId": "c73e53824ede816012",
  "frameRate": 15,
  "groupId": "67335cela504b55f4a5",
  "height": 480,
  "resizeMode": "none",
  "width": 640,
  "videoKind": "color"
}
```

## 思考时间

上面我们一起学习了如何通过**getUserMedia**获取到音视频数据。而在真实的场景中，我们往往不但要获取到默认设备的音视频数据，还要能获取到**某个指定的设备**的音视频数据。比如，手机上一般都有两个摄像头——前置摄像头和后置摄像头。那么，你有没有办法采集到指定摄像头的视频数据呢？

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。

## 参考

getUserMedia API 控制设备的参数及其含义如下：

参数	含义	备注
width	视频宽度	
height	视频高度	
aspectRatio	视频宽/高度	
frameRate	帧率	
facingMode	user: 前置摄像头 enviroment: 后置摄像头 left: 前置左侧摄像头 right: 前置右侧摄像头	enum VideoFacingModeEnum { "user", "environment", "left", "right" }
resizeMode	是否允许调整图像大小	
volume	音量大小	
sampleRate	音频采样率	
sampleSize	音频采样大小	
echoCancellation	是否开启回音消除	
autoGainControl	是否开启自动增益	
noiseSuppression	是否开启降噪	
latency	延迟大小	
channalCount	声道数	
deviceID	设备ID	指定使用哪个输入/输出设备
groupID	设置组ID	如果两个设备属于同一个物理设备，则他们具有相同的 groupID。例如具有麦克风功能的耳机。

# 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 开篇词 | 5G的到来将会为音视频插上飞翔的翅膀

下一篇 02 | 如何通过WebRTC进行音视频设备检测呢？

## 精选留言 (33)

 写留言

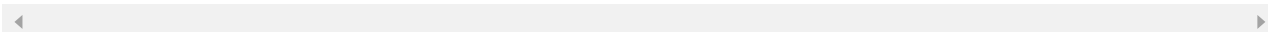


翌

2019-07-16

其他平台学过老师的课程，讲的很详细。一直希望老师讲讲Android端相关的知识点，后来想来，音视频重点还是在服务端，Web 目前又比较普及，无论是PC还是移动终端，都越来越多的支持 Web 了。

作者回复: 谢谢，铁粉呀！



3



西西弗与卡夫卡

2019-07-16

获取前置摄像头

```
const mediaStreamConstraints = {  
  video: { facingMode: "user" },  
  audio: true  
};...
```

展开 ▾

💬 2

👍 3



**当当**

2019-07-16

老师，直接在chrome测试成功，声音不正常（回声很想），加了参数消除回音，还是不行，为什么啊？

作者回复: 自己的音频没有mute 吧？把video 标签里加个muted 试试



💬

👍 1



**超威**

2019-07-16

请问NAT穿越是啥？

展开 ▾

作者回复: P2P，端与端直接进行连接，不需要服务器中转数据，这样可以节省服务器带宽，但并不意味着不需要服务器，服务器作为辅助功能



💬 1

👍 1



**XE.COM**

2019-07-16

你好，老师，我是做webrtc的原生Android客户端的，您的课程有这方面的讲解吗？

