

iOS SDK V3.1.0

函数列表

QQ 互联项目组

2016/5/5

目录

目录.....	2
1. QQApiInterface 依赖的请求类.....	4
1.1 QQApiTextObject.....	4
1.2 QQApiURLObject.....	4
1.3 QQApiExtendObject.....	5
1.4 QQApiWebImageObject.....	6
1.5 QQApiAudioObject.....	7
1.6 QQApiVideoObject.....	7
1.7 QQApiNewsObject.....	8
1.8 QQApiPayObject.....	8
1.9 QQApiCommonContentObject.....	9
1.10 QQApiWPAObject.....	9
1.11 QQApiAddFriendObject.....	10
1.12 QQApiGameConsortiumBindingGroupObject.....	10
1.13 QQApiGroupChatObject.....	11
2. QQApiInterface 依赖帮助类.....	12
2.1 GetMessageFromQQReq.....	12
2.2 GetMessageFromQQResp.....	12
2.3 SendMessageToQQReq.....	12
2.4 SendMessageToQQResp.....	12
2.5 ShowMessageFromQQReq.....	13
2.6 ShowMessageFromQQResp.....	13
3. QQApiInterface 接口封装类.....	14
3.1 QQApiInterfaceDelegate 协议.....	14
3.2 QQApiInterface.....	14
4. Tencent 业务相关类.....	17
4.1 TCAPIRequestDelegate 协议.....	17
4.2 TencentApiReq.....	17
4.3 TencentApiResp.....	18
4.4 TencentBaseMessageObj.....	18
4.5 TencentTextMessageObjV1.....	18
4.6 TencentImageMessageObjV1.....	19

4.7 TencentAudioMessageObjV1.....	19
4.8 TencentVideoMessageV1.....	20
4.9 TencentImageAndVideoMessageObjV1.....	20
4.10 请求类.....	21
5. TencentApiInterface.....	22
5.1 TencentApiInterfaceDelegate.....	22
5.2 TencentApiInterface.....	22
6. TencentOAuth.....	24
6.1 TencentOAuth 授权登录相关方法.....	24
6.2 QZone 相册/粉丝/日志/说说/设置 QQ 头像.....	28
6.3 QQ 定向分享.....	29
6.4 发送应用邀请.....	30
6.5 发起 PK 或发送炫耀.....	30
6.6 赠送或请求礼物.....	31
6.7 取消指定 API 调用.....	32
6.8 CGI 类任务创建接口.....	32
6.9 TencentOpenApi 发送任务统一接口.....	33
6.10 获取用户 OpenID.....	33
6.11 TencentLoginDelegate.....	33
6.12 TencentSessionDelegate.....	34
6.13 TencentWebViewDelegate.....	38
附录一 iOS SDK V2.9.5 函数列表汇总.....	39

1. QQApiInterface 依赖的请求类

1.1 QQApiTextObject

用于分享文本类型的对象。

初始化方法

```
-(id)initWithText:(NSString*)text;
```

工厂方法

```
+(id)objectWithText:(NSString*)text;
```

参数	含义
text	文本内容

1.2 QQApiURLObject

用于分享 URL 地址所指向的目标类型的对象。

初始化方法

```
-(id)initWithURL:(NSURL*)url  
    title:(NSString*)title  
    description:(NSString*)description  
    previewImageData:(NSData*)data  
    targetContentType:(QQApiURLTargetType)targetContentType;
```

```
-(id)initWithURL:(NSURL*)url  
    title:(NSString*)title  
    description:(NSString*)description  
    previewImageURL:(NSURL*)previewURL  
    targetContentType:(QQApiURLTargetType)targetContentType;
```

工厂方法

```
+(id)objectWithURL:(NSURL*)url  
    title:(NSString*)title
```

```

        description:(NSString*)description
        previewImageData:(NSData*)data
        targetContentType:(QQApiURLTargetType)targetContentType;

+ (id)objectWithURL:(NSURL*)url
        title:(NSString*)title
        description:(NSString*)description
        previewImageURL:(NSURL*)previewURL
        targetContentType:(QQApiURLTargetType)targetContentType;

```

参数	含义
url	URL 地址
title	标题
description	简要描述
previewImageURL	预览图片 URL
previewImageData	预览图片数据
targetContentType	URL 的数据类型

其中，targetContentType 的类型为枚举类型 QQApiURLTargetType，有如下四种值：

QQApiURLTargetType	含义
QQApiURLTargetTypeNotSpecified	未指定类型
QQApiURLTargetTypeAudio	音频类型
QQApiURLTargetTypeVideo	视频类型
QQApiURLTargetTypeNews	新闻类型

1.3 QQApiExtendObject

用于分享扩展数据类型的对象。

初始化方法

```

- (id)initWithData:(NSData*)data
    previewImageData:(NSData*)previewImageData
        title:(NSString*)title
    description:(NSString*)description;

```

```

- (id)initWithData:(NSData *)data
previewImageData:(NSData*)previewImageData
    title:(NSString *)title
    description:(NSString *)description
    imageDataArray:(NSArray *)imageDataArray;

```

工厂方法

```

+ (id)objectWithData:(NSData*)data
    previewImageData:(NSData*)previewImageData
        title:(NSString*)title
    description:(NSString*)description;

+ (id)objectWithData:(NSData*)data
    previewImageData:(NSData*)previewImageData
        title:(NSString*)title
    description:(NSString*)description
    imageDataArray:(NSArray*)imageDataArray;

```

参数	含义
data	数据内容
title	标题
description	简要描述
previewImageURL	预览图片 URL
previewImageData	预览图片数据
imageDataArray	发送多张图片的队列

1.4 QQApiWebImageObject

用于分享网络图片内容的对象，需指定一个网络图片的 URL。

初始化方法

```

- (id)initWithPreviewImageURL :(NSURL*)previewImageURL
    title:(NSString*)title
    description:(NSString*)description;

```

工厂方法

```
+ (id)objectWithPreviewImageURL : (NSURL*)previewImageURL  
                                title:(NSString*)title  
                                description : (NSString*)description;
```

各个参数含义同上，此处不再赘述。

1.5 QQApiAudioObject

用于分享目标内容为音频的 URL 的对象。

初始化方法

```
+ (id)objectWithURL : (NSURL*)url  
                    title:(NSString*)title  
                    description:(NSString*)description  
                    previewImageData:(NSData*)data;
```

工厂方法

```
+ (id)objectWithURL:(NSURL*)url  
                    title:(NSString*)title  
                    description:(NSString*)description  
                    previewImageURL:(NSURL*)previewURL;
```

各个参数含义同上，此处不再赘述。

1.6 QQApiVideoObject

用于分享目标内容为视频的 URL 的对象。

工厂方法

```
+ (id)objectWithURL : (NSURL*)url  
                    title:(NSString*)title  
                    description:(NSString*)description  
                    previewImageData:(NSData*)data;  
  
+ (id)objectWithURL : (NSURL*)url  
                    title:(NSString*)title  
                    description:(NSString*)description
```

```
previewImageURL:(NSURL*)previewURL;
```

各个参数含义同上，此处不再赘述。

1.7 QQApiNewsObject

用于分享目标内容为新闻的 URL 的对象。

工厂方法

```
+(id)objectWithURL:(NSURL*)url
           title:(NSString*)title
    description:(NSString*)description
previewImageData:(NSData*)data;

+(id)objectWithURL:(NSURL*)url
           title:(NSString*)title
    description:(NSString*)description
previewImageURL:(NSURL*)previewURL;
```

