Andriod_SDK_V3.2.1 接口调用说明

1. 接口调用说明

接口主要有：

- 登录/校验登录态：login(Activity activity, String scope, IUiListener listener)
- 注销：logout(Context context)
- 应用邀请：invite(Activity activity, Bundle params, IUiListener listener) [需授权，暂不接受新申请]
- 应用分享：story(Activity activity, Bundle params, IUiListener listener)
- 设置 QQ 头像：setAvatar(Activity activity, Bundle params, IUiListener listener) [需授权，申请]
- 增量授权：reAuth(Activity activity, String scope, IUiListener listener)
- 发送请求：ask(Activity activity, Bundle params, IUiListener listener) [需授权，暂不接受新申请]
- 赠送礼物：gift(Activity activity, Bundle params, IUiListener listener) [需授权，暂不接受新申请]
- 发送挑战：challenge(Activity activity, Bundle params, IUiListener listener) [需授权，暂不接受新申请]
- 炫耀：brag(Activity activity, Bundle params, IUiListener listener [需授权，暂不接受新申请]
- 分享消息到 QQ（无需 QQ 登录）：shareToQQ(Activity activity, Bundle params, IUiListener listener)
- 分享到 QQ 空间（无需 QQ 登录）：shareToQzone(Activity activity, Bundle params, IUiListener listener)/publishToQzone(Activity activity, Bundle params, IUiListener listener)
- 应用评价：grade(Activity activity, Bundle params, IUiListener listener)
- 发表语音：voice(Activity activity, Bundle params, IUiListener listener) [需授权，暂不接受新申请]
- 好友召回：reactive(Activity activity, Bundle params, IUiListener listener) [需授权，暂不接受新申请]
- 查询附近的人：searchNearby(Activity activity, Bundle params, IUiListener listener) [需授权，申请]
- 删除用户位置信息：deleteLocation(Activity activity, Bundle params, IUiListener listener) [需授权，申请]

- 启动应用吧：startAppbar(Activity activity, String toPage)
- 打开应用吧标签页：startAppbarLabel(Activity activity, String label)
- 一键加群：joinQQGroup(Activity activity, String key)
- 一键绑群：bindQQGroup(Activity activity, Bundle params)
- 游戏内加好友：makeFriend(Activity activity, Bundle params)

除注销和查询附近的人，删除用户位置接口没有 UI 交互外，调用以上 SDK 提供的接口后，会弹出相应的界面，以完成后续的操作。

1.1 登录/校验登录态

通过调用 Tencent 类的 login 函数发起登录/校验登录态。

该接口具有两个作用，1. 如果开发者没有调用 mTencent 实例的 setOpenId、setAccessToken 接口，则该接口执行正常的登录操作；2. 如果开发者先调用 mTencent 实例的 setOpenId、setAccessToken 接口，则该接口执行校验登录态的操作。如果登录态有效，则返回成功给 app，如果登录态失效，则会自动进入登录流程，将最新的登录态数据返回给 app。

建议开发者在每次 app 启动时调用一次该接口(先调用 setOpenId、setAccessToken)，以确保每次打开 app 时用户都是有登录态的。

调用登录接口的示例代码如下：

```
private void doLogin() {
    IUiListener listener = new BaseUiListener() {
        @Override
        protected void doComplete(JSONObject values) {
            updateLoginButton();
        }
    };
    mTencent.login(this, SCOPE, listener);
}
```

调用登录接口的参数说明如下：

参数	参数说明
activity	调用者 activity。应用使用 SDK 时，会从应用自己的 Activity 跳转到 SDK 的 Activity，应用调用 SDK 的 Activity 即为这里的调用者 activity。
scope	应用需要获得哪些接口的权限，由“，”分隔。例如： SCOPE = “get_simple_userinfo,add_topic”；所有权限用“all”
listener	回调接口，IUiListener 实例。