作者回复: 这个专栏中没有，其实学完这个专栏你自己可以用android 实现了，很简单



💬

👍 1

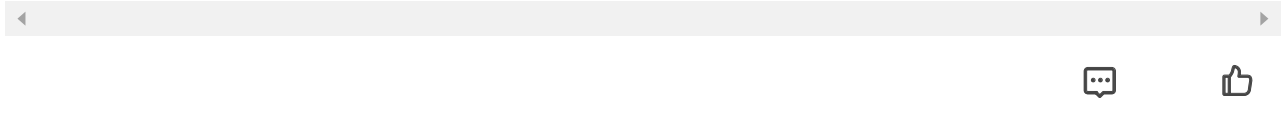


子飞鱼

2019-07-19

移动端切换前置/后置摄像头可通过设置video的属性值facingMode来处理，分别为user和environment

作者回复: 是的，可以在移动端上自己新手试一试！

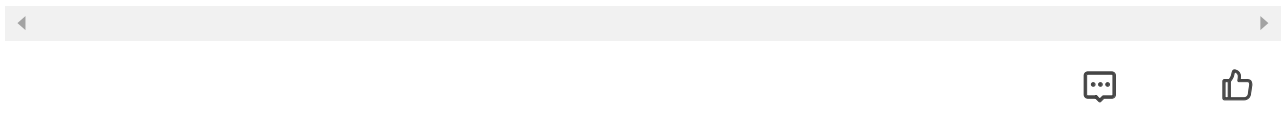


风筝

2019-07-19

<link rel="stylesheet", href="css/client.css" />老是这个css没有呢？有demo吗

作者回复: 有 demo，下周就会放到github上，耐心等待

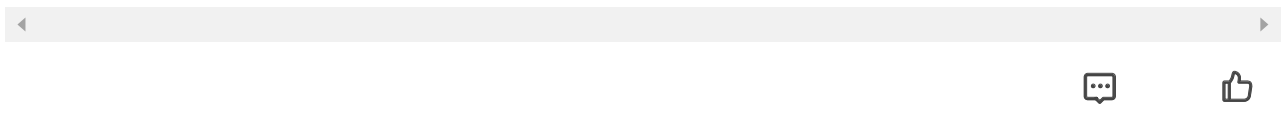


曹士远

2019-07-19

STUN/TURN 服务器是否有开源的，我之前看到都是使用谷歌的。

作者回复: 是的coturn就可以，大家都用它！



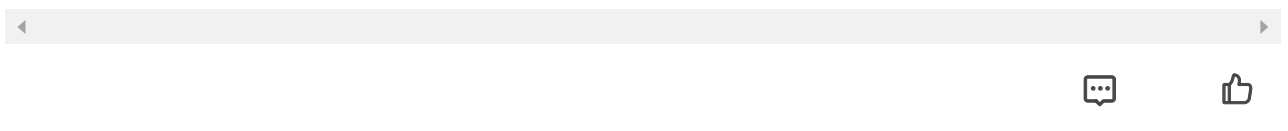
曹士远

2019-07-19

如果使用freeswitch在其中作为一个什么角色？

展开 ∨

作者回复: freeswitch可以做混音服务器，或mcu。不过在直播中，一般用它做服务器混音。







我听你说.....

2019-07-19

老师，通接口获取老是调用catch怎么回事，怎么解决

展开 ∨

作者回复: 函数没有调用成功，到catch之后报的具体错误是什么？



曹志翔

2019-07-18

关于“video 标签的 playsinline 属性”的兼容性，请教老师2个问题：

- 1、部分品牌的手机中（比如小米、锤子），即使设置了 video 标签的 playsinline 属性，还是会调用系统播放器播放音视频流，该如何处理？
- 2、iOS 中，设置了 video 标签 playsinline 属性的同时，设置 autoplay 属性，不播放声音，该如何处理？

展开 ∨

作者回复: 目前移动端的浏览器确实还存在很多问题，所以移动端一般还是使用native开发比较好。在PC和Mac上可以直接使用浏览器，因为Native与浏览器之间是可以互通的，所以这个问题可以通过这种方式很好的解决。



曹志翔

2019-07-18

请教老师：

如果要传输控制信息（比如，远程控制小车），控制信息是通过 DataStream 实现传输吗？

DataStream 后面课程是否会讲到？

作者回复: 信令的控制要通过专门的信令通道（如 http/https ws/wss）与信令服务器交互。DataStream 主要用于传输二进制数据，所以这是两回事儿。这两个后面都会讲到。





曹志翔-18

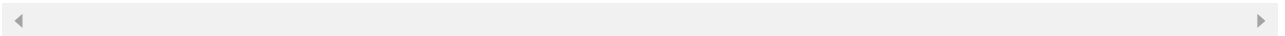
关于“getUserMedia的兼容性”，老师能不能传授些经验。

我在 caniuse.com 中查到4条信息：

- 1、IE浏览器不支持 getUserMedia。
- 2、旧浏览器不能使用 srcObject。...

展开 ∨

作者回复: 目前新版本的浏览器都已经支持，IE 确实是个问题，有一些不是特别好的办法，如给 IE 编写的一控件安装上去，来解决这个问题。但更多的直播系统是通过多客户端来解决这个问题，如果用户浏览器实在不支持的话，就用native方案！

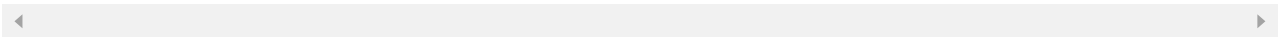


eques

2019-07-18

Cannot read property 'getUserMedia' of undefined ? ? ? ?

