IOS SDK API 使用说明

1. SSO 安全登录 QQ 帐号

iOS SDK 支持应用跳转到手机 QQ 进行登录，给用户提供更加安全、快捷的体验 。

如果用户没有安装手机 QQ，且开发者具有 webview 权限，则显示登录页；如果开发者没有 webview 权限，sdk 版本大于等于 2.9，则显示登录页，sdk 版本小于 2.9，则显示下载页。由于跳转至下载页在当前苹果 app 审核有被拒风险，所以，希望开发者尽快升级是用最新版 sdk。

1.1 授权登录

整个授权过程可以分为，发送请求获取用户授权以及获取用户授权结果两个部分。

✓ 发送请求获取用户授权：

在取得用户授权之前，首先必须清楚自己需要用户的哪些信息，iOS SDK 提供多种选择，开发者可以根据自己的需要请求用户不同信息的授权。具体可以获取的授权信息参见 1.2 获取用户授权参数说明。

在设置完需要请求的授权信息之后，就可以发送请求了（备注：inSafari 参数从 iOS SDK1.3 版本后废除）。

```
NSArray* permissions = [NSArray arrayWithObjects:
                        kOPEN_PERMISSION_GET_USER_INFO,
                        kOPEN_PERMISSION_GET_SIMPLE_USER_INFO,
                        kOPEN_PERMISSION_ADD_SHARE,
                        nil];
TencentOAuth *_tencentOAuth =
                        [TencentOAuth initWithAppid:appid andDelegate:delegate];
[_tencentOAuth authorize:permissions inSafari:NO];
```

其中_tencentOAuth 为 TencentOAuth 类的实例。

✓ 获取用户授权结果

获取用户授权结果，采用的是代理回调的方式，所以开发者需要实现 TencentLoginDelegate 协议，登录结果分为三种：登录成功、登录失败和登录取消。

```
- (void)tencentDidLogin;   // 登录成功后的回调
- (void)tencentDidNotLogin:(BOOL)cancelled;   // 登录失败后的回调
- (void)tencentDidNotNetWork;   // 登录时网络有问题的回调
```

如果登录成功，则开发者可以通过实现 TencentLoginDelegate 协议中的以下方法获取：

```
// 登录时权限信息的获得
- (NSArray *)getAuthorizedPermissions:(NSArray *)permissions
                        withExtraParams:(NSDictionary *)extraParams;
```

1.2 获取用户授权参数说明

字符串	含义
kOPEN_PERMISSION_GET_USER_INFO	获取用户信息
kOPEN_PERMISSION_GET_SIMPLE_USER_INFO	移动端获取用户信息
kOPEN_PERMISSION_GET_INFO	获取登录用户自己的详细信息
kOPEN_PERMISSION_GET_VIP_RICH_INFO	获取会员用户详细信息
kOPEN_PERMISSION_GET_VIP_INFO	获取会员用户基本信息
kOPEN_PERMISSION_GET_OTHER_INFO	获取其他用户的详细信息
kOPEN_PERMISSION_ADD_TOPIC	发表一条说说到 QQ 空间（**需要申请权限**）
kOPEN_PERMISSION_ADD_ONE_BLOG	发表一篇日志到 QQ 空间（**需要申请权限**）
kOPEN_PERMISSION_ADD_ALBUM	创建一个 QQ 空间相册（**需要申请权限**）
kOPEN_PERMISSION_UPLOAD_PIC	上传一张照片到 QQ 空间相册（**需要申请权限**）
kOPEN_PERMISSION_LIST_ALBUM	获取用户 QQ 空间相册列表（**需要申请权限**）
kOPEN_PERMISSION_ADD_SHARE	同步分享到 QQ 空间、腾讯微博
kOPEN_PERMISSION_CHECK_PAGE_FANS	验证是否认证空间粉丝