各个参数含义同上，此处不再赘述。

1.8 QQApiPayObject

用于 QQ 支付的对象。

初始化方法

```
-(id)initWithOrderNo:(NSString*)OrderNo
      AppInfo:(NSString*)AppInfo;
```

工厂方法

```
+(id)objectWithOrderNo:(NSString*)OrderNo
      AppInfo:(NSString*)AppInfo;
```

参数	含义
OrderNo	支付订单号
AppInfo	支付来源信息

1.9 QQApiCommonContentObject

通用模板类型对象，用于分享一个固定显示模板的图文混排对象。

图片列表和文本列表不能同时为空。

工厂方法

```
+(id)objectWithLayoutType:(int)layoutType
    textArray:(NSArray*)textArray
    pictureArray:(NSArray*)pictureArray
    previewImageData:(NSData*)data;
```

参数	含义
layoutType	预定义的界面布局类型
textArray	文本列表
pictureArray	图片列表
previewImageData	预览图片数据

```
+(id)objectWithDictionary:(NSDictionary*)dic;
-(NSDictionary*)toDictionary;
```

以上两方法是将一个 NSDictionary 对象转化为 QQApiCommomContentObject，如果无法转换，则返回空。此方法也可以用于初始化。

1.10 QQApiWPAObject

用于发起 WPA（临时对话）的对象

初始化方法

```
-(id)initWithUin:(NSString*)uin;
```

工厂方法

```
+(id)objectWithUin:(NSString*)uin;
```

参数	含义
uin	想要对话的 QQ 号

1.11 QQApiAddFriendObject

用于添加 QQ 好友的对象

初始化方法

```
-(id)initWithOpenID:(NSString*)openID;
```

工厂方法

```
+(id)objectWithOpenID:(NSString*)openID;
```

参数	含义
openID	想要添加的对方的 OpenID

1.12 QQApiGameConsortiumBindingGroupObject

用于游戏公会绑定群的对象

初始化方法

```
-(id)initWithGameConsortium :(NSString*)signature  
unionid:(NSString*)unionid  
zoneID :(NSString*)zoneID  
appDisplayName:(NSString*)appDisplayName;
```

工厂方法

```
+(id)objectWithGameConsortium:(NSString*)signature  
unionid:(NSString*)unionid  
zoneID:(NSString*)zoneID  
appDisplayName :(NSString*)appDisplayName;
```

参数	含义
signature	游戏盟主身份验证签名， 从游戏后台获取
unionid	公会 ID（一般为数字）
zoneID	区域 ID（一般为数字）

appDisplayName

第三方 app 名称

1.13 QQApiGroupChatObject

用于发起群会话的对象

初始化方法

```
-(id)initWithGroup:(NSString*)groupID;
```

工厂方法

```
+(id)objectWithGroup:(NSString*)groupID;
```

参数

含义

groupID

想要对话的群号

2. QQApiInterface 依赖帮助类

2.1 GetMessageFromQQReq

请求帮助类，其工厂方法

```
+ (GetMessageFromQQReq *)req;
```

2.2 GetMessageFromQQResp

应答帮助类，其工厂方法

```
+ (GetMessageFromQQResp *)respWithContent:(QQApiObject *)message;
```

参数	含义
message	具体获取消息的实例

2.3 SendMessageToQQReq

请求帮助类，其工厂方法

```
+ (SendMessageToQQReq *)reqWithContent:(QQApiObject *)message;
```

参数	含义
message	具体发送请求帮助消息的实例

2.4 SendMessageToQQResp

应答帮助类，其工厂方法

```
+ (SendMessageToQQResp *)respWithResult:(NSString *)result  
errorDescription:(NSString *)errDesp
```

```
extendInfo:(NSString*)extendInfo;
```

参数	含义
result	请求处理结果
errDesp	具体错误描述信息
extendInfo	扩展信息

2.5 ShowMessageFromQQReq

请求帮助类，其工厂方法

```
+ (ShowMessageFromQQReq *)reqWithContent:(QQApiObject *)message;
```

参数	含义
message	具体待展示消息的实例

2.6 ShowMessageFromQQResp

应答帮助类，其工厂方法

```
+ (ShowMessageFromQQResp *)respWithResult:(NSString *)result  
errorDescription:(NSString *)errDesp;
```

参数	含义
result	展现消息结果
errDesp	具体错误描述信息

3. QQApiInterface 接口封装类

3.1 QQApiInterfaceDelegate 协议

处理来自 QQ 的请求及响应的回调协议

```
// 处理来自 QQ 的请求
```

```
- (void)onReq:(QQBaseReq *)req;
```

```
// 处理来自 QQ 的响应
```

```
- (void)onResp:(QQBaseResp *)resp;
```

```
// 处理 QQ 在线状态的回调
```

```
- (void)isOnlineResponse:(NSDictionary *)response;
```

3.2 QQApiInterface

对 QQApi 的简单封装类。QQApi 为处理发送请求的执行类，非对外开放。

```
// 处理由手 Q 唤起的跳转请求
```

```
+ (BOOL)handleOpenURL:(NSURL *)url
```

```
delegate :(id<QQApiInterfaceDelegate>)delegate;
```

参数	含义
url	待处理的 url 跳转请求
delegate	第三方应用用于处理来自 QQ 请求及响应的委托对象
返回值	跳转请求处理结果，YES 表示成功处理，NO 表示不支持的请求协议或处理失败

该函数一般添加与工程中的 AppDelegate 类的处理 URL 函数（当收到外部的 URL 跳转时，app 会第一时间调用此函数）中，即

```
- (BOOL)application:(UIApplication *)application
    openURL:(NSURL *)url
    sourceApplication:(NSString *)sourceApplication
    annotation:(id)annotation
```

```
// 向手 Q 发起分享请求
```

```
+ (QQApiSendResultCode)sendReq:(QQBaseReq *)req;
```

参数	含义
req	分享内容的请求
返回值	请求发送结果码

返回值为 QQApiSendResultCode 枚举类型，其枚举值含义如下：

QQApiSendResultCode	含义
EQQAPISENDSUCCESS	发送成功
EQQAPIQQNOTINSTALLED	未安装手 Q
EQQAPIQQNOTSUPPORTAPI	不支持 API
EQQAPIMESSAGEINVALID	发送消息的 type 类型无效
EQQAPIMESSAGECONTENTNULL	发送消息内容为空
EQQAPIMESSAGECONTENTINVALID	发送消息内容无效
EQQAPIAPPNOTREGISTERED	App 未注册
EQQAPIAPPSHAREASYNC	异步分享
EQQAPIQQNOTSUPPORTAPI_WITH_ERRORSHOW	不支持数据线
EQQAPISENDFAILD	发送失败
EQQAPIQZONENOTSUPPORTTEXT	qzone 分享不支持 text 类型分享
EQQAPIQZONENOTSUPPORTIMAGE	qzone 分享不支持 image 类型分享

```
// 向手 Q QZone 结合版发起分享请求
```

```
+ (QQApiSendResultCode)SendReqToQZone:(QQBaseReq *)req;
```

```
// 向手 Q 群部落发起分享请求
```

```
+ (QQApiSendResultCode)SendReqToQQGroupTribe:(QQBaseReq *)req;
```

```
// 向手 Q 发送应答消息
```

```
+ (QQApiSendResultCode)sendResp:(QQBaseResp *)resp;
```

