

待到枫叶红成海

博客园

首页

新随笔

联系

订阅

管理

Hbuilder打包H5开发的App读取写入手机本地文件

在平时开发的html页面中，我们写的Js是没有读取用户电脑本地文件的权限的，这是出于浏览器运行时的安全考虑的，但在我们在使用h5打包app时，如果再像浏览器上让用户下载，上传文件，就会使用户的使用体验远不如电脑上操作，而且经常会有一些想要缓存本地的数据，cookies太小，满足不了需求，所以就想到能不能有内容时，我能直接操作手机的存取，我使用的是hbuilder打包app的，所以到官网上找找，就找到了有一节专门关于io的介绍，研究了一下使用方法，所以特此记录一下

本文所参考官方文档 https://www.html5plus.org/doc/zh_cn/io.html

先明确几个主要对象及方法，这里与java里的对照来看，会java的朋友可能会更容易理解

requestFileSystem 请求本地文件系统对象的方法，获取指定的文件系统，可通过type指定获取文件系统的类型。获取指定的文件系统对象成功通过succesCB回调返回，失败则通过errorCB返回。

- 参数：
- type: (Number) 必选 本地文件系统常量
 - 可取plus.io下的常量，如plus.io.PRIVATE_DOC、plus.io.PUBLIC_DOCUMENTS等。

公告

昵称： 待那枫叶红成海
园龄： 3年6个月
粉丝： 3
关注： 3
[+加关注](#)

< 2022年5月 >						
日	一	二	三	四	五	六
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4
5	6	7	8	9	10	11

搜索

找找看

谷歌搜索

我的标签

- java(3)
- web(2)
- javascript(1)

- succesCB: ([FileSystemSuccessCallback](#)) 必选 请求文件系统成功的回调
- errorCallback: ([FileErrorCallback](#)) 可选 请求文件系统失败的回调

注意，这里的type取值共4个，对应于不同的空间，也有不同的访问权限限制，一定要注意区分，如果读取或写入文件时报错时，记得检查这个地方， 以下为原文

为了安全管理应用的资源目录，规范对文件系统的操作，5+ API在系统应用目录的基础设计了应用沙盒目录，分为私有目录和公共目录两种类型，私有目录仅应用自身可以访问，公共目录在多应用环境时（如小程序SDK）所有应用都可访问。

- 应用私有资源目录，对应常量 [plus.io.PRIVATE_WWW](#)，仅应用自身可读
- 应用私有文档目录，对应常量 [plus.io.PRIVATE_DOC](#)，仅应用自身可读写
- 应用公共文档目录，对应常量 [plus.io.PUBLIC_DOCUMENTS](#)，多应用时都可读写，常用于保存应用间共享文件
- 应用公共下载目录，对应常量 [plus.io.PUBLIC_DOWNLOADS](#)，多应用时都可读写，常用于保存下载文件

DirectoryEntry 文件系统中的目录对象，用于管理特定的本地目录或者文件，对应于java中可以理解为File对象，这是操作文件最基础的对象，一个directoryEntry对象就对应于一个文件或文件夹

属性：

- **isFile**: 操作对象的是否为文件，DirectoryEntry对象固定其值为false
- **isDirectory**: 操作对象是否为目录，DirectoryEntry对象固定其值为true
- **name**: 目录操作对象的名称，不包括路径
- **fullPath**: 目录操作对象的完整路径，文件系统的绝对路径
- **fileSystem**: 文件操作对象所属的文件系统对象，参考FileSystem

方法：

- **getMetadata**: 获取目录的属性

Hbuilder打包H5 App(1)

随笔档案

- 2022年5月(1)
- 2022年3月(1)
- 2021年9月(1)
- 2020年11月(1)
- 2020年1月(1)
- 2019年7月(1)
- 2019年5月(1)

