## 1.QQ互联网认证OA信息，进行uth2.0 JS

**功能描述：**通过qq认证获取qq账号信息，进行用户登录

### 步骤：1引用JS SDK的JavaScript文件

1.1. 首先需要申请接入qq登录，并成功获取到appid和appkey。  
1.2. 在html页面适当的位置引入JS脚本包（将下面代码中的“APPID”替换为申请接入QQ登录时获得的appid；"REDIRECTURI"替换为申请接入QQ登录时输入的回调地址）：

<script type="text/javascript"  
src=<http://qzonestyle.gtimg.cn/qzone/openapi/qc_loader.js> data-appid="APPID" data-redirecturi="REDIRECTURI" charset="utf-8"></script>

2.放置qq登录按钮

<span id="qqLoginBtn"></span>

<script type="text/javascript">

QC.Login({

btnId:"qqLoginBtn" //插入按钮的节点id

});

</script>

然后就会出现以下这个按钮

Connect_logo_7.png

若需要对登录按钮进行设置，请[自定义登录按钮](#_3._自定义登录按钮)。将修改后的代码粘贴到页面中放置登录按钮处。

3.回传Access Token 和 openId

该步骤的作用是回调地址将获取到的Access Token和OpenID返回给调用页面。

在回调地址页面，即1.1节中回调地址“REDIRECTURI”指定的页面，粘贴如下代码：

<script type="text/javascript"  
src="http://qzonestyle.gtimg.cn/qzone/openapi/qc\_loader.js" charset="utf-8" data-callback="true"></script>

4.调用QQ登录openApi

调用方式说明：

QQ互联在JS SDK中封装了所有的OpenAPI接口，开发者只需要传递OpenAPI名称，以及OpenAPI需要的相关参数，就可以调用OpenAPI。  
调用OpenAPI时，请统一遵循下述调用方式：

QC.api(api, paras, fmt, method):Request

**函数说明**：  
用JS SDK协助调用OpenAPI。

**参数说明**：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **是否必须** | **描述** |
| **api** | 必须 | 指定要调用的OpenAPI名称。例如：调用[website/add\_share](http://wiki.open.qq.com/wiki/website/add_share)时，OpenAPI名称为“add\_share”。  各OpenAPI的名称具体请参见[API列表](http://wiki.open.qq.com/wiki/website/API%E5%88%97%E8%A1%A8)。 |
| **paras** | 必须 | 指定要调用的OpenAPI对应的参数。各参数使用JSON的键值对格式列出。  OpenAPI对应的参数具体请参见[API列表](http://wiki.open.qq.com/wiki/website/API%E5%88%97%E8%A1%A8)中各OpenAPI的参数说明。 |
| **fmt** | 可选 | 指定OpenAPI的返回格式，可用值为“JSON”或“XML”。默认为“JSON”。 |
| **method** | 可选 | 指定OpenAPI调用请求的发起方式，可用值为“GET”或“POST”。根据配置，默认发送数据为“POST”，获取数据为“GET”。 |

## 2. JS SDK的其他公开方法

### 2.1. 直接打开QQ登录弹窗

QC.Login.showPopup(oOpts):Window

**参数说明**：  
oOpts：需要指定appId，回调地址redirect\_URI。  
参数示例如下：

oOpts：{

appId:"222222",

redirectURI:"http://yousite.com/qc\_back.html"

}

**返回值说明**：  
返回浏览器弹窗对象。

**函数使用示例**：

QC.Login.showPopup({

appId:"222222",

redirectURI:"http://yousite.com/qc\_back.html"

});

### 2.2. 注销当前登录用户

QC.Login.signOut():void

### 2.3. 获取当前登录用户的Access Token以及OpenID

QC.Login.getMe(function(openId, accessToken){}):void

**参数说明**：  
这里的参数为回调函数，通过回调函数获取openId和accessToken。  
openId：用户身份的唯一标识。建议保存在本地，以便用户下次登录时可对应到其之前的身份信息，不需要重新授权。  
accessToken：表示当前用户在此网站/应用的登录状态与授权信息，建议保存在本地。

### 2.4. 检测当前登录状态

QC.Login.check():Boolean

**返回值说明**：  
true：说明登录成功。  
false：说明登录失败。

## 3. 自定义登录按钮

开发者可以根据自己的需要，自定义登录按钮，示例如下：

<span id="qqLoginBtn"></span>

<script type="text/javascript">

//调用QC.Login方法，指定btnId参数将按钮绑定在容器节点中

QC.Login({

//btnId：插入按钮的节点id，必选

btnId:"qqLoginBtn",

//用户需要确认的scope授权项，可选，默认all

scope:"all",

//按钮尺寸，可用值[A\_XL| A\_L| A\_M| A\_S| B\_M| B\_S| C\_S]，可选，默认B\_S

size: "A\_XL"

}, function(reqData, opts){//登录成功

//根据返回数据，更换按钮显示状态方法

var dom = document.getElementById(opts['btnId']),

\_logoutTemplate=[

//头像

'<span><img src="{figureurl}" class="{size\_key}"/></span>',

//昵称

'<span>{nickname}</span>',

//退出

'<span><a href="javascript:QC.Login.signOut();">退出</a></span>'

].join("");

dom && (dom.innerHTML = QC.String.format(\_logoutTemplate, {

nickname : QC.String.escHTML(reqData.nickname),

figureurl : reqData.figureurl

}));

}, function(opts){//注销成功

alert('QQ登录 注销成功');

}

);

</script>

开发者也可以使用下列函数作为登录成功的回调接收函数：  
① 登录成功的回调函数：

cbLoginFun(oInfo, oOpts)