iOS SDK V2.9.5 废除了腾讯微博相关的接口，关于废除的参数，特此说明

字符串（已废除）	含义
kOPEN_PERMISSION_ADD_PIC_T	上传图片并发表消息到腾讯微博
kOPEN_PERMISSION_DEL_T	删除一条微博信息
kOPEN_PERMISSION_GET_REPOST_LIST	获取一条微博的转播或评论信息列表
kOPEN_PERMISSION_GET_FANSLIST	获取登录用户的听众列表
kOPEN_PERMISSION_GET_IDOLLIST	获取登录用户的收听列表
kOPEN_PERMISSION_ADD_IDOL	收听腾讯微博上的用户
kOPEN_PERMISSION_DEL_IDOL	取消收听腾讯微博上的用户
kOPEN_PERMISSION_GET_INTIMATE_FRIENDS_WEIBO	获取微博中最近 at 的好友
kOPEN_PERMISSION_MATCH_NICK_TIPS_WEIBO	获取微博中匹配昵称的好友

2. 分享到 QQ 和 QQ 空间

2.1 分享到 QQ 好友

在用户安装了手机 QQ 时通过手机 QQ 进行分享，否则调用浏览器页面进行分享。

其中文本消息，图文消息和音频消息的 title 是必须的，summary 可以不填，具体调用请参考 2.3 分享示例代码。

使用分享到 QQ 好友功能需要设置 QQ 业务回调，请参考 6.处理 QQ 业务的回调。

2.2 分享到 QQ 空间

分享到 QQ 空间的接口用于取代老的分享接口 addShareWithParams（该接口已经废弃）。

在用户安装了手机 QQ（4.6 版本以上）时通过手机 QQ 中的 QZone 结合版进行分享，否则调用浏览器页面进行分享。分享时调用浏览器页面进行分享。其中 title 是必须的，summary 可以不填，具体调用请参考 2.3 分享示例代码。使用分享到 QQ 空间功能需要设置 QQ 业务回调，请参考 6.处理 QQ 业务的回调。

在分享到 QQ 好友和 QQ 空间的时候，根据是本地分享还是浏览器中的分享，支持分享的消息类型不同。

因为 webQQ 好友分享和 web QQ 空间的分享都不支持非 URL 类型的分享，所以这里建议在分享到 QQ 好友或者 QQ 空间的时候尽量避免这两种类型的调用，避免发生不支持的错误。

分享消息类型	QQ 好友	QQ 空间	web QQ 好友	web QQ 空间
QQApiTextObject	支持	不支持	不支持	不支持
QQApiImageObject	支持	不支持	不支持	不支持
QQApiNewsObject	支持	支持	支持	支持
QQApiAudioObject	支持	支持	支持	支持
QQApiVideoObject	支持	支持	支持	支持
QQApiGroupTribeImageObject	仅群部落	不支持	不支持	不支持
QQApiAddFriendObject	游戏好友	不支持	不支持	不支持
QQApiFileObject	仅数据线	不支持	不支持	不支持
QQApiGameConsortiumBindingGroupObject	仅群部落	不支持	不支持	不支持

2.3 分享示例代码（详细请参见 SDK 中 Demo）

下面是各种分享消息的实例代码，作为开发者调用 QQ 好友分享和 QQ 空间分享的参考：（注：如果可以分享到 QQ 空间，则会在示例代码中给出；不给出，则表示空间不支持该类分享）

纯文本分享

```
//开发者分享的文本内容
QQApiTextObject *txtObj = [QQApiTextObject objectWithText:@"text"];
SendMessageToQQReq *req = [SendMessageToQQReq reqWithContent:txtObj];
//将内容分享到 qq
QQApiSendResultCode sent = [QQApiInterface sendReq:req];
```

纯图片分享

```
//开发者分享图片数据
NSData *imgData = [NSData dataWithContentsOfFile:path];
QQApiImageObject *imgObj = [QQApiImageObject objectWithData:imgData
                                                    previewImageData:imgData
                                                    title:@"title"
                                                    description :@"description"];
SendMessageToQQReq *req = [SendMessageToQQReq reqWithContent:imgObj];
//将内容分享到 qq
QQApiSendResultCode sent = [QQApiInterface sendReq:req];
```

