酷开系统
浏览器 js-sdk 文档
V0.2_linux

修改记录

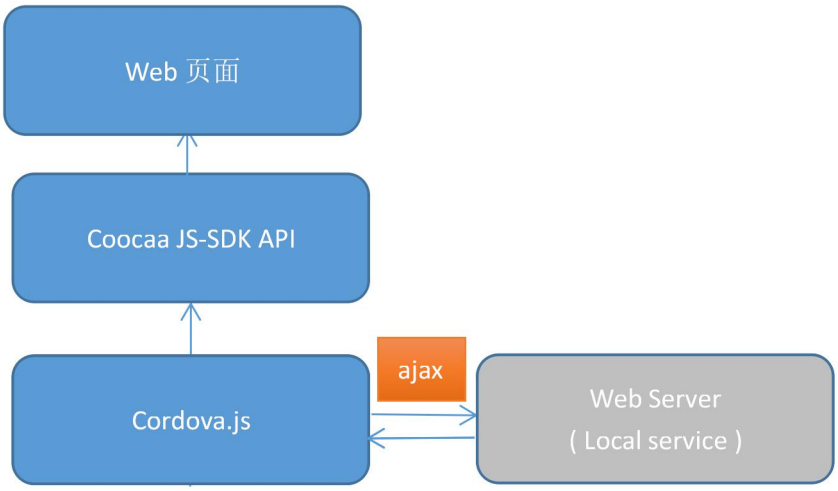
版本号	修改说明	修改日期	修改人
V0.1	初版发布	2019.5.24	
V0.2	精简接口	2019.6.18	

目录

一、概述

酷开平台 Web 页面开发接口，通过实现这套接口以支持酷开现有的 Web 页面，实现酷开系统切换到第三方浏览器的最小化改动目标。

1.1 酷开 Web 页面调用框图



第三方浏览器主要工作在：
在 Web Server 端接收 web 页面请求，并按指定格式返回结果。

1.2 酷开 Web 页面调用流程

第三方浏览器主要关注以下 Step 4 的实现：

Step 1. 在 Web 页面调用接口

获取设备信息：

```
coocaaosapi.getDeviceInfo(
  success(res) {
      console.log(res.panel);
      console.log(res.chip);
      console.log(res.activeid);
      ...
  }
)
```

期望正确时返回值如下：

```
{
```

```
        "panel": "32",
        "chip": "8S47",
        "chipid": "MST-6A338",
        "sid": "ebc9f795-788e-4c8f-b0cc-cd5371e112f3",
        "model": "E2A",
        "devid": "c86ae6c2352dd8b8d6624c6af6d8a5ad",
        "emmcid": "11010030303447363002129d2fa2a400",
        "activeid": "31140974",
        "brand": "Skyworth",
        "barcode": "32E2AXX-S093662-Z180516-406D9",
        "mac: "2835452aa239",
        "androidsdk": 19,
        "version": "7.00.190222"
}
```

Step 2. Coocaa JS-SDK API 层

```
//获取设备信息
CoocaaOSApi.prototype.getDeviceInfo= function(success, error) {
    argscheck.checkArgs('ff', 'CoocaaOSApi.getBaseInfo', arguments);
    exec(success, error, 'CoocaaOSApi', 'getDeviceInfo', []);
}
```

