# 设备管理模块

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Changed By | Changes |
| 1.0 | 2019.10 | Chrou | 初稿 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## 前言

设备管理模块除了包含设备得热拔插处理和设备能力扫描，还对取流进行封装，音频、视频和桌面演示使用统一接口取流。

## 功能概述

设备管理模块主要实现的功能包含：

1、音视频设备的枚举

2、音视频设备的热拔插监听

3、音视频设备的记录及IndexedDB本地存储

4、输入设备的能力扫描及记录, 屏幕大小也可以作为共享源的输入源作为一个记录

5、根据给定分辨率获取设备支持的最接近的分辨率

6、根据设备id获取设备支持的能力列表

7、获取音频、视频、桌面共享流，并在取流失败后实现重复取流直到完全失败

8、提供默认设备能力扫描列表QuickScanList

## 逻辑设计

### 3.1 设备热拔插检查和设备能力扫描

1、检查可用设备列表

2、更新localStorage存储

（1）本地存储没有任何值，直接设置获取的设备列表到localStorage中

（2）未获取当任何有效的设备列表，localStorage保存的设备全部设置为不可用

（3） 获取到设备列表，且localStorage中有设备存储信息时：

- 判断localStorage中的设备是否有还存在，不存在则设置状态为 unavailable，还存在的置为available

- 判断设备是否是新设备，是的话，添加到localStorage中

3、设置设备所支持的取流能力：frameRate, width, height

（1）判断是否支持关键字**（exact/ideal/max/min）**取流，支持则使用exact取流，不支持就不使用关键字取流

（2）localStorage中存储的所有cameras逐一遍历默认的快速扫描列表取流，取流成功则添加到该设备的capability列表，失败则不进行处理

- 关键字取流：直接使用exact

- 非关键字取流：取流成功后根据video的实际尺寸判断是否取流成功，如果video的videoWidth，videoHeight和当前扫描的分辨率已知，则认为成功

- 流存在且浏览器支持applyConstraints接口时使用applyConstraints应用分辨率

1. 根据给定的deviceId和frameRate、width、height获取最合适分辨率

5、设置设备定时检查开关，定时跟新设备状态，可用还是不可用等

### 3.2 取流实现

取流包括音频、视频和演示流，统一封装为一个接口。

Video取流限制顺序为exact，ideal，不使用关键字

1、获取取流类型的constraints（audio/video/screenShare）

2、取流：根据取流类型调用不同接口取流

- audio 使用getUserMedia

- video：流不存在或浏览器不支持applyConstraints接口时使用getUserMedia；流存在且浏览器支持applyConstraints接口时使用applyConstraints

- screenShare： 支持getDisplayMedia接口的使用getDisplayMedia，不支持的使用插件实现

1. 取流成功后调用回调，显示流
2. video取流失败后，重复取流：
   1. 根据当前取流失败的分辨率在设备支持的能力列表中查找比当前分辨率小得下一个分辨率
   2. 根据constraintsKeyWord判断是使用exact还是ideal还是不使用关键字取流。
   3. 如果获取不到下一个分辨率，说明当前限制条件下已经尝试完设备支持得所有能力，需要换限制取流：
      1. 如果之前使用exact，则替换为ideal
      2. 如果直线使用ideal，则不使用关键字
      3. 如果之前没有使用关键字，并且获取不到下一个分辨率，说明各种限制都无法取流，此时取流彻底失败

## 接口说明

### getQuickScanList

MediaDevice.prototype.getQuickScanList = function(){...}

获取给定设备能力扫描列表

示例：

mediaDeviceInstance.getQuickScanList ()