多图分享至 QQ 收藏

```
//开发者分享多个图片数据至 QQ 收藏
NSArray *imgArray = [NSArray arrayWithObjects: imgData, imgData1,
                                                imgData2, imgData3, nil];
QQApiImageObject *imgObj = [QQApiImageObject objectWithData:imgData
                                                    previewImageData:imgData
                                                    title:@"title"
                                                    description :@"description"
                                                   imageDataArray:imgArray];
[imgObj setCflag:kQQAPICtrlFlagQQShareFavorites];
SendMessageToQQReq *req = [SendMessageToQQReq reqWithContent:imgObj];
//将内容分享到 qq
QQApiSendResultCode sent = [QQApiInterface sendReq:req];
```

新闻分享

```
//分享跳转 URL
NSString *url = @"http://xxx.xxx.xxx/";
//分享图预览图 URL 地址
NSString *previewImageUrl = @"preImageUrl.png";
QQApiNewsObject *newsObj = [QQApiNewsObject
                            objectWithURL :[NSURL URLWithString:utf8String]
```

```
                        title: @"title";
                        description :@"description";
                        previewImageURL:[NSURL URLWithString:previewImageUrl]];
    SendMessageToQQReq *req = [SendMessageToQQReq reqWithContent:newsObj];
    //将内容分享到 qq
    //QQApiSendResultCode sent = [QQApiInterface sendReq:req];
    //将内容分享到 qzone
    QQApiSendResultCode sent = [QQApiInterface SendReqToQZone:req];
```

音乐分享

```
    //分享跳转 URL
    NSString *url = @"http://xxx.xxx.xxx/";
    //分享图预览图 URL 地址
    NSString *previewImageUrl = @"preImageUrl.png";
    //音乐播放的网络流媒体地址
    NSString *flashURL = @"xxx.mp3 ";
    QQApiAudioObject *audioObj =[QQApiAudioObject
                        objectWithURL :[NSURL URLWithString:url]
                        title:@"title"
                        description:@"description"
                        previewImageURL:[NSURL URLWithString:previewImageUrl]];
    //设置播放流媒体地址
    [audioObj setFlashUrl:flashURL];
    SendMessageToQQReq *req = [SendMessageToQQReq reqWithContent:audioObj]
    //将内容分享到 qq
    //QQApiSendResultCode sent = [QQApiInterface sendReq:req];
    //将被容分享到 qzone
    QQApiSendResultCode sent = [QQApiInterface SendReqToQZone:req];
```

群部落发表话题（2.8.1）

```
    //初始化一些图片数据
     ... ...
    QQApiGroupTribeImageObject *imgObj = [QQApiGroupTribeImageObject
                                          objectWithData:preImageData
                                          previewImageData:preImageData
                                          title:self.binding_title ? : @""
                                          description:self.binding_description ? : @""
                                          imageDataArray:imageArray];
    if ([self.binding_bid integerValue] > 0)
        imgObj.bid = self.binding_bid;
    if ([self.binding_groupTribeName length] > 0)
        imgObj.bname = self.binding_groupTribeName;
```

```
SendMessageToQQReq *req = [SendMessageToQQReq reqWithContent:imgObj]
//将内容分享到 qq
//QQApiSendResultCode sent = [QQApiInterface sendReq:req];
```

分享文件（仅数据线）(2.8.1)

```
NSString *filePath = [[[NSBundle mainBundle] resourcePath] stringByAppendingPathComponent:@"test.txt"];
NSData *fileData = [NSData dataWithContentsOfFile:filePath];
QQApiFileObject *fileObj = [QQApiFileObject
                            objectWithData:fileData
                            previewImageData:nil
                            title:self.binding_title ? : @""
                            description:self.binding_description ? : @""];
if (self.binding_description != nil && ![self.binding_description isEqualToString:@""])
    fileObj.fileName = self.binding_description;
else
    fileObj.fileName = @"test.txt";
[fileObj setCflag:kQQAPICtrlFlagQQShareDataline];
SendMessageToQQReq *req = [SendMessageToQQReq reqWithContent:fileObj];
//将内容分享到 qq
//QQApiSendResultCode sent = [QQApiInterface sendReq:req];
```

添加好友（2.8.1）

```
QQApiAddFriendObject *object = [[QQApiAddFriendObject alloc]
                                initWithOpenID:self.binding_openID];
object.description = self.binding_description;
object.subID = self.binding_subID;
object.remark = self.binding_remark;
SendMessageToQQReq* req = [SendMessageToQQReq reqWithContent:object];
QQApiSendResultCode sent = [QQApiInterface sendReq:req];
```