Step 3. Cordova.js 层

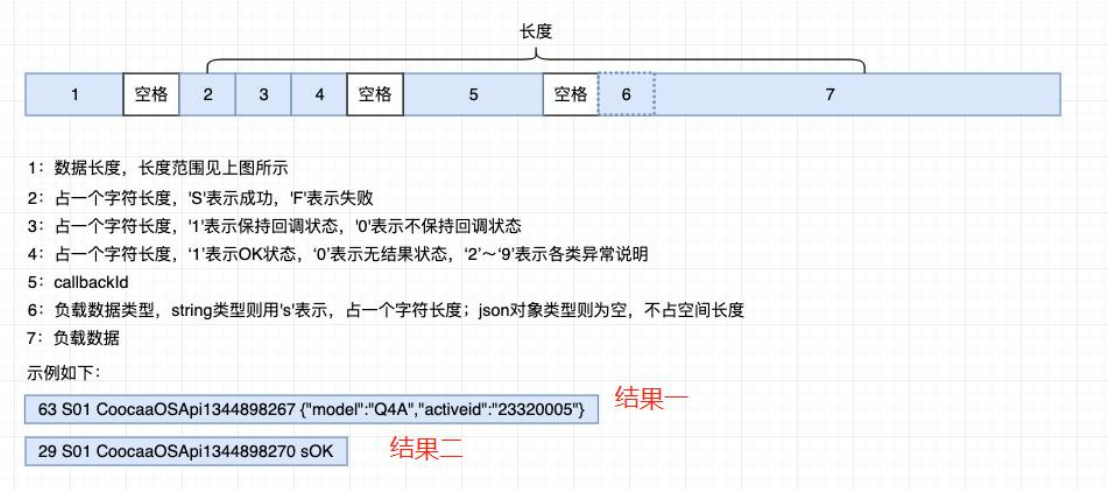
在 cordova.js `androidExec()`方法里通过 Ajax 调用本地 WebServer 接口：

```
    function androidExec(success, fail, service, action, args) {
        //参数转换...
        // Process any ArrayBuffers in the args into a string.
            for (var i = 0; i < args.length; i++) {
                if (utils.typeName(args[i]) == 'ArrayBuffer') {
                    args[i] = base64.fromArrayBuffer(args[i]);
                }
            }
        var callbackId = service + cordova.callbackId++;
        Var argsJson = JSON.stringify(args);

        If(service == “CoocaaOSApi”){
            Switch(action){
                Case “getDeviceInfo”:
                            $.ajax({
```

```
                        type: "post",
                        async: true,
                        timeout: 10000,
                        dataType: 'json',
                        url: serverUrl + "/getDeviceInfo",
                        data: {
                                "callback": callbackId,
                                "args": argsJson,
                        },
                        success: function(msg) {
                                console.log(JSON.stringify(data));

                        },
                        error: function(err) {
                                console.log(err);
                        },
                        complete: function(XMLHttpRequest, status) {
                                console.log(status);
                        }
                });
                Break;
            }
        }
        ...//省略部分代码
    }
```

Step 4. Web Server 端处理 web 页面请求

Web Server 端接收 web 页面请求，处理并返回响应结果；
响应结果必须按如下格式拼接后返回：



结果一说明：

63：表示返回数据总长度为 63（从上图中字段 2 开始到字段 7 的总长度）；

S01：S 表示接口执行成功；0 表示不保持回调状态； 1 表示 OK 状态；

CoocaaOSApi1344898267：是 web 页面请求时带的 callbackId，server 端直接使用即可；

{“model”:”Q4A”,”activeid”:”23320005”}: 表示负载数据，类型为 Json 类型；

结果二说明：

sOK：小写 s（必须为小写）表示负载数据类型为 String，

”OK”表示负载数据为字符串“OK”；

其它字段说明同结果一；

1.3 注册事件回调函数

Web 页面可能需要监听以下系统消息：

值	说明
NET_CHANGGED	网络状态变化事件
USER_CHANGGED	用户登录状态变化
COMMON_CHANGED	通用监听（播放器播放状态事件）

Web Server 需要提供事件触发时的通知机制。

addEventListener

注册指定事件监听

```
$.ajax({
    url: serverUrl + "/addEventListener",
    data: {
      "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
      "args": argsJson,
    }
});
```

其中：

```
var args = [“NET_CHANGGED”];
Var argsJson = JSON.stringify(args);
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。

期望返回值类型：String 字符串；

返回值格式如下：