### enumDevices

MediaDevice.prototype.enumDevices = function(deviceInfoCallback, error) {}

获取音视频设备并进行分类

参数说明：

\* @param deviceInfoCallback：设备信息获取成功后使用回调返回信息没有回调时直接return

\* @param error：浏览器不支持enumerate devices

|  |  |  |  |
| --- | --- | --- | --- |
| Device kind | name | Set Attributes | Description |
| audioinput | microphone | Label/deviceId/groupId/status | 音频输入 |
| audiooutput | speaker | Label/deviceId/groupId/status | 音频输出 |
| videoinput | camera | Label/deviceId/groupId/status/ capability | 视频输出 |
| - | screenResolution |  | 屏幕分辨率 |
|  | isConstraintsKeywordSupport |  | 是否支持关键字 |

示例：

mediaDeviceInstance.enumDevices (function(){}, error)

### updateDeviceInfo

MediaDevice.prototype.updateDeviceInfo = function(deviceInfos, type){}

更新localStorage存储

参数说明：

\* @param deviceInfos 所有的媒体数据

\* @param ： cameras / microphones / speakers， 更新的类型

示例：

mediaDeviceInstance.updateDeviceInfo (deviceInfo, “cameras”)

### closeStream

MediaDevice.prototype.closeStream = function(stream){}

清除流

参数说明：

\* @param stream 要清除的流

示例：

mediaDeviceInstance.closeStream (stream)

### isConstraintsKeywordSupport

MediaDevice.prototype.isConstraintsKeywordSupport = async function(){}

判断取流是否支持关键字：min/max/exact/ideal，常见：测试一体机不支持关键字

返回值：

True 支持关键字

False 不支持关键字

示例：

mediaDeviceInstance.isConstraintsKeywordSupport()

### getStreamWithExactConstraints

MediaDevice.prototype.getStreamWithExactConstraints = async function(){}

使用exact关键字取流，用于设备能力扫描，内部遍历当前存在的所有设备，根据快速扫描列表取流，判断设备支持哪些分辨率

示例：

mediaDeviceInstance.getStreamWithExactConstraints ()

### getStreamWithoutConstraintsKeyWords

MediaDevice.prototype.getStreamWithoutConstraintsKeyWords = async function(scanListIndex, cameraIndex){}

兼容不支持min/max/ideal/exact的情况，使用{audio:false, video: { width: 1280, height: 720}} 格式取流。通过取流后的video实际尺寸判断取流是否成功

参数说明：

\* @param scanListIndex 当前扫描的分辨率列表的索引值

\* @param cameraIndex 当前扫描的摄像头列表值

\* @returns {Promise<void>}

示例：

mediaDeviceInstance.getStreamWithoutConstraintsKeyWords (scanListIndex, cameraIndex)

### displayVideoDimensions

MediaDevice.prototype.displayVideoDimensions = function(scanListIndex, cameraIndex) {}

video 元数据加载完成后触发，用于判断取流后的分辨率与实际要求是否一致。用于不支持exact取时判断

参数说明：

\* @param scanListIndex 当前扫描的分辨率列表的索引值

\* @param cameraIndex 当前扫描的摄像头列表值

示例：

mediaDeviceInstance.displayVideoDimensions (scanListIndex, cameraIndex)

### setDeviceCapability

MediaDevice.prototype.setDeviceCapability = async function (){}

设置设备所支持的取流能力：frameRate, width, height。调用这里开始设备能力扫描

示例：

mediaDeviceInstance.setDeviceCapability ()

### checkAvailableDev

MediaDevice.prototype.checkAvailableDev = function (){}

检查可用设备列表

示例：

mediaDeviceInstance.checkAvailableDev ()

### setDeviceCheckInterval

MediaDevice.prototype.setDeviceCheckInterval = function (switchOn) {}

设备定时检查开关

参数说明：

@param switchOn: true 开启定时器； false 关闭定时器

示例：

mediaDeviceInstance.setDeviceCheckInterval (true)

### getSuitableResolution

MediaDevice.prototype.getSuitableResolution = function (expectRes){}

获取最接近，最合适的设备支持的分辨率

参数说明：

@param expectRes 当前希望获取的分辨率

|  |  |  |
| --- | --- | --- |
| 参数 | 是否可空 | 示例 |
| deviceId | N | 4b5305afd805f2d8439eac80dc94b14846799929d44d18c7dd8fc97eda75c046 |
| frameRate | Y | 30 |
| width | Y | 1280 |
| height | Y | 720 |
|  |  |  |