各个参数含义同上，此处不再赘述。

```
// 检测是否已安装 QQ, 如果 QQ 已安装则返回 YES，否则返回 NO
```

```
+ (BOOL)isQQInstalled;
```

```
// 检测 QQ 是否支持 API 调用, 如果当前安装 QQ 版本支持 API 调用则返回 YES，否则  
返回 NO
```

```
+ (BOOL)isQQSupportApi;
```

```
// 启动 QQ, 成功返回 YES，否则返回 NO
```

```
+ (BOOL)openQQ;
```

```
// 获取 QQ 下载地址, 如果 App 通过[QQApiInterface isQQInstalled]和[QQApiInterface  
isQQSupportApi]检测发现 QQ 没安装或当前版本 QQ 不支持 API 调用，可引导用户通过打开  
此链接下载最新版 QQ。
```

```
+ (NSString *)getQQInstallUrl;
```

```
// 批量检测 QQ 号码是否在线
```

```
+ (void)getQQUinOnlineStatues :(NSArray *)QQUins
```

```
delegate:(id<QQApiInterfaceDelegate>)delegate;
```

参数	含义
QQUins	待检测的 QQ 号码的队列
delegate	接收返回结果的代理对象

此 delegate 需实现- (void)isOnlineResponse:(NSDictionary *)response;方法。

4. Tencent 业务相关类

4.1 TCAPIRequestDelegate 协议

该协议用于实现 TCAPIRequest 请求的回调

```
- (void)cgiRequest:(TCAPIRequest *)request
    didResponse:(APIResponse *)response;
```

参数	含义
request	发送的请求
response	服务器返回的结果

补充 TCAPIRequest 类与 APIResponse 类的简介

类名	用途
TCAPIRequest	CGI 请求的参数字典封装辅助基类，将相应属性的值以 key-value 的形式保存到参数字典中
APIResponse	封装服务器返回的结果，包括错误码、错误信息、原始返回数据以及返回数据的 json 格式字典

4.2 TencentApiReq

请求包类，用于向其他业务发送请求包，其工厂方法

```
+ (TencentApiReq *)reqFromSeq:(NSInteger)apiSeq
    type:(TencentReqMessageType)type;
```

参数	含义
apiSeq	请求序列号

type	请求类型
------	------

其中，type 为 TencentReqMessageType 枚举类型，其枚举值含义如下：

TencentReqMessageType	含义
ReqFromTencentAppQueryContent	TX APP 请求内容填充 (需要第三方开发者填充内容后需要主动调用 sendRespMessageToTencentApp)
ReqFromTencentAppShowContent	TX APP 请求展现内容 (不用调用答复)
ReqFromThirdAppQueryContent	第三方 APP 请求内容
ReqFromThirdAppShowContent	第三方 APP 请求展现内容 (类似分享)

4.3 TencentApiResp

答复包类，用于向其他业务发送答复包，其工厂方法

```
+ (TencentApiResp *)respFromReq:(TencentApiReq *)req;
```

参数	含义
req	答复对应的请求包

4.4 TencentBaseMessageObj

应用之间传递消息体消息体的根类

```
// 判断消息是否有效，如果有效返回 YES，否则返回 NO
```

```
- (BOOL)isVaild;
```

4.5 TencentTextMessageObjV1

应用之间传递的文本消息体，其初始化方法

```
- (id)initWithText:(NSString *)text;
```

参数	含义
text	文本，长度不能超过 4096 个字

4.6 TencentImageMessageObjV1

应用之间传递的图片消息，其初始化方法

```
- (id)initWithImageData:(NSData *)dataImage;
- (id)initWithImageUrl:(NSString *)url;
- (id)initWithType:(TencentApiImageSourceType)type;
```

参数	含义
dataImage	图片数据，不能大于 10M
url	图片 url
type	图片类型

其中，type 为 TencentApiImageSourceType 枚举类型，其枚举值含义如下：

TencentApiImageSourceType	含义
AllImage	图片数据是 url 或二进制数据
UrlImage	图片数据是 url
DataImage	图片数据是二进制数据

4.7 TencentAudioMessageObjV1

应用之间传递音频的消息，其初始化方法

```
- (id)initWithAudioUrl:(NSString *)url;
```

参数	含义
url	音频 URL

4.8 TencentVideoMessageV1

应用之间传递视频的消息，其初始化方法

```
- (id)initWithVideoUrl:(NSString *)url
    type:(TencentApiVideoSourceType)type;

- (id)initWithType:(TencentApiVideoSourceType)type;
```

参数	含义
url	视频 URL 其长度不能超过 1024
type	视频来源类型

其中，type 为 TencentApiVideoSourceType 枚举类型，其枚举值含义如下：

TencentApiImageSourceType	含义
AllVideo	视频来源于本地或网络
LocalVideo	视频来源于本地
NetVideo	视频来源于网络

4.9 TencentImageAndVideoMessageObjV1

应用之间传递视频图片的消息体，图片视频可以任选一个且只能选择一个内容填充。

// 初始化方法

```
- (id)initWithMessage:(NSData *)dataImage videoUrl:(NSString *)url;
```

// 设置图片方法

```
- (void)setDataImage:(NSData *)dataImage;
```

// 设置视频方法

```
- (void)setVideoUrl:(NSString *)videoUrl;
```

参数	含义
dataImage	图片数据
url/ videoUrl	视频 url

4.10 请求类

由于以下所有类均继承自 TCAPIRequest，且函数方法都大致相同，故不单独分小结。

以下各工厂方法结果均为发挥一个相应对象，来进行 API 参数的填充。返回的对象均为自动释放的。

```
+ (TCAddTopicDic *) dictionary;
```

```
+ (TCAddOneBlogDic *) dictionary;
```

```
+ (TCAddAlbumDic *) dictionary;
```

```
+ (TCUploadPicDic *) dictionary;
```

```
+ (TCAddShareDic *) dictionary;
```

```
+ (TCCheckPageFansDic *) dictionary;
```

```
+ (TCSetUserHeadpic *) dictionary;
```

```
+ (TCListPhotoDic *) dictionary;
```

```
+ (TCSendStoryDic *) dictionary;
```

以下类均可以直接填写相应参数后将对象当作参数传入 API 中，其作用如下：

类名	用途
TCAddTopicDic	发表说说
TCAddOneBlogDic	发表日志
TCAddAlbumDic	创建空间相册
TCUploadPicDic	上传一张照片到 QQ 空间相册
TCAddShareDic	同步分享到 QQ 空间
TCCheckPageFansDic	验证是否是认证空间粉丝
TCSetUserHeadpic	设置用户头像
TCListPhotoDic	获取用户 QQ 空间相册中的照片列表
TCSendStoryDic	QQ 空间定向分享

具体使用方法，请参照《IOS SDK API V2.9.3 使用说明》。

5. TencentApiInterface

5.1 TencentApiInterfaceDelegate

TencentApiInterface 的回调

```
// 请求获得内容, 当前版本只支持第三方相应腾讯业务请求
```

```
- (BOOL)onTencentReq:(TencentApiReq *)req;
```

```
// 响应请求答复, 当前版本只支持腾讯业务相应第三方的请求答复
```

```
- (BOOL)onTencentResp:(TencentApiResp *)resp;
```

5.2 TencentApiInterface

```
// 发送答复返回腾讯业务
```

```
+ (TencentApiRetCode)sendRespMessageToTencentApp:(TencentApiResp *)resp;
```