```
“OK”或“Fail detail reason”
```

removeEventListener

删除事件监听函数：

```
$.ajax({
```

```
        url: serverUrl + "/removeEventListener",
        data: {
          "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
          "args": argsJson,
        }
    });
    其中:
     var args = [“NET_CHANGGED”];
     Var argsJson = JSON.stringify(args);
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。
期望返回值类型：String 字符串；
返回值格式如下：

```
“OK”或“Fail detail reason”
```

addGlobalEventListener

注册全局指定事件监听

```
    $.ajax({
        url: serverUrl + "/addGlobalEventListener",
        data: {
          "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
          "args": argsJson,
        }
    });
    其中:
     var args = [“NET_CHANGGED”];
     Var argsJson = JSON.stringify(args);
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。
期望返回值类型：String 字符串；
返回值格式如下：

```
“OK”或“Fail detail reason”
```

removeGlobalEventListener

删除全局事件监听

```
    $.ajax({
        url: serverUrl + "/removeGlobalEventListener",
        data: {
          "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
          "args": argsJson,
        }
    });
    其中:
```

```
    var args = [“NET_CHANGGED”];
    Var argsJson = JSON.stringify(args);
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。
期望返回值类型：String 字符串；
返回值格式如下：

```
“OK”或“Fail detail reason”
```

二、接口定义

以下是目前使用到的接口说明，分模块介绍，可能会有部分接口尚未添加进来，后续版本持续完善。

2.1 系统接口

coocaaosapi.getAppInfo()

获取本机应用程序版本信息

```
$.ajax({
    url: serverUrl + "/getAppInfo",
    data: {
      "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
      "args": argsJson,
    }
});
其中：
 var packageName = '{"pkgList":["com.tianci.user","com.coocaa.mall"]}'
 var args = [{ 'pkgList': packageName}];
 Var argsJson = JSON.stringify(args);
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。
期望返回值类型：json 字符串；
返回值格式如下：

属性	类型	说明
versionCode	Number	
Status	String	0:ok
versionName	String	

返回值 JSON 字符串示例：

```
    {\"com.coocaa.mall\":{\"versionCode\":31100003,\"status\":\"0\",\"versionName\":\
"3.11.3[CC][1904030048]\"},\"com.tianci.user\":{\"versionCode\":4100018,\"status\":\"0
\",\"versionName\":\"4.10.18\"}}
```

coocaaosapi.getPropertiesValue()

获取系统指定属性值。

```
$.ajax({
    url: serverUrl + "/getPropertiesValue",
    data: {
      "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
      "args": argsJson,
    }
});
其中：
 var args = [{'propertiesKey': “third.get.4k”}]; //是否支持 4K
 Var argsJson = JSON.stringify(args);
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。

期望返回值类型：json 字符串；

返回值格式如下：

```
{"propertiesValue":"0"}
```

coocaaosapi.startBlueToothSetting()

启动蓝牙设置。

```
$.ajax({
    url: serverUrl + "/startBlueToothSetting",
    data: {
      "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
      }
});
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。

期望返回值类型：String 字符串；

返回值格式如下：

```
“OK”
```

coocaaosapi.startLocalMedia()

启动本地媒体库（U 盘首页）。

```
$.ajax({
    url: serverUrl + "/startLocalMedia",
    data: {
      "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
      }
});
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。

期望返回值类型：String 字符串；

返回值格式如下：

```
“OK”
```

coocaaosapi.startSourceList()

启动电视信号源列表。

```
$.ajax({
    url: serverUrl + "/launchSourceList",
    data: {
      "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
      }
});
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。
期望返回值类型：String 字符串；
返回值格式如下：

```
“OK”
```

coocaaosapi.startTVSetting()

启动电视设置。

```
$.ajax({
    url: serverUrl + "/startTVSetting",
    data: {
      "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
      "args": argsJson,
    }
});
其中：
 var args = [["com.tianci.setting", "com.tianci.setting.TianciSetting"]];
 Var argsJson = JSON.stringify(args);
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。
期望返回值类型：String 字符串；
返回值格式如下：

```
“OK”
```

coocaaosapi.startNetSetting()

启动网络设置。

```
$.ajax({
    url: serverUrl + "/startNetSetting",
    data: {
      "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
      }
});
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。

期望返回值类型：String 字符串；
返回值格式如下：