参数说明：  
oInfo：当前登录用户的基本信息，即OpenAPI中[get\_user\_info](http://wiki.open.qq.com/wiki/website/get_user_info)返回的数据。  
参数示例如下：

oInfo：{

"ret":0,

"msg":"",

"nickname":"遲來の垨堠",

"figureurl":"http://qzapp.qlogo.cn/qzapp/100229030/ECCC463F76A2E3C1D9217BBC30418164/30",

"figureurl\_1":"http://qzapp.qlogo.cn/qzapp/100229030/ECCC463F76A2E3C1D9217BBC30418164/50",

"figureurl\_2":"http://qzapp.qlogo.cn/qzapp/100229030/ECCC463F76A2E3C1D9217BBC30418164/100",

"gender":"男",

"vip":"1",

"level":"7"

}

oOpts：回传初始化参数，多个按钮时可用来区分来源，用来区分一个页面多个登录按钮的情况。  
参数示例如下：

oOpts：{

btnId:"qqLoginBtn"

}

回调函数使用示例如下：

var cbLoginFun = function(){};  
QC.Login({btnId:"qqLoginBtn"}, function(oInfo, oOpts){

alert(oInfo.nickname) //昵称

});

② 注销成功的回调函数：

cbLogoutFun()

## 4. Request与Response内置对象说明

### 4.1 Request

JS SDK在初始化时会根据浏览器环境创建不同的请求代理，QC.api的每次调用都是一个Request对象。  
  
Request对象的公开方法如下：  
.success(resp): Request：请求完成并且返回码为0的回调。  
.error(resp): Request：请求完成并且返回码不为0的回调。  
.complete(resp): Request：请求完成后的回调。

调用时序为success/error -> complete，每个方法都可以调用多次，每次调用返回Request本身，支持链式调用。  
resp参数为回调函数，回调函数参数为Response对象。

### 4.2. Response

Response为Request对象绑定的回调函数的返回参数，每次QC.api调用的异步响应都会返回一个Response对象，该对象在Request对象的success/error -> complete调用流程中传递。  
  
Response的公开方法如下：  
.stringifyData：返回该Response对象包含的数据体的文本串。  
  
Response的公开属性如下：  
.status：响应状态，-1：代表未知；404：响应错误；200：响应成功。  
.fmt：响应数据格式，json/xml。  
.code/.ret：响应返回码，0为成功，其他为失败。  
.data：响应数据实体，json对象/xml对象。  
.seq：响应序号，从1000开始编号。

## 2.QQ互联网认证OA信息，进行OAuth2.0 JAVA

**功能描述：**通过qq认证获取qq账号信息，进行用户登录

**方法定义**：getQQOAthUrl(String appid,String returnUrl)

输入参数：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **类别** | **参数名称** | **类型** | **是否可空** | **说明** |
| 输入 | appid | string | 不可空 | Qq申请的appid |
| 输入 | returnUrl | string | 不可空 | 回调地址 |

返回 一个url，例如https://graph.qq.com/oauth/show?which=error&display=pc&error=100008&client\_id=appid&redirect\_uri= returnUrl…可直接重定向到这个网址，授权页面

## QQ登录成功回调页面获取用户信息

**功能描述：**回调页面获取用户信息

**方法定义**：getQQUserInfo(HttpServletRequest request,appid, appkey,string returnUrl)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **类别** | **参数名称** | **类型** | **是否可空** | **说明** |
| 输入 | request | HttpServletRequest | 不可空 | 回调页面请求的参数（qq发来的通知消息参数） |
| 输入 | appid | string | 不可空 | 申请QQ登录成功后，分配给应用的appid。 |
| 输入 | appkey | string | 不可空 | 申请QQ登录成功后，分配给网站的appkey。 |
| 输入 | returnUrl | string | 不可空 | 成功授权后的回调地址，必须是注册appid时填写的主域名下的地址，建议设置为网站首页或网站的用户中心。注意需要将url进行URLEncode。 |

**返回结果说明：**返回结果QQUserInfoDTO

|  |  |  |  |
| --- | --- | --- | --- |
| **类别** | **属性名称** | **类型** | **说明** |
| 输出 | ret | string | 返回码 |
| 输出 | msg | string | 如果ret<0，会有相应的错误信息提示，返回数据全部用UTF-8编码。 |
| 输出 | nickName | string | 用户在QQ空间的昵称。 |
| 输出 | figureUrl\_30 | string | 大小为30×30像素的QQ空间头像URL。 |
| 输出 | figureUrl\_50 | string | 大小为50×50像素的QQ空间头像URL。 |
| 输出 | figureUrl\_100 | string | 大小为100×100像素的QQ空间头像URL。 |
| 输出 | figureUrl\_qq\_40 | string | 大小为40×40像素的QQ头像URL。 |
| 输出 | figureUrl\_qq\_100 | string | 大小为100×100像素的QQ头像URL。需要注意，不是所有的用户都拥有QQ的100x100的头像，但40x40像素则是一定会有。 |
| 输出 | gender | string | 性别。 如果获取不到则默认返回"男" |
| 输出 | isYellowVip | int | 标识用户是否为黄钻用户（0：不是；1：是）。 |
| 输出 | vip | int | 标识用户是否为黄钻用户（0：不是；1：是）？qq文档上的，感觉像qq会员 |
| 输出 | yellowVipLevel | int | 黄钻等级 |
| 输出 | level | int | 黄钻等级?qq文档是这这么说的，感觉像vip等级 |
| 输出 | isYellowYearVip | int | 标识是否为年费黄钻用户（0：不是； 1：是） |