注：在某些低端机上调用登录后，由于内存紧张导致 APP 被系统回收，登录成功后无法成功回传数据。

解决办法如下

在调用 login 的 Activity 或者 Fragment 重写 onActivityResult 方法，示例代码如下：

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    If (requestCode == Constants.REQUEST_LOGIN) {
        Tencent.onActivityResultData(requestCode,resultCode,data,loginListener);
    }
    super.onActivityResult(requestCode, resultCode, data);
}
```

1.2 注销

通过调用 Tencent 类的 logout 函数注销。

调用注销接口的示例代码如下：

```
mTencent.logout(this);
```

调用注销接口的参数说明如下：

参数	参数说明
context	调用者的 context。Context 是上下文的意思，每一个 Activity 都有对应的 Context。示例中的

	this 为调用者 Activity 对应的 Context。

1.3 邀请好友

特别声明：该接口目前仅支持白名单应用。出于信息安全的考虑，本接口目前升级中。升级完成前暂不接受开发者的授权申请。升级完成后，所有接入 QQ 登录的开发者将均可调用。 已接入过该接口的开发者，不需要更改调用的接口，将自动兼容。

邀请好友的基本流程为：

（1）判断用户是否已经登录，且应用是否获取了 openid；

（2）创建一个 Bundle，并填入必要参数；

（3）调用 SDK 中 Tencent 类的 invite 接口，启动邀请界面。

通过调用 Tencent 类的 invite 函数实现邀请功能。

调用邀请接口的示例代码如下：

```
private void onClickInvite() {
    if (mTencent.isSessionValid() && mTencent.getOpenId() != null) {
            Bundle params = new Bundle();
            params.putString(SocialConstants.PARAM_APP_ICON,
"http://imgcache.qq.com/qzone/space_item/pre/0/66768.gif");
            params.putString(SocialConstants.PARAM_APP_DESC,
                    "AndroidSdk_1_3: invite description!");
            params.putString(SocialConstants.PARAM_APP_CUSTOM,
                    "AndroidSdk_1_3: invite message!");
            params.putString(SocialConstants.PARAM_ACT, "进入应用");
            mTencent.invite(MainActivity.this, params, new BaseUiListener());
    }
}
```

调用邀请接口的参数说明如下：

参数	是否必传	类型	参数说明

SocialConstants.PARAM_APP_ICON	必传	String	邀请弹框中显示的应用图标的 URL。
SocialConstants.PARAM_APP_DESC	必传	String	邀请弹框中显示的邀请内容。
SocialConstants.PARAM_SOURCE	可选	String	由开发者自定义该参数内容，用于判断好友来源。 邀请成功后，被邀请方通过邀请链接进入应用时会携带该参数并透传给应用。

1.4 应用分享

应用分享的基本流程为：

（1）判断用户是否已经登录，且应用是否获取了 openid；

（2）创建一个 Bundle，并填入必要参数；

（3）调用 SDK 中 Tencent 类的 story 接口，启动分享界面。

通过调用 Tencent 类的 story 函数实现分享功能。

调用分享接口的示例代码如下：

```
private void onClickStory() {
    if (mTencent.isSessionValid() && mTencent.getOpenId() != null) {
        Bundle params = new Bundle();
        params.putString(SocialConstants.PARAM_TITLE,
                "AndroidSdk_1_3:UiStory title");
        params.putString(SocialConstants.PARAM_COMMENT,
                "AndroidSdk_1_3: UiStory comment");
        params.putString(SocialConstants.PARAM_IMAGE,
                "http://imgcache.qq.com/qzone/space_item/pre/0/66768.gif");
        params.putString(SocialConstants.PARAM_SUMMARY,
                "AndroidSdk_1_3: UiStory summary");
        params.putString(
                SocialConstants.PARAM_PLAY_URL,
                "http://player.youku.com/player.php/Type/Folder/"
```