游戏绑定工会群（2.8.1）

```
QQApiGameConsortiumBindingGroupObject *object =
                    [[QQApiGameConsortiumBinding GroupObject alloc]
                    initWithGameConsortium:self.binding_GroupID
                    unionid:self.binding_GameSectionID
                    zoneID:self.binding_ownerSignature
                    appDisplayName:@"天天酷跑"];
SendMessageToQQReq* req = [SendMessageToQQReq reqWithContent:object];
QQApiSendResultCode sent = [QQApiInterface sendReq:req];
```

3. 调用 OpenAPI

iOS SDK 中具体支持的 API 种类和每条 API 的参数说明，请参照 API 列表。这里用设置用户头像举例说明。

3.1 OpenAPI 参数字典封装

在封装各接口的参数字典时，推荐使用为每个接口新增的参数封装辅助类，如：

接口(BOOL)addShareWithParams:(NSMutableDictionary *)params

对应辅助类 TCAddShareDic

TCAddShareDic 辅助类中属性：

@property (nonatomic, retain) TCRequiredStr paramTitle;

对应于 CGI 请求中参数"title"

TCRequiredStr 表示这是一个必填参数，类型是字符串

TCOptionalStr 表示这是一个可选参数，类型是字符串

3.2 设置用户头像调用示例

设置 QQ 头像时，调用 TencetnOAuth 对象的 setUserHeadpic 方法：

```
TCSetUserHeadpic *params = [TCSetUserHeadpic dictionary];
params.paramImage = image;
params.paramFileName = @"make";
UIViewController *headController = nil;
[_tencentOAuth setUserHeadpic:params andViewController:&headController];
UIViewController *rootController = [[[app delegate] window] rootViewController];
[rootController dismissModalViewControllerAnimated:NO];
[rootController presentModalViewController:headController animated:YES];
```

设置头像完成后，会调用 TencentSessionDelegate 中的 tencentOAuth:doCloseViewController 通知应用界面需要关闭:

```
- (void)tencentOAuth:(TencentOAuth *) tencentOAuth doCloseViewController: (UIViewController *) viewController {
    if (tencentOAuth == _tencentOAuth)  {
    UIApplication *app = [UIApplication sharedApplication];
    UIViewController *rootController = [[[app delegate] window] rootViewController];
    [rootController dismissModalViewControllerAnimated:YES];
    }
}
```

设置头像完成后，会调用 TencentSessionDelegate 中的 setUserHeadpicResponse 返回调用结果：

```
- (void)setUserHeadpicResponse:(APIResponse*) response {
    if (nil == response)
        return ;
    if (URLREQUEST_FAILED == response.retCode
        && kOpenSDKErrorUserHeadPicLarge == response.detailRetCode) {
        UIAlertView *alert = [[UIAlertView alloc] initWithTitle:@"操作失败" message:[NSString
stringWithFormat:@"您的图片大小超标啦，请更换一张试试呢:)"]
        delegate:self cancelButtonTitle:@"我知道啦" otherButtonTitles: nil];
        [alert show];
        [alert release];
    }
}
```

3.3 使用增量授权

当第三方应用调用某个 API 接口时，如果服务器返回操作未被授权，则会触发增量授权逻辑。第三方应用需自行实现 tencentNeedPerformIncrAuth:withPermissions:协议接口才能够进入增量授权逻辑 ,否则默认第三方应用放弃增量授权。示例如下：

```
- (BOOL)tencentNeedPerformIncrAuth:(TencentOAuth *)tencentOAuth withPermissions: (NSArray
*) permissions {
    // incrAuthWithPermissions 是增量授权时需要调用的登录接口
    // permissions 是需要增量授权的权限列表
    [tencentOAuth incrAuthWithPermissions:permissions];
    // 返回 NO 表明不需要再回传未授权 API 接口的原始请求结果，否则返回 YES
    return NO;
}
```