参数	含义
resp	答复内容
返回值	处理结果的返回码

其中, 返回码的类型为 TencentApiRetCode 枚举类型, 其枚举值含义如下:

TencentPlatformType	含义
kTencentApiSuccess	成功
kTencentApiPlatformUninstall	平台未安装
kTencentApiPlatformNotSupport	平台不支持
kTencentApiParamsError	平台错误
kTencentApiFail	失败

```
// 是否可以处理拉起协议, 可以返回 YES, 否则返回 NO
```

```
+ (BOOL)canOpenURL:(NSURL *)url delegate:(id<TencentApiInterfaceDelegate>)delegate;
```

```
// 处理应用拉起协议，处理成功返回 YES，否则返回 NO
```

```
+ (BOOL)handleOpenURL:(NSURL *)url delegate:(id<TencentApiInterfaceDelegate>)delegate;
```

```
// 用户设备是否安装腾讯 APP，安装返回 YES，否则返回 NO
```

```
+ (BOOL)isTencentAppInstall:(TencentPlatformType)platform;
```

```
// 用户设备是否支持调用 SDK，支持返回 YES，否则返回 NO
```

```
+ (BOOL)isTencentAppSupportTencentApi:(TencentPlatformType)platform;
```

参数	含义
url	待处理的 URL
delegate	处理回调的代理
platform	指定的腾讯业务

其中，platform 为 TencentPlatformType 枚举类型，其枚举值含义如下：

TencentPlatformType	含义
kIphoneQQ	手机 QQ
kIphoneQZONE	手机 QZone
kThirdApp	第三方 app

6. TencentOAuth

6.1 TencentOAuth 授权登录相关方法

获取 SDK 基本信息的相关函数

```
// 获取当前 SDK 的版本号
```

```
+ (NSString*)sdkVersion;
```

```
// 获取当前 SDK 的小版本号
```

```
+ (NSString*)sdkSubVersion;
```

```
// 获取当前 SDK 是否为精简版
```

```
+ (BOOL)isLiteSDK;
```

```
// 判断是否有登陆被发起，但是还没有过返回结果
```

```
+ (TencentAuthorizeState *)authorizeState;
```

```
// 获取当前手机 QQ 的版本号
```

```
+ (QQVersion)iphoneQQVersion;
```

其中，TencentAuthorizeState 为枚举类型，其枚举值含义如下：

TencentAuthorizeState	含义
kTencentNotAuthorizeState	无授权
kTencentSSOAuthorizeState	有人发起了 sso 授权还未返回
kTencentWebviewAuthorzieState	有人发起了 webview 授权还未返回

QQVersion 为枚举类型，其枚举值含义如下：

QQVersion	含义
kQQUninstall	未安装 QQ
kQQVersion3_0	QQ3.0
kQQVersion4_0	支持 sso 登录
kQQVersion4_2_1	iOS7 兼容

kQQVersion4_5	WPA 会话
kQQVersion4_6	sso 登陆信令通道切换
kQQVersion4_7	QQ4.7

判断 SSO 登录准备事项是否支持

```
// 判断用户手机上是否安装手机 QQ，安装返回 YES，否则返回 NO
```

```
+ (BOOL)iphoneQQInstalled;
```

```
// 判断用户手机上的手机 QQ 是否支持 SSO 登录，支持返回 YES，否则返回 NO
```

```
+ (BOOL)iphoneQQSupportSSOLogin;
```

```
// 判断用户手机上是否安装手机 QZone，安装返回 YES，否则返回 NO
```

```
+ (BOOL)iphoneQZoneInstalled;
```

```
// 判断用户手机上的手机 QZone 是否支持 SSO 登录，支持返回 YES，否则返回 NO
```

```
+ (BOOL)iphoneQZoneSupportSSOLogin;
```

初始化 TencentOAuth 对象方法

```
- (id)initWithApplId:(NSString *)applId  
andDelegate:(id<TencentSessionDelegate>)delegate;
```

参数	含义
applId	第三方应用在互联开放平台申请的唯一标识
delegate	第三方应用用于接收请求返回结果的委托对象

其中 delegate 遵循 TencentSessionDelegate 协议，6.2 小结有详细介绍。

登录授权

```
- (BOOL)authorize:(NSArray *)permissions;
```

```
- (BOOL)authorize:(NSArray *)permissions
```

```
inSafari:(BOOL)bInSafari;
```

```
- (BOOL)authorize:(NSArray *)permissions
```

```
localAppId:(NSString *)localAppId
inSafari:(BOOL)bInSafari;
```

参数	含义
permissions	授权信息列表
bInSafari	是否使用 safari 进行登录，iOS SDK 1.3 版本开始废除此参数
localAppId	应用 APPID

增量授权，因用户没有授予相应接口调用的权限，需要用户确认是否授权

```
- (BOOL)incrAuthWithPermissions:(NSArray *)permissions;
```

参数含义同上，此处不再赘述

重新授权，因 token 废除或失效导致接口调用失败，需用户重新授权

```
- (BOOL)reauthorizeWithPermissions:(NSArray *)permissions;
```

参数含义同上，此处不再赘述

```
// 处理应用拉起协议，处理成功返回 YES，否则返回 NO
```

```
+ (BOOL)HandleOpenURL:(NSURL *)url;
```

```
// SDK 是否可以处理应用拉起协议，可以处理返回 YES，否则返回 NO
```

```
+ (BOOL)CanHandleOpenURL:(NSURL *)url;
```

参数	含义
url	处理第三方应用被呼起后的逻辑

这两个函数常常一起调用，先判断是否可以处理，再进行处理。该函数一般添加与工程中的 appDelegate 类的处理 URL 函数（当收到外部的 URL 跳转时，app 会第一时间调用此函数）中，即

```
- (BOOL)application :(UIApplication *)application
    openURL:(NSURL *)url
    sourceApplication :(NSString *)sourceApplication
    annotation:(id)annotation
```

```
// 获取 TencentOAuth 调用的上一次错误信息
```

```
+ (NSString *)getLastErrorMsg;
```

```
// 以 Server Side Code 模式授权登录时，通过此接口获取返回的 accessToken 值(Access Token 凭证，用于后续访问各开放接口);
```

```
// 以 Client Side Token 模式授权登录时，忽略此接口。
```

```
- (NSString *)getServerSideCode;
```

```
// 退出登录(退出登录后，TencentOAuth 失效，需要重新初始化)
```

```
- (void)logout:(id<TencentSessionDelegate>)delegate;
```

参数	含义
delegate	第三方应用用于接收请求返回结果的委托对象

```
// 判断登录态是否有效，YES 表示有效，NO 表示无效，请用户重新登录授权
```

```
- (BOOL)isSessionValid;
```

```
// 获取用户个人信息，YES 表示 API 调用成功，NO 表示 API 调用失败，登录态失败，重新登录。
```

```
- (BOOL)getUserInfo;
```

```
// SDK 内置 webview 实现定向分享时，第三方应用可以根据应用是否在白名单里来开启该配置开关，默认为关闭；在白名单里的应用调用该接口后，即打开 sdk 内置 webview 的二级白名单开关（相对与 sdk 后台的白名单），那么在 sdk 后台白名单校验请求失败的情况下，会继续先尝试采用内置 webview 进行分享。
```

```
- (void)openSDKWebViewQQShareEnable;
```