```
“OK”
```

2. 2 网络接口

coocaaosapi.isNetConnected()

获取网络连接状态。

```
$.ajax({
    url: serverUrl + "/isNetConnected",
    data: {
      "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
      }
});
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。
期望返回值类型：json 字符串；
返回值格式如下：

属性	类型	说明
isConnected	String	“true”/”false”：当前是否有网络连接

```
{"isConnected":"true"}
```

coocaaosapi.getNetType()

获取网络类型。

```
$.ajax({
    url: serverUrl + "/getNetType",
    data: {
      "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
    }
});
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。
期望返回值类型：json 字符串；

属性	类型	说明
networkType	string	网络类型（wifi/cable/...）

```
{"networkType":"wifi"}
```

coocaaosapi.getIpInfo()

获取网络参数。

```
$.ajax({
    url: serverUrl + "/getIpInfo",
```

```
        data: {
          "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
          }
    });
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。

期望返回值类型：json 字符串；

返回值格式如下：

属性	类型	说明
ip	string	
Mac	String	
Gateway	String	
Netmask	String	
Dns0	String	
Dns1	String	

返回值示例代码：

```
返回值：
{
          "dns1": "172.20.135.2",
          "mac": "60:42:7f:e1:31:52",
          "gateway": "172.20.146.1",
          "netmask": "255.255.255.0",
          "dns0": "172.20.135.1",
          "ip": "172.20.146.147"
}
```

coocaaosapi.addNetChangedListener(function callback)

网络状态变更事件监听

监听到网络状态变化事件时的回调函数参数类型： JSON；

返回值格式如下：

属性	类型	说明
isConnected	boolean	当前是否有网络连接
networkType	string	网络类型（wifi/cable/...）

2.4 多媒体接口

coocaaosapi.startCommonWebview()

启动 web 播放器。

```
    $.ajax({
        url: serverUrl + "/startCommonWebview",
```

```
      data: {
        "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
        "args": argsJson,
      }
  });
  其中：
  var args = [
        ["action", "app_browser.intent.action.PLAYER", "com.coocaa.app_browser"],
        [{ "extra.id": id }, { "extra.uri": uri }, { "extra.tips": tips },
          { "extra.height": height },{ "extra.width": width },
          { "extra.http_call_url": call_url }, { "extra.type": type },
          { "extra.name": name }]];
   Var argsJson = JSON.stringify(args);
```

args 参数说明：

属性	类型	默认值	必填	说明
id	String	-	否	
uri	string	-	是	
name	string	-	否	
Height	string	-	否	
width	string	-	否	
Call_url	String		否	
type	String		否	
name	String		否	

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。
期望返回值类型：String 字符串；
返回值格式如下：

```
    “OK”
```

coocaaosapi.addCommonListener(function callback)

web 播放器播放状态监听

监听到网络状态变化时的回调函数参数类型：JSON 字符串；
返回值格式如下：

属性	类型	说明
web_player_event	String	网络播放器播放状态

res.web_player_event 的合法值

值	说明
on_start	网络播放器开始播放
on_complete	网络播放器播放完成
on_interrupt	网络播放器播放中断（按返回键等）
on_error	网络播放器播放失败

示例代码：

```
coocaaosapi.addCommonListener(
```

```
  function(res) {
    if(res.web_player_event == "on_start") {
        //codes...
    }else if(message.web_player_event == "on_complete") {
        //codes...
    }else if(message.web_player_event == "on_interrupt") {
        //codes...
    }
  }
)
```

2.5 设备信息

coocaaosapi.getBaseInfo()

获取本机磁盘和内存信息。

```
    $.ajax({
        url: serverUrl + "/getSpaceInfo",
        data: {
          "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
        }
    });
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。

期望返回值类型：json 字符串；

返回值格式如下：

属性	类型	说明
freeSpace	number	可用硬盘空间
totalSpace	number	总共硬盘空间
totalMem	number	总共内存大小
leftMem	number	可用内存大小

返回值 JSON 字符串示例：