阅读排行榜

- 1. java web项目 使用elfinder 实现文件管理器(9025)
- 2. thymeleaf在js中取后台放在model中值的各种方式及区别(2894)
- 3. java web项目 使用elfinder 实现文件管理器 （二） (2183)
- 4. Hbuilder打包H5开发的App读取写入手机本地文件(2010)
- 5. spring security框架中，自定义登录页面和校验路径，CSRF防御引起的403 Forbidden(33)

评论排行榜

- 1. java web项目 使用elfinder 实现文件管理器(7)
- 2. java web项目 使用elfinder 实现文件管理器 （二） (3)

- moveTo: 移动目录
- copyTo: 拷贝目录
- toURL: 获取目录路径转换为URL地址
- toLocalURL: 获取目录路径转换为本地路径URL地址
- toRemoteURL: 获取目录路径转换为网络路径URL地址
- remove: 删除目录
- getParent: 获取目录所属的父目录
- createReader: 创建目录读取对象
- getDirectory: 创建或打开子目录
- getFile: 创建或打开文件
- removeRecursively: 递归删除目录

FileEntry 文件系统中的文件对象，用于管理特定的本地文件，对应于java中可以理解为File对象

属性:

- isFile: 文件操作对象的是否为文件，FileEntry对象固定其值为true
- isDirectory: 文件操作对象是否为目录，FileEntry对象固定其值为false
- name: 文件操作对象的名称，不包括路径
- fullPath: 文件操作对象的完整路径，文件系统的绝对路径
- fileSystem: 文件操作对象所属的文件系统对象，参考FileSystem

方法:

- getMetadata: 获取文件的属性信息
- moveTo: 移动文件
- copyTo: 拷贝文件
- toURL: 获取文件路径转换为URL地址
- toLocalURL: 获取文件路径转换为本地路径URL地址
- toRemoteURL: 获取文件路径转换为网络路径URL地址
- remove: 删除文件
- getParent: 获取文件所属的父目录
- createWriter: 获取文件关联的写文件操作对象 FileWriter
- file: 获取文件数据对象

推荐排行榜

- 1. java web项目 使用elfinder 实现文件管理器(2)
- 2. thymeleaf在js中取后台放在model中值的各种方式及区别(1)

最新评论

1. Re:java web项目 使用elfinder 实现文件管理器

按钮可以定制吗?

--咸鱼翻身

2. Re:java web项目 使用elfinder 实现文件管理器

博主，你预览怎么解决的?

--SophieRoyal

3. Re:java web项目 使用elfinder 实现文件管理器（二）

这个支持上传文件夹吗

--段志轩

4. Re:java web项目 使用elfinder 实现文件管理器（二）

@ SophieRoyal我这里是jsp页面，直接放在model里，前端EL表达式就可以取出来，其他各种模板引擎有自己的传值方法，这种一般很少需要单独请求吧...

--待那枫叶红成海

5. Re:java web项目 使用elfinder 实现文件管理器（二）

你最后一个菜单是通过什么请求来的？是单独写一个方法传到前端，然后再赋值?

--SophieRoyal

FileReader 创建读取文件对象，主要是文件读取相关的操作，文件以文本或者Base64编码的字符串形式读出来，对应于java中可理解为InputStream

属性：

- **readyState**: 当前读取文件所处的状态
- **result**: 已读取文件的内容
- **error**: 文件操作错误代码

方法：

- **abort**: 终止文件读取操作
- **readAsDataURL**: 以URL编码格式读取文件数据内容
- **readAsText**: 以文本格式读取文件数据内容

事件：

- **onloadstart**: 读取文件开始时的回调函数
- **onload**: 读取文件成功完成的回调函数
- **onabort**: 取消读取文件时的回调函数
- **onerror**: 文件读取操作失败时调用的回调函数
- **onloadend**: 文件读取操作完成时的回调函数

FileWriter 文件系统中的写文件对象，用于写入文件内容，用户注册自己的事件监听器来接收writestart、progress、write、writeend、error和abort事件，一个FileWriter对象是为单个文件的操作而创建，可以使用该对象多次对相应文件进行写入操作。FileWriter维护该文件的指针位置及长度属性，这样就可以寻找和写入文件的任何地方。默认情况下，FileWriter从文件的开头开始写入（将覆盖现有数据），seek方法可设置文件操作指定位置，如fw.seek(fw.length-1)写入操作就会从文件的末尾开始，对应于java中理解为OutputStream