```
                + "Fid/15442464/Ob/1/Pt/0/sid/XMzA0NDM2NTUy/v.swf");
        mTencent.story(MainActivity.this, params, new BaseUiListener());
    }
}
```

调用分享接口的参数说明如下：

参数	是否必传	类型	参数说明
SocialConstants.PARAM_TITLE	必传	String	分享的标题。（该参数暂时不起作用，title 固定为应用名称）
SocialConstants.PARAM_IMAGE	必传	String	分享的图片 URL。
SocialConstants.PARAM_COMMENT	可选	String	用户分享时的评论内容，可由用户输入。
SocialConstants.PARAM_SUMMARY	可选	String	分享的故事摘要。
SocialConstants.PARAM_SOURCE	可选	String	由开发者自定义该参数内容，用于判断好友来源。应用分享成功后，被邀请方通过邀请链接进入应用时会携带该参数并透传给应用。
SocialConstants.PARAM_ACT	可选	String	分享 feeds 中显示的操作区文字，参数值可为： '进入应用', '领取奖励', '获取能量', '帮助 TA'。
SocialConstants.PARAM_PLAY_URL	可选	String	分享内容中携带的视频链接。

1.5 设置 QQ 头像

特别声明：出于信息安全的考虑，本接口仅对可信赖的合作应用开放。已经成功接入“QQ 登录”的应用需提

交申请以获取访问本接口的权限。

设置 QQ 头像使用 Tencent 类中的 setAvatar 接口，setAvatar 使用了一个 Activity 来让用户调整图片，因此要在项目的 AndroidManifest.xml 中增加一个 Activity 配置，如下：

```
<application>
    <activity android:name="com.tencent.plus.ImageActivity" />
</application>
```

设置头像失败将在 SDK 自带的界面上提示用户重试，但可以指定设置成功后将跳转到哪个 Activity。若不指定，则返回上一个 Activity。

setAvatar 的接口格式如下：

- setAvatar(Activity activity, Bundle params, IUiListener listener)
- setAvatar(Activity activity, Bundle params, IUiListener listener, int enterAnim, int exitAnim)

注意：

1. V1.6 版本支持传入 IUiListener，用于获得设置头像成功或者出错的通知。通过传入的 IUiListener，指定设置成功后跳转的 Activity，跳转逻辑在 IUiListener#onComplete 的实现中完成；

2. V1.6 版还提供了切换动画的支持（enterAnim 和 exitAnim），其值为动画的资源文件（XML），可以查看 Demo 文件夹中的 /res/anim/zoomin.xml 和 zoomout.xml 获得详细示例。

setAvatar 的调用示例如下：

```
private void onClick() {
    if (mTencent.isSessionValid() && mTencent.getOpenId() != null) {
            Bundle params = new Bundle();
            params.putString(SocialConstants.PARAM_AVATAR_URI,
"xxx");

            //注意：SDK 1.4 不支持传入 listener， 该参数是 V1.6 新增的
            mTencent.setAvatar(this, params, new BaseUiListener());
            //指定切换动画的调用形式（V1.6 新增）
            mTencent.setAvatar(this,params,new BaseUiListener(),
                    R.anim.zoomin, R.anim.zoomout);
    }
```

```
}
```

基本流程是：

（1）判断用户是否存在登录态并且是否获取了 openid；

（2）创建一个 Bundle，并填入必要参数；

（3）调用 SDK 中 Tencent 类的 setAvatar 方法，启动设置头像界面。

下面的表格是设置 QQ 头像 API 的 params 参数说明：

参数	是否必传	类型	参数说明
SocialConstants.PARAM_AVATAR_URI	必传	String	将要设置为 QQ 头像的图片的本地路径，不支持网络图片，不支持动态图（gif），如果传入 gif 格式的图片，则只显示和使用第一帧的画面

1.6 增量授权

通过调用 Tencent 类的 reAuth 函数进行授权。

当应用调用 API 返回没有权限（返回码为 100030)时，可以调用增量授权函数让用户重新进行授权。调用时所使用的参数跟登录是一样的，只是在授权页面那里只会显示要增量授权的项，即所传的 scope 参数。

注意：这里的 scope 参数跟登录时所传的 scope 参数作用是不一样的，这里只需要传需要增量授权的项，即用户之前没有授权的项。一般来说，就是前面调用失败的那个 API 的 scope。

调用增量授权的示例代码如下：