```
    {
        "freeSpace": 630980608,
        "totalMem": 484253696,
        "totalSpace": 2039562240,
        "leftMem": 111505408
    }
```

coocaaosapi.getDeviceInfo(Object object)

获取设备信息(屏幕大小/机芯机型/激活 ID 等）。

```
    $.ajax({
```

```
        url: serverUrl + "/getDeviceInfo",
        data: {
          "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
          }
    });
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。

期望返回值类型：json 字符串；

返回值格式如下：

属性	类型	说明
panel	String	本机屏幕尺寸
chip	String	本机机芯型号
chipid	String	本机主芯片型号
sid	String	
model	String	本机机型
devid	String	设备 ID
emmcid	String	EMMC ID
activeid	String	激活 ID
brand	String	品牌
barcode	String	二维码
mac	String	MAC 地址
androidsdk	number	Android SDK 版本
version	String	本机软件系统版本

返回值 JSON 字符串示例：

```
    {
        "panel": "32",
        "chip": "8S47",
        "chipid": "MST-6A338",
        "sid": "ebc9f795-788e-4c8f-b0cc-cd5371e112f3",
        "model": "E2A",
        "devid": "c86ae6c2352dd8b8d6624c6af6d8a5ad",
        "emmcid": "11010030303447363002129d2fa2a400",
        "activeid": "31140974",
        "brand": "Skyworth",
        "barcode": "32E2AXX-S093662-Z180516-406D9",
        "mac: "2835452aa239",
        "androidsdk": 19,
        "version": "7.00.190222"
    }
```

2.6 用户信息

coocaaosapi.hasCoocaaUserLogin()

用户是否登录。

```
$.ajax({
    url: serverUrl + "/hasCoocaaUserLogin",
    data: {
      "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
      }
});
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。

期望返回值类型：json 字符串；

返回值格式如下：

属性	类型	说明
haslogin	String	“true/false”是否已登录状态

返回值 JSON 字符串示例：

```
{"haslogin":”true”}
```

coocaaosapi.getUserInfo()

获取用户信息。

```
$.ajax({
    url: serverUrl + "/getUserInfo",
    data: {
      "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
    }
});
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。

期望返回值类型：json 字符串；

返回值格式如下：

属性	类型	说明
sky_id	String	
update_image_times	String	
open_id	String	
visit_num	String	
loginFlag	String	
external_info	Array.<object>	
Avatar	String	
tel1	String	
nick_name	String	
cardNo	String	
Balance	String	
create_date	String	
Email	String	
last_name	String	

Current	String	
Gender	String	
Avatars	Object	
third_account	String	
Qq	String	
Revenue	String	
Mobile	String	
Account_type	String	

object.external_info 参数

属性	类型	说明
external_flag	String	qq/weixin
Login	Boolean	是否登录
Vusession	String	
Unionid	String	
bind_time	Number	
external_nickname	String	
external_id	String	
Vuserid	String	
external_avatar	String	

object.avatars 参数

属性	类型	说明
f_50	String	
f_70	String	
f_200	String	
f_250	String	
f_370	String	
f_500	String	
f_800	String	