属性：

- **readyState**: 当前写入文件所处的状态
- **length**: 文件当前的长度，单位为字节
- **position**: 文件当前操作的指针位置
- **error**: 文件写入操作错误代码

方法：

- **abort**: 终止文件写入操作

- seek: 定位文件操作位置
- truncate: 按照指定长度截断文件
- write: 向文件中写入数据

事件:

- onwritestart: 写入文件开始时的回调函数
- onwrite: 写入文件成功完成的回调函数
- onabort: 取消写入文件时的回调函数
- onerror: 文件写入操作失败时调用的回调函数
- onwriteend: 文件写入操作完成时的回调函数

下面是读写文件的主要代码及说明


1、获取文件读写的基础，FileEntry对象



```
1      //读取应用公共文档目录
2
plus.io.requestFileSystem(plus.io.PUBLIC_DOCUMENTS,
function(fs) {
3      // 通过fs.root获取DirectoryEntry对象进行操作,
    获取文件操作的根目录
4      //这里的filePath即你要读取的文件所在的相对路径,可
    随意定义,但不得是 _www开头,因为_www开头是应用私有资源目录的专
    用,只有读权限,不能写入,写入时会报错
5      var filePath = 'abc/haha/test.txt';
6      var rootDirectoryEntry = fs.root;
7      rootDirectoryEntry.getFile(filePath, {
8          //这个参数的作用是 指示如果文件或目录不存在时是
            否进行创建,默认值为false,设为true表示如果这个filePath下的
            test.txt文件不存在就创建,当然,如果存在就直接返回,不会创建
9          create: true
10     }, function(fileEntry) {
11         //FileEntry对象获取成功,对应就是test.txt文
            件了,可以接着进行相应的读写操作了
12
13     }, function(e) {
14         console.log(e.message);
15     });
16 }, function(e) {
17     console.log(e.message);
18 });
```



2、在上一步获取到文件操作对象FileEntry的基础上,开始读操作



```
1      //获取文件
2      fileEntry.file(function (file) {
3          //console.log(file.size + ' <--> ' +
            file.name);
```

```

4      //创建一个文件读取工具，在java中理解就是
InputStream输入流
5      var fileReader = new plus.io.FileReader();
6      //成功读取到文件内容时的回调，其中
evt.target.result就是文件中的文本内容
7      fileReader.onloadend = function (evt) {
8          console.log(evt.target.result);
9      }
10     //文件读取操作失败时调用的回调函数
11     fileReader.onerror = function (e) {
12         console.log(e.message);
13     }
14     //将刚才请求到的文件以utf-8编码，文本的形式读出
15     fileReader.readAsText(file, 'utf-8');
16 };

```



3、还是以文件操作对象FileEntry为基础，进行写入文件的操作

```

1      //通过fileEntry的createWriter创建输出流，向文件写入
内容，对应java中的OutputStream
2      fileEntry.createWriter(writer => {
3          //文件写入成功后的回调
4          writer.onwrite = function(event) {
5              //写入成功
6              console.log('写入成功');
7          }
8          //文件写入操作失败时调用的回调函数
9          writer.onerror = function(e) {
10             console.log(e.message);
11         }
12         //设置文件写入的起点，writer.length就是上次文件里
面内容的最后位置，这样即将新的内容追加到文本最末尾，如果想覆盖原
先的内容，直接设置为0即可
13         var cursor = writer.length;
14         writer.seek(cursor);
15         //将要写入的文本dataStr写入到文件中去
16         writer.write(dataStr);
17     }, function(e) {
18         console.log(e.message);
19     });