6.2 QZone 相册/粉丝/日志/说说/设置 QQ 头像

将这些方法放到一起，是因为这类方法参数及返回值均相似，所以，整理到一起，方便对比。

这些方法的参数均为 NSDictionary 类型，其关键字均对应 TencentOAuthObject.h 中的不同的类。

返回值类型均为 BOOL，且含义相同，YES 表示 API 调用成功，NO 表示 API 调用失败，登录态失败，重新登录。

```
// 获取用户 QZone 相册列表
```

```
- (BOOL)getListAlbum;
```

```
// 获取用户 QZone 相册列表，params 对应 TencentOAuthObject.h 中的 TListPhotoDic；
```

```
// 在本节的以下内容中，上述说明将简写为：
```

```
//获取用户 QZone 相册列表， TListPhotoDic；
```

```
- (BOOL)getListPhotoWithParams:(NSMutableDictionary *)params;
```

```
// 分享到 QZone，TAddShareDic
```

```
- (BOOL)addShareWithParams:(NSMutableDictionary *)params;
```

```
// 上传照片到 QZone 指定相册，TUploadPicDic
```

```
- (BOOL)uploadPicWithParams:(NSMutableDictionary *)params;
```

```
// 在 QZone 相册中创建一个新的相册，TAddAlbumDic
```

```
- (BOOL)addAlbumWithParams:(NSMutableDictionary *)params;
```

```
// 检查是否是 QZone 某个用户的粉丝，TCheckPageFansDic
```

```
- (BOOL) checkPageFansWithParams:(NSMutableDictionary *)params;
```

```
// 在 QZone 中发表一篇日志，TAddOneBlogDic
```

```
- (BOOL) addOneBlogWithParams:(NSMutableDictionary *)params;
```

```
// 在 QZone 中发表一条说说，TAddTopicDic
```

```
- (BOOL) addTopicWithParams:(NSMutableDictionary *)params;
```

```
// 设置 QQ 头像 使用默认的效果处理设置头像的界面，TCSetUserHeadpic
```

```
- (BOOL) setUserHeadpic:(NSMutableDictionary *)params;
```

```
// 设置 QQ 头像 会返回设置头像由第三方自己处理界面的弹出方式 ,TCSetUserHeadpic
```

```
- (BOOL)setUserHeadpic:(NSMutableDictionary *)params  
andViewController:(UIViewController **)viewController;
```

```
// 获取 QQ 会员信息(仅包括是否为 QQ 会员,是否为年费 QQ 会员)
```

```
- (BOOL)getVipInfo;
```

```
// 获取 QQ 会员详细信息
```

```
- (BOOL)getVipRichInfo;
```

6.3 QQ 定向分享

```
// QZone 定向分享 ,可以@到具体好友 ,完成后将触发 responseDidReceived:forMessage:  
回调 , message : "SendStory"
```

```
- (BOOL)sendStory:(NSMutableDictionary *)params  
friendList:(NSArray *)fopenIdArray;
```

参数	含义
params	参数字典，字典的关键字参见 TencentOAuthObject.h 中的 TCSendStoryDic
fopenIdArray	第三方应用预传人好友列表，好友以 openid 标识
返回值	与 6.2 节中的方法含义相同

6.4 发送应用邀请

```
// 发送应用邀请，完成后将触发 responseDidReceived:forMessage:回调，message：  
"AppInvitation"
```

```
- (BOOL)sendAppInvitationWithDescription:(NSString *)desc  
                                imageURL:(NSString *)imageUrl  
                                source:(NSString *)source;
```

参数	含义
desc	应用的描述文字，不超过 35 字符，如果为 nil 或@ "" 则显示默认描述
imageUrl	应用的图标，如果为 nil 或者@ "" 则显示默认图标
source	透传参数，由开发者自定义该参数内容
返回值	与 6.2 节中的方法含义相同

6.5 发起 PK 或发送炫耀

```
// 发起 PK 或者发送炫耀，完成后将触发 responseDidReceived:forMessage:回调，  
message：“AppChallenge”
```

```
- (BOOL)sendChallenge:(NSString *)receiver  
                    type:(NSString *)type  
                    imageURL:(NSString *)imageUrl  
                    message:(NSString *)message  
                    source:(NSString *)source;
```

参数	含义
receiver	必须指定一位进行 PK 或者炫耀的好友，填写其 OpenID，填写多个 OpenID 将截取第一个

type	类型, "pk"或者 "brag"
imageUrl	炫耀/挑战场景图的 URL
message	炫耀/挑战中的内容描述, 不超过 50 个字符, 超过限制则自动截断
source	透传参数, 由开发者自定义该参数内容
返回值	与 6.2 节中的方法含义相同

6.6 赠送或请求礼物

// 赠送或者请求礼物, 完成后将触发 `responseDidReceived:forMessage:`回调, `message` : "AppGiftRequest"

```
- (BOOL)sendGiftRequest:(NSString *)receiver
    exclude:(NSString *)exclude
    specified:(NSString *)specified
    only:(BOOL)only
    type:(NSString *)type
    title:(NSString *)title
    message:(NSString *)message
    imageURL:(NSString *)imageUrl
    source:(NSString *)source;
```

参数	含义
receiver	赠送或者请求礼物的好友的 OpenID ,支持填写多个, OpenID 之用","分隔, 为 nil 时将由用户通过好友选择器选择好友
exclude	用户通过好友选择器选择好友场景下, 希望排除的好友 (不显示在好友选择器)
specified	用户通过好友选择器选择好友场景下, 希望出现的指定好友
only	是否只显示 specified 指定的好友
type	类型, "request"或者 "freegift"
title	免费礼物或请求名称, 不超过 6 个字符

message	礼物或请求的默认赠言，控制在 35 个汉字以内， 超过限制自动截断
imageUrl	请求或礼物配图的 URL，如果不传，则默认在弹 框中显示应用的 icon
source	透传参数，由开发者自定义该参数内容
返回值	与 6.2 节中的方法含义相同

6.7 取消指定 API 调用

```
// 退出指定 API 调用
```

```
- (BOOL)cancel:(id)userData;
```

参数	含义
userData	用户调用某条 API 的时候传入的保留参数
返回值	处理结果，YES 表示成功 NO 表示失败

6.8 CGI 类任务创建接口

```
// CGI 类任务创建接口
```

```
- (TCAPIRequest *)cgiRequestWithURL:(NSURL *)apiURL
                        method:(NSString *)method
                        params:(NSDictionary *)params
                        callback:(id<TCAPIRequestDelegate>)callback;
```

参数	含义
apiURL	CGI 请求的 URL 地址
method	CGI 请求方式："GET"，"POST"
params	CGI 请求参数字典
callback	CGI 请求结果的回调接口对象
返回值	CGI 请求任务实例，用于取消任务，返回 nil 代表任务创建失败

6.9 TencentOpenApi 发送任务统一接口

```
// TencentOpenApi 发送任务统一接口
```

```
- (BOOL)sendAPIRequest :(TCAPIRequest *)request  
        callback:(id<TCAPIRequestDelegate>)callback;
```

参数	含义
request	请求发送的任务
callback	任务发送后的回调地址
返回值	发送结果，YES 表示成功 NO 表示失败

6.10 获取用户 OpenID

```
// 获取用户 OpenID
```

```
- (NSString *)getUserOpenID;
```

参数	含义
返回值	用户的 OpenID

6.11 TencentLoginDelegate

TencentLoginDelegate 为授权登录回调协议，第三方应用实现登录的回调协议。

```
// 登录成功后的回调
```

```
- (void)tencentDidLogin;
```

```
// 登录失败后的回调
```

```
- (void)tencentDidNotLogin:(BOOL)cancelled;
```