注意：在用户通过增量授权页重新授权登录后，第三方应用需更新自己维护的 token 及有效期限等信息。

用户在增量授权时是可以更换帐号进行登录的，强烈要求第三方应用核对增量授权后的用户 openid 是否一致，以添加必要的处理逻辑（用户帐号变更需重新拉取用户的资料等信息）

增量授权成功时，会通过 tencentDidUpdate:协议接口通知第三方应用：

```
- (void)tencentDidUpdate:(TencentOAuth *)tencentOAuth {
    _labelTitle.text = @"增量授权完成";
    if (tencentOAuth.accessToken
        && 0 != [tencentOAuth.accessToken length]) {
        // 在这里第三方应用需要更新自己维护的 token 及有效期限等信息
        // **务必在这里检查用户的 openid 是否有变更，变更需重新拉取用户的资料等信息**
        _labelAccessToken.text = tencentOAuth.accessToken;
    } else {
```

```
        _labelAccessToken.text = @"增量授权不成功，没有获取 accesstoken";
    }
}
```

增量授权失败时，会通过 tencentFailedUpdate:协议接口通知第三方应用：

```
- (void)tencentFailedUpdate:(UpdateFailType)reason {
    switch (reason) {
        case kUpdateFailNetwork:
            _labelTitle.text=@"增量授权失败，无网络连接，请设置网络";
            break;
        case kUpdateFailUserCancel:
            _labelTitle.text=@"增量授权失败，用户取消授权";
            break;
        case kUpdateFailUnknown:
        default:
            _labelTitle.text=@"增量授权失败，未知错误";
            break;
    }
}
```

3.4 返回数据说明

APIResponse 属性：

retCode - 网络请求返回码，主要表示服务器是否成功返回数据

seq - 请求的序列号，依次递增，方便内部管理

errorMsg - 错误消息

jsonResponse - 由服务器返回的 json 格式字符串转换而来的 json 字典数据（具体参数字段请参见对应 API 说明文档）

message - 服务器返回的原始字符串数据

detailRetCode - 新增的详细错误码，以区分不同的错误原因（v1.2 以及之前的 SDK 接口无此参数）

3.5 返回码说明

retCode 网络请求返回码说明：

返回码	含义
0	表示成功，请求成功发送到服务器，并且服务器返回的数据格式正确
1	表示失败，可能原因有网络异常，或服务器返回的数据格式错误，无法解析

detailRetCode 详细错误码说明：

返回码	含义

kOpenSDKInvalid	无效的错误码
[公共错误码]	
kOpenSDKErrorSuccess	成功
kOpenSDKErrorUnknown	未知错误
kOpenSDKErrorUserCancel	用户取消
kOpenSDKErrorReLogin	token 无效或用户未授权相应权限需要重新登录
kOpenSDKErrorOperationDeny	第三方应用没有该 api 操作的权限
[网络相关错误码]	
kOpenSDKErrorNetwork	网络错误，网络不通或连接不到服务器
kOpenSDKErrorURL	URL 格式或协议错误
kOpenSDKErrorDataParse	数据解析错误，服务器返回的数据解析出错
kOpenSDKErrorParam	传入参数错误
kOpenSDKErrorConnTimeout	http 连接超时
kOpenSDKErrorSecurity	安全问题
kOpenSDKErrorIO	下载和文件 IO 错误
kOpenSDKErrorServer	服务器端错误
[webview 中特有错误]	
kOpenSDKErrorWebPage	页面错误
[设置头像 自定义错误码段]	
kOpenSDKErrorUserHeadPicLarge	图片过大 设置头像自定义错误码

4. WPA 临时会话

iOS SDK 支持发起 QQ 临时会话，获取指定 QQ 帐号在线状态。使用 WPA 功能需要设置 QQ 业务回调，请参考 6. 处理 QQ 业务的回调。

4.1 发起 QQ 临时会话

下面是向指定 QQ 号码发起临时会话的示例代码：

```
- (void)onOpenWPA:(QElement *)sender {
    [self.view endEditing:YES];
    [self.root fetchValueUsingBindingsIntoObject:self];
    QQApiWPAObject *wpaObj = [QQApiWPAObject objectWithUin:self.binding_uin];
    SendMessageToQQReq *req = [SendMessageToQQReq reqWithContent:wpaObj];
    QQApiSendResultCode sent = [QQApiInterface sendReq:req];
    [self handleSendResult:sent];
}
```