作者回复: 用https或http://localhost



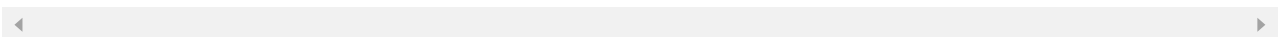
tommy\_zhang

2019-07-18

老师，有没有对应的讲义可以下载？

展开 ∨

作者回复: 没有讲义，文章已经写的足够细了

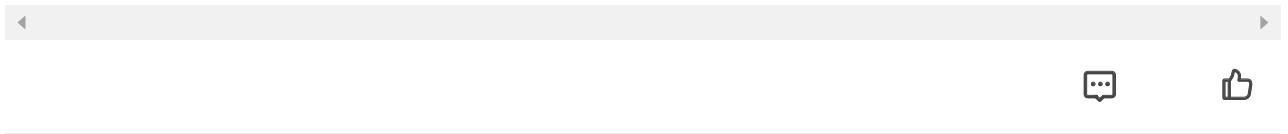


蓝桥

2019-07-18

你好，老师，现有的 rtmp 的直播解决方案和 webRTC 之间优劣势在什么地方？

作者回复: Rtmp 底层用的tcp,webrtc底层主要使用udp,使用tcp 就注定他在极端网络情况下没法实时通信

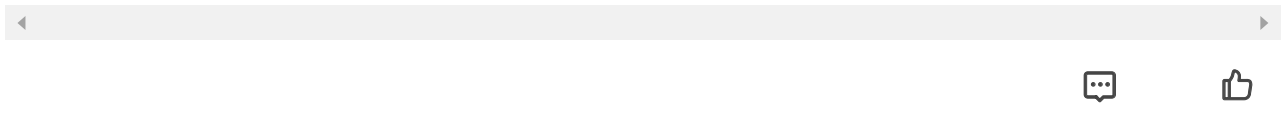


咸平四年望帝遗心

2019-07-17

你好，我是做Android端的，也想了解下音视频开发，请问你这课程适合我吗？

作者回复: 适合，无论你做少，只要对音视频感兴趣，想做音视频直播相关的工作都应该看看这个专栏哈



空白

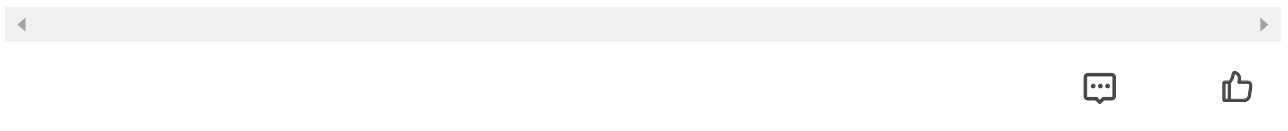
2019-07-17

在使用nuxt创建的vue项目中，  
使用ip访问：navigator.mediaDevices.是 undefined  
使用localhost访问：navigator.mediaDevices. 存在

这是为什么呢

展开 ▾

作者回复: 出于安全的原因，你只能用localhost 访问或https 访问时才能检测到mediaDevice



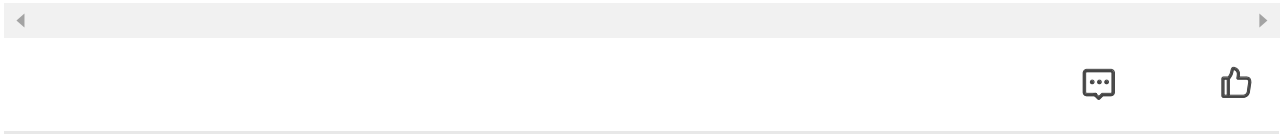
林

2019-07-17

老师我请教一个问题，我把写好的页面使用nginx部署，页面文件叫video.html。 使用 http://localhost/video.html访问一切正常，但是使用http://我的ip地址/video.html访问却报错，错误信息：navigator.getUserMedia is not a function。请问可能是什么原因造成的？

展开 ▾

作者回复: 必须是https,主要出于安全的考虑,不能让随便一个网站就能访问你的摄像头不是?另外就是本地测试也是可以的!



**种花家的兔子**

2019-07-17

老师,您好,我看完第一小节,在本地实现了,我想Android通过JsBridge桥接本地Html文件,来实现实现视频流采集,然后通过android推流,有什么好的方案吗

作者回复: android 端可以直接用原生的方案哈,用原生的是不是更高效些?