参数	含义
----	----

cancelled

代表用户是否主动退出登录

// 登录时网络有问题的回调

- (void)tencentDidNotNetWork;

// 登录时权限信息的获得

```

- (NSArray *)getAuthorizedPermissions:(NSArray *)permissions
    withExtraParams:(NSDictionary *)extraParams;

```

参数	含义
permissions	登录时获取的授权信息列表
extraParams	登录时获取的授权信息列表及其具体参数
返回值	需要获取的授权信息 这个参数将被传至 QQ

6.12 TencentSessionDelegate

TencentSessionDelegate 为开放接口回调协议,第三方应用需要实现每条需要调用的 API 的回调协议。

// 退出登录的回调

- (void)tencentDidLogout;

// 因用户未授予相应权限而需要执行增量授权。在用户调用某个 api 接口时,如果服务器返回操作未被授权,则触发该回调协议接口,由第三方决定是否跳转到增量授权页面,让用户重新授权。

```

- (BOOL)tencentNeedPerformIncrAuth :(TencentOAuth *)tencentOAuth
    withPermissions:(NSArray *)permissions;

```

参数	含义
tencentOAuth	登录授权对象
permissions	需增量授权的权限列表
返回值	是否仍然回调返回原始的 api 请求结果

备注：不实现该协议接口则默认为不开启增量授权流程。若需要增量授权请调用

[TencentOAuth incrAuthWithPermissions:]

注意：增量授权时用户可能会修改登录的帐号

// [该逻辑未实现]因 token 失效而需要执行重新登录授权。在用户调用某个 api 接口时，如果服务器返回 token 失效，则触发该回调协议接口，由第三方决定是否跳转到登录授权页面，让用户重新授权。

- (BOOL)tencentNeedPerformReAuth:(TencentOAuth *)tencentOAuth;

参数含义同上，此处不做赘述。

备注：不实现该协议接口则默认为不开启重新登录授权流程。若需要重新登录授权请调用

[TencentOAuth reauthorizeWithPermissions:]

注意：重新登录授权时用户可能会修改登录的帐号

// 用户通过增量授权流程重新授权登录，token 及有效期限等信息已被更新。

- (void)tencentDidUpdate:(TencentOAuth *)tencentOAuth;

参数	含义
tencentOAuth	token 及有效期限等信息更新后的授权实例对象

// 用户增量授权过程中因取消或网络问题导致授权失败。

- (void)tencentFailedUpdate:(UpdateFailType)reason;

参数	含义
reason	授权失败原因

reason 为 UpdateFailType 枚举类型，其枚举值含义如下：

UpdateFailType	含义
kUpdateFailUnknown	未知原因
kUpdateFailUserCancel	用户取消
kUpdateFailNetwork	网络问题

以下是本协议中一系列回调函数，参数均为 APIResponse 类型，表示 API 返回结果，每

个回调对应这一个操作：

```
// 获取用户个人信息回调
```

```
- (void)getUserInfoResponse:(APIResponse*) response;
```

```
// 获取用户 QZone 相册列表回调
```

```
- (void)getListAlbumResponse:(APIResponse*) response;
```

```
// 获取用户 QZone 相片列表回调
```

```
- (void)getListPhotoResponse:(APIResponse*) response;
```

```
// 检查是否是 QZone 某个用户的粉丝回调
```

```
- (void)checkPageFansResponse:(APIResponse*) response;
```

```
// 分享到 QZone 回调
```

```
- (void)addShareResponse:(APIResponse*) response;
```

```
// 在 QZone 相册中创建一个新的相册回调
```

```
- (void)addAlbumResponse:(APIResponse*) response;
```

```
// 上传照片到 QZone 指定相册回调
```

```
- (void)uploadPicResponse:(APIResponse*) response;
```

```
// 在 QZone 中发表一篇日志回调
```

```
- (void)addOneBlogResponse:(APIResponse*) response;
```

```
// 在 QZone 中发表一条说说回调
```

```
- (void)addTopicResponse:(APIResponse*) response;
```

```
// 设置 QQ 头像回调
```

```
- (void)setUserHeadpicResponse:(APIResponse*) response;
```

```
// 获取 QQ 会员信息回调
```

```
- (void)getVipInfoResponse:(APIResponse*) response;
```

```
// 获取 QQ 会员详细信息回调
```

```
- (void)getVipRichInfoResponse:(APIResponse*) response;
```

```
// sendStory 分享的回调（已废弃，使用 responseDidReceived:forMessage:）
```

```
- (void)sendStoryResponse:(APIResponse*) response;
```

除了以上一对一的回调函数，还有一个回调函数集成了几个操作，如下：

```
// 社交 API 统一回调接口
```

```
- (void)responseDidReceived:(APIResponse*)response
    forMessage:(NSString *)message;
```

参数	含义
response	API 返回结果
message	响应的消息，目前支持 'SendStory'，'AppInvitation'， 'AppChallenge'， 'AppGiftRequest'

```
// post 请求的上传进度
```

```
- (void)tencentOAuth :(TencentOAuth *)tencentOAuth
    didSendBodyData :(NSInteger)bytesWritten
    totalBytesWritten :(NSInteger)totalBytesWritten
    totalBytesExpectedToWrite:(NSInteger)totalBytesExpectedToWrite
    userData :(id)userData;
```

参数	含义
tencentOAuth	返回回调的 tencentOAuth 对象
bytesWritten	本次回调上传的数据字节数
totalBytesWritten	总共已经上传的字节数
totalBytesExpectedToWrite	总共需要上传的字节数

userData

用户自定义数据

// 通知第三方界面需要被关闭

- (void)tencentOAuth:(TencentOAuth *)tencentOAuth

doCloseViewController:(UIViewController *)viewController;

参数

含义

tencentOAuth

返回回调的 tencentOAuth 对象

viewController

需要关闭的 viewController

6.13 TencentWebViewDelegate

TencentWebViewDelegate 为 H5 登录 webview 旋转方向回调协议，第三方应用可以根据自己 APP 的旋转方向限制，通过此协议设置

// webview 是否自动适应旋转方向

- (BOOL) tencentWebViewShouldAutorotateToInterfaceOrientation :

(UIInterfaceOrientation)toInterfaceOrientation;

参数

含义

toInterfaceOrientation

将要旋转到的方向

toInterfaceOrientation 为 UIInterfaceOrientation 枚举类型。

// webview 支持的显示方向，返回值类型为 UIInterfaceOrientationMask 枚举类型

- (NSUInteger) tencentWebViewSupportedInterfaceOrientationsWithWebkit;

// webview 是否自动适应旋转方向

- (BOOL) tencentWebViewShouldAutorotateWithWebkit;