4.2 获取指定 QQ 号码的在线状态

下面是获取指定 QQ 号码在线状态的示例代码：

```
- (void)getQQUinOnlineStatues:(QElement *)sender {
    [self.view endEditing:YES];
    [self.root fetchValueUsingBindingsIntoObject:self];
    NSArray *ARR = [NSArray arrayWithObjects:self.binding_uin, nil];
    [QQApiInterface getQQUinOnlineStatues:ARR delegate:self];
}
```

5. QZone

iOS SDK 支持获取 QQ 用户的信息、发表说说、设置及浏览获取空间相册。

5.1 获取用户信息

获取用户信息分为两个步骤：

首先，发送获取用户信息的请求；

```
[_tencentOAuth getUserInfo])
```

其中，_tencentOAuth 为 TencentOAuth 类的实例。

其次，实现协议 TencentSessionDelegate 的获取到用户信息后的回调方法：

```
//可以通过 response 获取数据的返回结果
- (void)getUserInfoResponse:(APIResponse*) response;
```

下面以显示所有信息为例，进行说明：

```
- (void)getUserInfoResponse:(APIResponse*) response {
    if (URLREQUEST_SUCCEED == response.retCode
        && kOpenSDKErrorSuccess == response.detailRetCode) {
        NSMutableString *str = [NSMutableString stringWithFormat:@""];
        for (id key in response.jsonResponse) {
            [str appendString: [NSString stringWithFormat:
                @"%@:%@\n", key, [response.jsonResponse objectForKey:key]]];
        }
        UIAlertView *alert = [[UIAlertView alloc] initWithTitle:@"操作成功"
                                                message: [NSString stringWithFormat:@"%@",str];
        delegate:self cancelButtonTitle:@"我知道啦" otherButtonTitles: nil];
        [alert show];
    } else {
        NSString *errMsg = [NSString stringWithFormat:@"errorMsg:%@\n%@",
            response.errorMsg, [response.jsonResponse objectForKey:@"msg"]];
        UIAlertView *alert = [[UIAlertView alloc] initWithTitle:@"操作失败"
                                                        message:errMsg
                                                        delegate:self
                                                        cancelButtonTitle:@"我知道啦"
                                                        otherButtonTitles: nil];
        [alert show];
    }
}
```

5.2 分享到 QZone

分享方式与第二章中的分享文本及图片等类似，举例如下：

```
// 设置预览图片
NSURL *previewURL = [NSURL URLWithString:@"..."];
// 设置分享链接
NSURL* url = [NSURL URLWithString: @"..."];
QQApiNewsObject* imgObj = [QQApiNewsObject objectWithURL:url
                                                   title: @"..."
                                                   description: @"..."
                                                   previewImageURL:previewURL];
// 设置分享到 QZone 的标志位
[imgObj setCflag: kQQAPICtrlFlagQZoneShareOnStart ];
SendMessageToQQReq* req = [SendMessageToQQReq reqWithContent:imgObj];
QQApiSendResultCode sent = [QQApiInterface sendReq:req];
```

获取分享结果的方法与 5.1 中相同，此时需要实现 TencentSessionDelegate 协议的方法如下，具体用法此处不再赘述使用方法。

```
// 分享到 QZone 回调
- (void)addShareResponse:(APIResponse*) response;
```

5.3 创建 QZone 相册

创建 QZone 相册的参数较多是通过 TCAddAlbumDic 传递的，举例如下：

```
- (void)addAlbum {
    TCAddAlbumDic *dic = [TCAddAlbumDic dictionary];
    // 设置相册名称
    dic.paramAlbumname = @"...";
    // 设置相册描述
    dic.paramAlbumdesc =@"...";
    NSInteger index = ...;
    if (index > 1) {
        //主要是参数是 1，3，4，5，缺少个 2
        index += 1;
    }
    // 设置相册权限
    dic.paramPriv = [NSString stringWithFormat:@"%d", index];
    if (5 == index) {       // 如果权限为回答问题的人可见
        // 设置问题
        dic.paramQuestion = @"...";
        // 设置问题答案
        dic.paramAnswer = @"...";
    }
    // 发送添加 QZone 相册请求
```

```
    if (NO == [_tencentOAuth addAlbumWithParams:dic]) {
        // 请求失败，显示相应提示
        // ...
    }
}
```