```
//调用"add_topic" API 的 listener
IUiListener listener = new BaseUiListener() {
    @Override
    protected void doComplete(JSONObject response, Object state) {
        int ret=response.getInt("ret");
        if(ret==100030){
```

```
                //这里进行增量授权的操作
                if(mNeedReAuth){
                    Runnable r=new Runnable(){
                        public void run(){
                            mTencent.reAuth(MainActivity.this,"add_topic",new BaseUiListener());
                        }
                    };
                    MainActivity.this.runOnUiThread(r);
                }
            }
        }
};
```

调用增量授权接口的参数说明如下：

参数	参数说明
activity	调用者 activity。应用使用 SDK 时，会从应用自己的 Activity 跳转到 SDK 的 Activity，应用调用 SDK 的 Activity 即为这里的调用者 activity。
scope	应用需要用户增量授权权限，由“，”分隔。例如： SCOPE = “get_simple_userinfo,add_topic” ；
listener	回调接口，IUiListener 实例。

1.7 发送请求

特别声明：该接口目前仅支持白名单应用。出于信息安全的考虑，本接口目前升级中。升级完成前暂不接受开发者的授权申请。升级完成后，所有接入 QQ 登录的开发者将均可调用。 已接入过该接口的开发者，不需要更改调用的接口，将自动兼容。

通过调用 Tencent 类的 ask 函数发起请求。

调用发送请求接口的示例代码如下：

```
private void onClickAsk() {
    if (mTencent.isSessionValid() && mTencent.getOpenId() != null) {
        Bundle b = new Bundle();
        b.putString(SocialConstants.PARAM_RECEIVER, "15762FF138EE42D88FE2234D3B89C44B");
        b.putString(SocialConstants.PARAM_TITLE, "title 字段测试");
        b.putString(SocialConstants.PARAM_SEND_MSG, "msg 字段测试");
        b.putString(SocialConstants.PARAM_IMG_URL,
"http://i.gtimg.cn/qzonestyle/act/qzone_app_img/app888_888_75.png");;
        b.putString("exclude", "");
        b.putString("specified", "");
        b.putString("only", "0"); // 0 或者 1
        b.putString(SocialConstants.PARAM_SOURCE, URLEncoder.encode(""));
        mTencent.ask(this, b, new BaseUiListener());
    }
}
```

调用发送请求接口的参数说明如下：

参数	是否必传	类型	参数说明
SocialConstants.PARAM_RECEIVER	必传	String	若 receiver 包含 OpenID，则表示使用模式一（应用指定接收者模式）；若 receiver 为空，代表使用模式二（用户指定接收者模式） 1) 传入的 OpenID 必须为应用内关系链，即为已安装该应用的 QQ 好友 2) 传入个数不得超过 10 个，若超过 10 个则自动截取前 10 个，剩余舍弃 授权使用应用内关系链的应用才可使用模式一（应用指定接收者）
SocialConstants.PARAM_TITLE	必传	String	免费礼物或请求名称（最多 6 个汉字）
SocialConstants.PARAM_SEND_MSG	必传	String	✓ 礼物或请求的默认赠言，由应用传入，当应用未传入时使用系统默认赠言 ✓ 长度限制：35 个汉字以内，超过限制自动截断
SocialConstants.PARAM_IMG_URL	必传	String	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

SocialConstants.PARAM_EXCLUDE	可选	String	在用户自己指定好友的场景中，如果开发者不希望某些用户显示在好友选择器中，可传入这些用户的 openid，多个 openid 之间用“,”分隔（“exclude”和“specified”一共最多可传入 50 个 openid），好友选择器则不显示这些 openid 对应的用户。
SocialConstants.PARAM_SPECIFIED	可选	String	在用户自己指定好友的场景中，如果开发者需要在好友选择器中显示指定的用户，可传入这些用户的 openid，多个 openid 之间用“,”分隔（“exclude”和“specified”一共最多可传入 50 个 openid），好友选择器会显示这些 openid 对应的用户。
SocialConstants.PARAM_ONLY	可选	String	仅当设置了“specified”，需要传入该参数，用于标识是否在好友选择器中只显示“specified”指定的用户。默认值为“0”。 “0”：显示“specified”指定的用户，同时显示好友列表和已安装好友。 “1”：只显示“specified”指定的用户，无好友列表和已安装好友。此时必须保证“specified”有可用的 openid 传入，否则会报错。
SocialConstants.PARAM_SOURCE	可选	String	由开发者自定义该参数内容，用于判断用户接收的哪个好友的礼物或请求。 例如： 用户 A 向 C 赠送礼物/发送好友请求时，source 的值为“openid=001”，用户 B 向 C 赠送礼物/发送好友请求

			时，source 的值为“openid=002”。当用户 C 点击链接进入免费礼物或好友请求的查看页面时，url 中会携带 app_custom 参数，app_custom 的值为“openid=001”，说明 C 响应的是 A 的请求，如果 app_custom 的值为“openid=002”，说明 C 响应的是 B 的请求。

1.8 赠送礼物

特别声明：该接口目前仅支持白名单应用。出于信息安全的考虑，本接口目前升级中。升级完成前暂不接受开发者的授权申请。升级完成后，所有接入 QQ 登录的开发者将均可调用。 已接入过该接口的开发者，不需要更改调用的接口，将自动兼容。

通过调用 Tencent 类的 gift 函数赠送礼物。

赠送礼物接口的参数与发送请求的参数相同，调用赠送礼物接口的示例代码如下：