附录一 iOS SDK V2.9.5 函数列表汇总

本表用于汇总 iOS SDK V2.9.3 中所有对外提供的函数，如需详细的解释，请参见正文相关章节。

函数	所属类
<code>-(id)initWithText:(NSString*)text;</code> <code>+(id)objectWithText:(NSString*)text;</code>	QQApiTextObject
<code>-(id)initWithURL:(NSURL*)url</code> <code>title:(NSString*)title</code> <code>description:(NSString*)description</code> <code>previewImageData:(NSData*)data</code> <code>targetContentType:(QQApiURLTargetType)targetContentType;</code>	QQApiURLObject
<code>-(id)initWithURL:(NSURL*)url</code> <code>title:(NSString*)title</code> <code>description:(NSString*)description</code> <code>previewImageURL:(NSURL*)previewURL</code> <code>targetContentType:(QQApiURLTargetType)targetContentType;</code>	
<code>+(id)objectWithURL:(NSURL*)url</code> <code>title:(NSString*)title</code> <code>description:(NSString*)description</code> <code>previewImageData:(NSData*)data</code> <code>targetContentType:(QQApiURLTargetType)targetContentType;</code>	
<code>+(id)objectWithURL:(NSURL*)url</code> <code>title:(NSString*)title</code> <code>description:(NSString*)description</code> <code>previewImageURL:(NSURL*)previewURL</code> <code>targetContentType:(QQApiURLTargetType)targetContentType;</code>	
<code>-(id)initWithData:(NSData*)data</code> <code>previewImageData:(NSData*)previewImageData</code> <code>title:(NSString*)title</code> <code>description:(NSString*)description;</code>	QQApiExtendObject

<pre> - (id)initWithData:(NSData *)data previewImageData:(NSData*)previewImageData title:(NSString *)title description:(NSString *)description imageDataArray:(NSArray *)imageDataArray; </pre>	
<pre> + (id)objectWithData:(NSData*)data previewImageData:(NSData*)previewImageData title:(NSString*)title description:(NSString*)description; </pre>	
<pre> + (id)objectWithData:(NSData*)data previewImageData:(NSData*)previewImageData title:(NSString*)title description:(NSString*)description imageDataArray:(NSArray*)imageDataArray; </pre>	
<pre> - (id)initWithPreviewImageURL :(NSURL*)previewImageURL title:(NSString*)title description:(NSString*)description; </pre>	QQApiWebImageObject
<pre> + (id)objectWithPreviewImageURL:(NSURL*)previewImageURL title:(NSString*)title description :(NSString*)description; </pre>	
<pre> +(id)objectWithURL :(NSURL*)url title:(NSString*)title description:(NSString*)description previewImageData:(NSData*)data; </pre>	QQApiAudioObject
<pre> +(id)objectWithURL:(NSURL*)url title:(NSString*)title description:(NSString*)description previewImageURL:(NSURL*)previewURL; </pre>	
<pre> +(id)objectWithURL :(NSURL*)url title:(NSString*)title description:(NSString*)description previewImageData:(NSData*)data; </pre>	QQApiVideoObject
<pre> +(id)objectWithURL :(NSURL*)url </pre>	

title:(NSString*)title description:(NSString*)description previewImageURL:(NSURL*)previewURL;	
+(id)objectWithURL:(NSURL*)url title:(NSString*)title description:(NSString*)description previewImageData:(NSData*)data;	QQApiNewsObject
+(id)objectWithURL:(NSURL*)url title:(NSString*)title description:(NSString*)description previewImageURL:(NSURL*)previewURL;	
-(id)initWithOrderNo:(NSString*)OrderNo AppInfo:(NSString*)AppInfo;	QQApiPayObject
+(id)objectWithOrderNo:(NSString*)OrderNo AppInfo:(NSString*)AppInfo;	
+(id)objectWithLayoutType:(int)layoutType textArray:(NSArray*)textArray pictureArray :(NSArray*)pictureArray previewImageData:(NSData*)data;	QQApiCommonContentObject
+(id)objectWithDictionary:(NSDictionary*)dic;	
-(NSDictionary*)toDictionary;	
-(id)initWithUin:(NSString*)uin;	QQApiWPAObject
+(id)objectWithUin:(NSString*)uin;	
-(id)initWithOpenID:(NSString*)openID;	QQApiAddFriendObject
+(id)objectWithOpenID:(NSString*)openID;	
-(id)initWithGameConsortium :(NSString*)signature unionid:(NSString*)unionid zoneID :(NSString*)zoneID appDisplayName:(NSString*)appDisplayName;	QQApiGameConsortiumBindingGroupObject
+(id)objectWithGameConsortium:(NSString*)signature unionid:(NSString*)unionid zoneID:(NSString*)zoneID appDisplayName :(NSString*)appDisplayName;	

<code>-(id)initWithGroup:(NSString*)groupId;</code>	QQApiGroupChatOb
<code>+(id)objectWithGroup:(NSString*)groupId;</code>	ject
<code>+ (GetMessageFromQQReq *)req;</code>	GetMessageFromQ QReq
<code>+ (GetMessageFromQQResp *)respWithContent:(QQApiObject *)message;</code>	GetMessageFromQ QResp
<code>+ (SendMessageToQQReq *)reqWithContent:(QQApiObject *)message;</code>	SendMessageToQQ Req
<code>+ (SendMessageToQQResp *)respWithResult:(NSString *)result errorDescription:(NSString *)errDesp extendInfo:(NSString*)extendInfo;</code>	SendMessageToQQ Resp
<code>+ (ShowMessageFromQQReq *)reqWithContent:(QQApiObject *)message;</code>	ShowMessageFrom QQReq
<code>+ (ShowMessageFromQQResp *) respWithResult: (NSString *)result errorDescription:(NSString *)errDesp;</code>	ShowMessageFrom QQResp
<code>- (void)onReq:(QQBaseReq *)req;</code>	QQApiInterfaceDele gate
<code>- (void)onResp:(QQBaseResp *)resp;</code>	
<code>- (void)isOnlineResponse:(NSDictionary *)response;</code>	
<code>+ (BOOL)handleOpenURL:(NSURL *)url delegate :(id<QQApiInterfaceDelegate>)delegate;</code>	QQApiInterface
<code>+ (QQApiSendResultCode)sendReq:(QQBaseReq *)req;</code>	
<code>+ (QQApiSendResultCode)sendReqToQZone:(QQBaseReq *)req;</code>	
<code>+ (QQApiSendResultCode)sendReqToQQGroupTribe: (QQBaseReq *)req;</code>	
<code>+ (QQApiSendResultCode)sendResp:(QQBaseResp *)resp;</code>	
<code>+ (BOOL)isQQInstalled;</code>	
<code>+ (BOOL) isQQSupportApi;</code>	
<code>+ (BOOL)openQQ;</code>	
<code>+ (NSString *)getQQInstallUrl;</code>	
<code>+ (void)getQQUinOnlineStatues :(NSArray *)QQUins delegate:(id<QQApiInterfaceDelegate>)delegate;</code>	