示例：

mediaDeviceInstance.getSuitableResolution ({

deviceId: 4b5305afd805f2d8439eac80dc94b6799929d44d1a75c046

frameRate: 30,

width: 640,

height: 480

})

# 取流

### getMedia

MediaDevice.prototype.getMedia = function (data, constraints){}

获取音频、视频或桌面演示流

参数说明：

\* @param data

|  |  |  |
| --- | --- | --- |
| 参数 | 是否可空 | 说明 |
| constraintsKeyWord | Y | 默认exact |
| streamType | N |  |
| frameRate | Y |  |
| width | Y |  |
| height | Y |  |
| deviceId | Y |  |

\* @param constraints 取流分辨率

示例：

Var data = {

constraintsKeyWord: "exact"

deviceId: "5e3722883e2e9337040a4f1ababf85a5bd2f6a36afc815fd391424ac05a84ab0"

frameRate: 30

height: 720

streamType: "video"

width: 1280

}

Var constraints = {

audio: false

video:

aspectRatio: {exact: 1.7777777777777777}

deviceId: {exact: "5e3722883e2e9337040a4f1ababf85a5bd2f6a36afc815fd391424ac05a84ab0"}

frameRate: {exact: 30}

height: {exact: 720}

width: {exact: 1280}

}

mediaDeviceInstance.getMedia (data, constraints)

### getConstraints

MediaDevice.prototype.getConstraints = function(data, reTry){}

获取分辨率

参数说明：

@param data, eg.{

constraintsKeyWord: "exact"

deviceId: "8cd24e4d2ff8de04d9170e94899fdb24a10ac7c9d09cb90bbe796e754f768d03"

frameRate: 30

height: 720

streamType: "video"

width: 1280

}

\* @param reTry: true 取流失败后重新取流， false 第一次取流，video使用。

示例：

mediaDeviceInstance.getConstraints (data, reTry)

### getVideoConstraints

MediaDevice.prototype.getVideoConstraints = function(data, reTry){}

获取video 分辨率

参数说明：

\* @param data 需要的参数

constraintsKeyWord: "exact"

deviceId: "8cd24e4d2ff8de04d9170e94899fdb24a10ac7c9d09cb90bbe796e754f768d03"

frameRate: 15

height: 720

streamType: "video"

width: 1280

\* @param reTry 需要得参数 : true 取流失败重新取流, false 第一次取流。因为第一次取流直接根据传入的分辨率匹配最合适分辨率，而重复取流需要根据当前取流失败的分辨率获取下一个支持的分辨率，然后再取流

返回值：

\* @returns {{audio: boolean, video: {frameRate: {exact: number}, width: {exact: number}, aspectRatio: {exact: number}, height: {exact: number}}}}

示例：

mediaDeviceInstance.getVideoConstraints (data, reTry)

### getNextConstraints

MediaDevice.prototype.getNextConstraints = function(data){}

取流失败后根据设备支持的能力列表获取下一个分辨率

参数 说明：

\* @param data = {

callback: ƒ (message)

constraints: {...}

constraintsKeyWord: "ideal"

error: {}

settings: {audio: false, video: {…}}

streamType: "video"

}

示例：

mediaDeviceInstance.getNextConstraints (data)

### getScreenShareConstraints

MediaDevice.prototype.getScreenShareConstraints = function(){}

获取屏幕共享分辨率

示例：

mediaDeviceInstance.getScreenShareConstraints ()

### getCapability

MediaDevice.prototype.getCapability = function(deviceId){}

根据deviceId 获取当前设备的设备支持能力列表，设备信息定时更新的，所以deviceId不会存在不匹配问题

参数说明：：

\* @param deviceId

示例：

mediaDeviceInstance.getCapability (deviceId)