其中，_tencentOAuth 为 TencentOAuth 类的实例。

获取分享结果的方法与 5.1 中相同，此时需要实现 TencentSessionDelegate 协议的方法如下，具体用法此处不再赘述使用方法。

```
// 在 QZone 相册中创建一个新的相册回调
- (void)addAlbumResponse:(APIResponse*) response;
```

相册访问权限及对应的值

访问权限	对应值
所有人可见	1
全部 QQ 好友可见	3
仅主人可见	4
回答问题的人可见	5

5.4 获取相册列表

获取相册列表的请求只需直接获取即可[_tencentOAuth getListAlbum]；获取相册列表的数据，只需实现 TencentSessionDelegate 协议的响应方法即可，该协议方法如下：

```
// 获取用户 QZone 相册列表回调
- (void)getListAlbumResponse:(APIResponse*) response;
```

取得相册列表的具体方法如下：

```
- (void)getListAlbumResponse:(APIResponse*) response {
    if (URLREQUEST_SUCCEED == response.retCode
        && kOpenSDKErrorSuccess == response.detailRetCode) {
        // 取得相册数据
        NSArray *blumArray = [[response jsonResponse] objectForKey:@"album"];
    } else {
        NSString *errMsg = [NSString stringWithFormat: @"errorMsg: %@\n%@",
            response.errorMsg, [response.jsonResponse objectForKey:@"msg"]];
        UIAlertView *alert = [[UIAlertView alloc] initWithTitle:@"操作失败"
                                                        message:errMsg
                                                        delegate:self
                                                        cancelButtonTitle:@"我知道啦"
                                                        otherButtonTitles: nil];
        [alert show];
    }
}
```

5.5 上传图片

上传图片可以通过设置 TCUploadPicDic 并发送相应请求完成。举例如下：

```
- (void)uploadPic {
    TCUploadPicDic *params = [TCUploadPicDic dictionary];
    // 设置上传的图片，UIImage
    params.paramPicture = image;
    // 上传图片所在的相册 ID，可以通过获得相册列表获取
    params.paramAlbumid = [self albumId];
    // 图片名称
    params.paramTitle = @"...";
    // 图片描述
    params.paramPhotodesc = @"...";
    // 如果传 1，则当 albumid 为空时，图片会上传到手机相册
    // 如果传 0 或不传时，则当 albumid 为空时，图片会上传到贴图相册
    params.paramMobile = @"1";
    // 标识图片上传时是否发 feed，1 为发，0 为不发
    params.paramNeedfeed = @"1";
    // 照片拍摄时的地理位置的经度
    params.paramX = @"39.909407";
    // 照片拍摄时的地理位置的纬度
    params.paramY = @"116.397521";
    // 发送请求
    if (NO == [_tencentOAuth uploadPicWithParams:params]) {
        // 请求失败提示
        // ...
    }
}
```

其中，_tencentOAuth 为 TencentOAuth 类的实例。

获取分享结果的方法与 5.1 中相同，此时需要实现 TencentSessionDelegate 协议的方法如下，具体用法此处不再赘述使用方法。

```
// 上传照片到 QZone 指定相册回调
- (void)uploadPicResponse:(APIResponse*) response;
```

5.6 发表说说

发表说说可以通过设置 TCAddTopicDic 并发送相应请求完成。举例如下：

```
-(void)addTopic {
    TCAddTopicDic *params = [TCAddTopicDic dictionary];
    // 发布说说时引用的信息类型，1 表示图片，2 表示网页，3 表示视频
    params.paramRichtype = @"3";
    // 其参数设置详见本小节末尾处的列表
    params.paramRichval = @"http://www.tudou.com/programs/view/C0FuB0FTv50/";
```

```
    // 发布说说内容
    params.paramCon = @"腾讯 addtopic 接口测试--失控小警察视频参数";
    // 发布说说时的地址
    params.paramLbs_nm = @"广东省深圳市南山区高新科技园腾讯大厦";
    // 第三方平台类型，1 表示 QQ 空间，2 表示腾讯朋友，3 表示腾讯微博，4 表示腾讯 Q+
    params.paramThirdSource = @"2";
    // 发布说说时的经度
    params.paramLbs_x = @"39.909407";
    // 发布说说时的纬度
    params.paramLbs_y = @"116.397521";
    // 此部分为用户自定义的保留字段
    [params setObject:@"test" forKey:PARAM_USER_DATA];
    // 发布请求
    if(NO == [_tencentOAuth addTopicWithParams:params]) {
        // 请求失败提示
        // ...
    }
}
```