- (void)cgiRequest:(TCAPIRequest *)request didResponse:(APIResponse *)response;	TCAPIRequestDelegate
+ (TencentApiReq *)reqFromSeq:(NSInteger)apiSeq type:(TencentReqMessageType)type;	TencentApiReq
+ (TencentApiResp *)respFromReq:(TencentApiReq *)req;	TencentApiResp
- (BOOL)isVaild;	TencentBaseMessageObj
- (id)initWithText:(NSString *)text;	TencentTextMessageObjV1
- (id)initWithImageData:(NSData *)dataImage;	TencentImageMessageObjV1
- (id)initWithImageUrl:(NSString *)url;	
- (id)initWithType:(TencentApiImageSourceType)type;	
- (id)initWithAudioUrl:(NSString *)url;	TencentAudioMessageObjV1
- (id)initWithVideoUrl:(NSString *)url type:(TencentApiVideoSourceType)type;	TencentVideoMessageObjV1
- (id)initWithType:(TencentApiVideoSourceType)type;	
- (id)initWithMessage:(NSData *)dataImage videoUrl:(NSString *)url;	TencentImageAndVideoMessageObjV1
- (void)setDataImage:(NSData *)dataImage;	
- (void)setVideoUrl:(NSString *)videoUrl;	
+ (TCAAddTopicDic *) dictionary;	TCAAddTopicDic
+ (TCAAddOneBlogDic *) dictionary;	TCAAddOneBlogDic
+ (TCAAddAlbumDic *) dictionary;	TCAAddAlbumDic
+ (TCUploadPicDic *) dictionary;	TCUploadPicDic
+ (TCAAddShareDic *) dictionary;	TCAAddShareDic
+ (TCCheckPageFansDic *) dictionary;	TCCheckPage-FansDic
+ (TCSetUserHeadpic *) dictionary;	TCSetUserHeadpic
+ (TCListPhotoDic *) dictionary;	TCListPhotoDic
+ (TCSendStoryDic *) dictionary;	TCSendStoryDic
- (BOOL)onTencentReq:(TencentApiReq *)req;	TencentApiInterface

- (BOOL)onTencentResp:(TencentApiResp *)resp;	Delegate
+ (TencentApiRetCode)sendRespMessageToTencentApp: (TencentApiResp *)resp;	TencentApiInterface
+ (BOOL)canOpenURL:(NSURL *)url delegate:(id<TencentApiInterfaceDelegate>)delegate;	
+ (BOOL)handleOpenURL:(NSURL *)url delegate:(id<TencentApiInterfaceDelegate>)delegate;	
+ (BOOL)isTencentAppInstall:(TencentPlatformType)platform;	
+ (BOOL)isTencentAppSupportTencentApi: (TencentPlatformType)platform;	
+ (NSString*)sdkVersion;	TencentOAuth
+ (NSString*)sdkSubVersion;	
+ (BOOL)isLiteSDK;	
+ (TencentAuthorizeState *)authorizeState;	
+ (QQVersion)iphoneQQVersion;	
- (id)initWithAppId:(NSString *)appId andDelegate:(id<TencentSessionDelegate>)delegate;	
- (BOOL)authorize:(NSArray *)permissions;	
- (BOOL)authorize:(NSArray *)permissions inSafari:(BOOL)bInSafari;	
- (BOOL)authorize:(NSArray *)permissions localAppId:(NSString *)localAppId inSafari:(BOOL)bInSafari;	
- (BOOL)incrAuthWithPermissions:(NSArray *)permissions;	
- (BOOL)reauthorizeWithPermissions:(NSArray *)permissions;	
+ (BOOL)HandleOpenURL:(NSURL *)url;	
+ (BOOL)CanHandleOpenURL:(NSURL *)url;	
+ (NSString *)getLastErrorMsg;	
- (NSString *)getServerSideCode;	
- (void)logout:(id<TencentSessionDelegate>)delegate;	
- (BOOL)isSessionValid;	
- (BOOL)getUserInfo;	

- (void)openSDKWebViewQQShareEnable;	
- (BOOL)getListAlbum;	
- (BOOL)getListPhotoWithParams:(NSMutableDictionary *)params;	
- (BOOL)addShareWithParams:(NSMutableDictionary *)params;	
- (BOOL)uploadPicWithParams:(NSMutableDictionary *)params;	
- (BOOL)addAlbumWithParams:(NSMutableDictionary *)params;	
- (BOOL) checkPageFansWithParams:(NSMutableDictionary *)params;	
- (BOOL) addOneBlogWithParams:(NSMutableDictionary *)params;	
- (BOOL) addTopicWithParams:(NSMutableDictionary *)params;	
- (BOOL) setUserHeadpic:(NSMutableDictionary *)params;	
- (BOOL)setUserHeadpic:(NSMutableDictionary *)params andViewController:(UIViewController **)viewController;	
- (BOOL)getVipInfo;	
- (BOOL)getVipRichInfo;	
- (BOOL)sendStory:(NSMutableDictionary *)params friendList:(NSArray *)fopenIdArray;	
- (BOOL)sendAppInvitationWithDescription:(NSString *)desc imageURL:(NSString *)imageUrl source:(NSString *)source;	
- (BOOL)sendChallenge:(NSString *)receiver type:(NSString *)type imageURL:(NSString *)imageUrl message:(NSString *)message source:(NSString *)source;	
- (BOOL)sendGiftRequest:(NSString *)receiver exclude:(NSString *)exclude specified:(NSString *)specified	

only:(BOOL)only type:(NSString *)type title:(NSString *)title message:(NSString *)message imageURL:(NSString *)imageUrl source:(NSString *)source;	
- (BOOL)cancel:(id)userData;	
- (TCAPIRequest *)cgiRequestWithURL:(NSURL *)apiURL method:(NSString *)method params:(NSDictionary *)params callback:(id<TCAPIRequestDelegate>)callback;	
- (BOOL)sendAPIRequest :(TCAPIRequest *)request callback:(id<TCAPIRequestDelegate>)callback;	
- (NSString *)getUserOpenID;	
- (void)tencentDidLogin;	TencentLoginDele gate
- (void)tencentDidNotLogin:(BOOL)cancelled;	
- (void)tencentDidNotNetWork;	
- (NSArray *)getAuthorizedPermissions:(NSArray *)permissions withExtraParams:(NSDictionary *)extraParams;	
- (void)tencentDidLogout;	TencentSessionDele gate
- (BOOL)tencentNeedPerformIncrAuth :(TencentOAuth *) tencentOAuth withPermissions:(NSArray *)permissions;	
- (BOOL)tencentNeedPerformReAuth:(TencentOAuth *) tencentOAuth;	
- (void)tencentDidUpdate:(TencentOAuth *)tencentOAuth;	
- (void)tencentFailedUpdate:(UpdateFailType)reason;	
- (void)getUserInfoResponse:(APIResponse*) response;	
- (void)getListAlbumResponse:(APIResponse*) response;	
- (void)getListPhotoResponse:(APIResponse*) response;	
- (void)checkPageFansResponse:(APIResponse*) response;	
- (void)addShareResponse:(APIResponse*) response;	

- (void)addAlbumResponse:(APIResponse*) response;	
- (void)uploadPicResponse:(APIResponse*) response;	
- (void)addOneBlogResponse:(APIResponse*) response;	
- (void)setUserHeadpicResponse:(APIResponse*) response;	
- (void)getVipInfoResponse:(APIResponse*) response;	
- (void)getVipRichInfoResponse:(APIResponse*) response;	
- (void)sendStoryResponse:(APIResponse*) response;	
- (void)responseDidReceived:(APIResponse*)response forMessage:(NSString *)message;	
- (void)tencentOAuth :(TencentOAuth *)tencentOAuth didSendBodyData :(NSInteger)bytesWritten totalBytesWritten :(NSInteger)totalBytesWritten totalBytesExpectedToWrite:(NSInteger)totalBytesExpectedToWrite userData :(id)userData;	
- (void)tencentOAuth:(TencentOAuth *)tencentOAuth doCloseViewController:(UIViewController *)viewController;	
- (BOOL)tencentWebViewShouldAutorotateToInterfaceOrientation : (UIInterfaceOrientation)toInterfaceOrientation;	TencentWebViewDe legate
- (NSUInteger) tencentWebViewSupportedInterfaceOrientationsWithWebkit;	
- (BOOL) tencentWebViewShouldAutorotateWithWebkit;	