返回值如下（demo）：


```
    {
          "sky_id": "200000000474",
          "update_image_times": "0",
          "open_id": "8151266f4ede11e6987500505687790a",
          "visit_num": "566",
          "loginFlag": "",
          "external_info":
"[{\"external_avatar\":\"\",\"vuserid\":\"\",\"external_id\":\"5E15FF5348E0B462797E4F4
965B640C3\",\"external_nickname\":\"\",\"bind_time\":1469062720000,\"vusession\":\"\",
\"login\":false,\"external_flag\":\"qq\"},{\"external_avatar\":\"http://thirdwx.qlogo.
cn/mmopen/vi_32/DYAIOgq83erkBY5Jun4VqFRlXgcNXfJ0CFWIUfBIkdV6b7CcttYy8zsWfgWzRJewDricGN
bO7yjmx9kcwiaIZlwA/132\",\"vuserid\":\"530287424\",\"external_id\":\"o-G_Utw3KfMLFKe6s
UcaVoIx9mPY\",\"external_nickname\":\"北纬 36 度
```

```
\",\"bind_time\":1518143947000,\"unionid\":\"o6sbmwL5YwqJLZs2318xFcRbagWg\",\"vusessio
n\":\"9b71d60e144de9ece3d6\",\"login\":false,\"external_flag\":\"weixin\"}]",
         "avatar":
"http://thirdwx.qlogo.cn/mmopen/vi_32/DYAIOgq83erkBY5Jun4VqFRlXgcNXfJ0CFWIUfBIkdV6b7Cc
ttYy8zsWfgWzRJewDricGNbO7yjmx9kcwiaIZlwA/132",
         "tel1": "",
         "nick_name": "          1",
         "cardNo": "1101415325916",
         "balance": "0",
         "create_date": "1469062824000",
         "email": "",
         "last_name": "          ",
         "current": "true",
         "gender": "1",
         "avatars":
"{\"f_70\":\"http://thirdwx.qlogo.cn/mmopen/vi_32/DYAIOgq83erkBY5Jun4VqFRlXgcNXfJ0CFWI
UfBIkdV6b7CcttYy8zsWfgWzRJewDricGNbO7yjmx9kcwiaIZlwA/132\",\"f_50\":\"http://thirdwx.q
logo.cn/mmopen/vi_32/DYAIOgq83erkBY5Jun4VqFRlXgcNXfJ0CFWIUfBIkdV6b7CcttYy8zsWfgWzRJewD
ricGNbO7yjmx9kcwiaIZlwA/132\",\"f_800\":\"http://thirdwx.qlogo.cn/mmopen/vi_32/DYAIOgq
83erkBY5Jun4VqFRlXgcNXfJ0CFWIUfBIkdV6b7CcttYy8zsWfgWzRJewDricGNbO7yjmx9kcwiaIZlwA/132\
",\"f_370\":\"http://thirdwx.qlogo.cn/mmopen/vi_32/DYAIOgq83erkBY5Jun4VqFRlXgcNXfJ0CFW
IUfBIkdV6b7CcttYy8zsWfgWzRJewDricGNbO7yjmx9kcwiaIZlwA/132\",\"f_500\":\"http://thirdwx.
qlogo.cn/mmopen/vi_32/DYAIOgq83erkBY5Jun4VqFRlXgcNXfJ0CFWIUfBIkdV6b7CcttYy8zsWfgWzRJew
DricGNbO7yjmx9kcwiaIZlwA/132\",\"f_250\":\"http://thirdwx.qlogo.cn/mmopen/vi_32/DYAIOg
q83erkBY5Jun4VqFRlXgcNXfJ0CFWIUfBIkdV6b7CcttYy8zsWfgWzRJewDricGNbO7yjmx9kcwiaIZlwA/132
\",\"f_200\":\"http://thirdwx.qlogo.cn/mmopen/vi_32/DYAIOgq83erkBY5Jun4VqFRlXgcNXfJ0CF
WIUfBIkdV6b7CcttYy8zsWfgWzRJewDricGNbO7yjmx9kcwiaIZlwA/132\"}",
         "third_account": "1",
         "qq": "",
         "revenue": "0",
         "mobile": "13728755689",
         "account_type": "1"
    }
```


coocaaosapi.getUserAccessToken()

获取用户 access_token。

```
    $.ajax({
        url: serverUrl + "/getUserAccessToken",
        data: {
          "callback": callbackId,//是 web 页面请求时自动生成的 callbackId;
          }
    });
```

success 回调函数返回参数需要满足 1.2 节 Step 4 的要求。

期望返回值类型：json 字符串；
返回值格式如下：

属性	类型	说明
accesstoken	String	

返回值格式示例：

```
{“accesstoken”: “2.b290780935a64bb0aed89fe8f509cxx”}
```

coocaaosapi.addUserChanggedListener()

账户状态变更事件监听

监听到用户登录状态变化时的回调函数参数类型：JSON 字符串
返回值格式如下：

属性	类型	说明
haslogin	string	

示例代码：

```
{“haslogin”: “false”}
```