```



以上就是在Hbuilder中打包H5的App读写文件的主要代码了及功能了

但是我们发现这样写会有无数的回调函数，不仅不好看，别人调用时也不好传参，所以我们可以采用ES6的写法加上Primse来封装改造一下，让这些方法看起来更加优雅一点



```

1      getFileEntry(filePath) {
2          return new Promise((resolve, reject) => {

```

```
3
plus.io.requestFileSystem(plus.io.PUBLIC_DOCUMENTS, fs
=> {
4
    fs.root.getFile(filePath, {create:
true}, fileEntry => {
5
        resolve(fileEntry);
6
    }, e => {
7
        reject(e)
8
    });
9
}, e => {
10
    reject(e)
11
});
12
})
13
},
14
readData(fileEntry) {
15
    return new Promise((resolve, reject) => {
16
        fileEntry.file(function (file) {
17
            let fileReader = new
plus.io.FileReader();
18
            fileReader.onloadend = function
(evt) {
19
                resolve(evt.target.result);
20
            }
21
            fileReader.onerror = function (e) {
22
                reject(e)
23
            }
24
            fileReader.readAsText(file, 'utf-
8');
25
        });
26
    })
27
},
28
writeData(fileEntry, dataStr, cursorStart) {
29
    return new Promise((resolve, reject) => {
30
        fileEntry.createWriter(writer => {
31
            writer.onwrite = e => {
32
                resolve()
33
            }
34
            writer.onerror = e => {
35
                reject(e)
36
            }
37
            //设置文件写入的起点
38
            let length = cursorStart !=
undefined ? cursorStart : writer.length;
39
            writer.seek(length);
40
            writer.write(dataStr);
41
        }, e => {
42
            reject(e)
43
        });
44
    })
45
}
```

这样在调用时，代码就会非常简洁了

读取文件内容

```
1
let filePath = '/abc/haha/test.json';
2
this.getFileEntry(filePath)
```

```
3      .then(fileEntry => {
4          // fileEntry.remove();    //删除文件
5          return this.readData(fileEntry);
6      }).then(fileText => {
7          alert("文件内容:>>> " + fileText)
8      }).catch(e => {
9          alert('文件读取失败! ')
10     })
```

写入文件内容

```
1      let writeDataStr = 'Hello World!!!'
2      this.getFileEntry()
3      .then(fileEntry => {
4          return this.writeData(fileEntry,
writeDataStr);
5      }).then(writer => {
6          alert('保存成功');
7      }).catch(e => {
8          alert('保存失败! ')
9      })
```

OK ! 打完收工

标签: Hbuilder打包H5 App

好文要顶 关注我 收藏该文 待那枫叶红成海 的博客园



待那枫叶红成海
关注 - 3
粉丝 - 3

0 0

+加关注

« 上一篇: thymeleaf在js中取后台放在model中值的各种方式及区别
» 下一篇: Js常用小技巧收藏合集（一）持续更新中...

posted @ 2020-11-21 15:11 待那枫叶红成海 阅读(2010) 评论(0)
编辑 收藏 举报

刷新评论 刷新页面 返回顶部

登录后才能查看或发表评论, 立即 登录 或者 逛逛 博客园首页

【推荐】百度智能云开发者赋能计划, 云服务器4元起, 域名1元起
【推荐】华为开发者专区, 与开发者一起构建万物互联的智能世界

编辑推荐:

- 离谱的 CSS! 从表盘刻度到艺术剪纸
- 记将一个大型客户端应用项目迁移到 dotnet 6 的经验和决策
- .NET性能优化-使用结构体替代类
- 踩到一个关于分布式锁的非比寻常的BUG!
- 通过代码解释什么是API, 什么是SDK?

最新新闻:

- 抖音上市, 估值几何?
 - 大厂员工被高薪宠坏了?
 - 满帮股权曝光: 软银持股19.9% 张晖有77.6%投票权
 - 被露营支配后, 年轻人又对飞盘上瘾了
 - 奈飞的困境, 其实是一切“订阅制娱乐内容”的困境
- » 更多新闻...

Copyright © 2022 待那枫叶红成海
Powered by .NET 6 on Kubernetes