其中，_tencentOAuth 为 TencentOAuth 类的实例。

获取分享结果的方法与 5.1 中相同，此时需要实现 TencentSessionDelegate 协议的方法如下，具体用法此处不再赘述使用方法。

```
// 在 QZone 中发表一条说说回调
- (void)addTopicResponse:(APIResponse*) response;
```

paramRichval 参数设置：

当 richtype 为网页或视频时，此值为该网页或视频的 URL

当 richtype 为图片时，详情如下：

1. 当该图片来自于网站时

参数名称	是否必须	类型	描述
url	必须	string	网站图片的 URL
height	必须	string	图片高度，单位：px
width	必须	string	图片宽度，单位：px

输入时每个值中间用“&”分隔，如下所示：

“url=http://qq.com/logo.png&width=25&height=21”

2. 当该图片来自于 QQ 空间相册时

参数名称	是否必须	类型	描述
albumid	必须	string	图片所属空间相册的 ID
pictureid	必须	string	图片 ID
sloc	必须	string	小图 ID
pictype		string	图片类型（JPG=1，GIF=2，PNG=3）
picheight		string	图片高度，单位：px

picwidth	string	图片宽度，单位：px

输入时每个值中间用逗号分隔，如下所示 ：

“albumid,pictureid,sloc,pictype,picheight,picwidth”

5.7 验证空间粉丝

验证空间粉丝可以通过设置 TCCheckPageFansDic 并发送相应请求完成。举例如下：

```
-(void)checkFans {
    TCCheckPageFansDic *params = [TCCheckPageFansDic dictionary];
    // API 参数中的保留字段，可以塞入任意字典支持的类型，再调用完成后会带回给调用方
    params.paramUserData = @"checkFans";
    //表示认证空间的 QQ 号码
    [params setParamPage_id:@"973751369"];
    // 发送请求
    if(NO == [_tencentOAuth checkPageFansWithParams:params]){
        // 请求失败提示
        // ...
    }
}
```

其中，_tencentOAuth 为 TencentOAuth 类的实例。

获取分享结果的方法与 5.1 中相同，此时需要实现 TencentSessionDelegate 协议的方法如下，具体用法此处不再赘述使用方法。

```
// 检查是否是 QZone 某个用户的粉丝回调
- (void)checkPageFansResponse:(APIResponse*) response;
```

6.群功能

6.1 一键加群

```
QQApiJoinGroupObject *object = [QQApiJoinGroupObject
                                objectWithGroupInfo: groupID
                                key: groupKey];
SendMessageToQQReq* req = [SendMessageToQQReq reqWithContent:object];
QQApiSendResultCode sent = [QQApiInterface sendReq:req];
```

groupID：群号码

groupKey： 群 KEY ，可以到 http://qun.qq.com 获取

7. 处理 QQ 业务的回调

在使用 QQApiInterface 的方法时需要设置回调才能正确调用。设置方法如下：

```
- (BOOL)application:(UIApplication *)application handleOpenURL:(NSURL *)url {
    [QQApiInterface handleOpenURL:url delegate:qqApiDelegate];
    return YES;
}
```

在 handleOpenURL 中添加[QQApiInterface handleOpenURL:url delegate: qqApiDelegate]代码，可以在 QQAPIDemoEntry 类中实现 QQApiInterfaceDelegate 的回调方法。更完整的示例请参考 SDKDemo。

添加关于 onReq 和 onResp 的说明，同时微信与 QQ 回调相同，提醒开发者注意采用不同的代理处理回调。