```
private void onClickGift() {
    if (mTencent.isSessionValid() && mTencent.getOpenId() != null) {
        Bundle b = new Bundle();
        b.putString(SocialConstants.PARAM_RECEIVER, "15762FF138EE42D88FE2234D3B89C44B");
        b.putString(SocialConstants.PARAM_TITLE, "title 字段测试");

        b.putString(SocialConstants.PARAM_SEND_MSG, "msg 字段测试");
        b.putString(SocialConstants.PARAM_IMG_URL,
"http://i.gtimg.cn/qzonestyle/act/qzone_app_img/app888_888_75.png");;
        b.putString("exclude", "");
        b.putString("specified", "");
        b.putString("only", "0"); // 0 或者 1
```

```
            b.putString(SocialConstants.PARAM_SOURCE, URLEncoder.encode(""));
            mTencent.gift(this, b, new BaseUiListener());
        }
}
```

调用赠送礼物接口的参数与发送请求的参数相同。

1.9 发送挑战

特别声明：该接口目前仅支持白名单应用。出于信息安全的考虑，本接口目前升级中。升级完成前暂不接受开发者的授权申请。升级完成后，所有接入 QQ 登录的开发者将均可调用。 已接入过该接口的开发者，不需要更改调用的接口，将自动兼容。

通过调用 Tencent 类的 challenge 函数发送挑战信息。

调用发送挑战接口的示例代码如下：

```
private void onClickChallenge() {
        if (mTencent.isSessionValid() && mTencent.getOpenId() != null) {
            Bundle b = new Bundle();
            b.putString(SocialConstants.PARAM_RECEIVER, "15762FF138EE42D88FE2234D3B89C44B");
            b.putString(SocialConstants.PARAM_SEND_MSG, "向某某某发起挑战");
            b.putString(SocialConstants.PARAM_IMG_URL,
"http://i.gtimg.cn/qzonestyle/act/qzone_app_img/app888_888_75.png");;
            b.putString(SocialConstants.PARAM_SOURCE, URLEncoder.encode(""));
            mTencent.challenge(this, b, new BaseUiListener());
        }
}
```

调用挑战接口的参数说明如下：

参数	是否必传	类型	参数说明
SocialConstants.PARAM_RECEIVER	必传	String	■ ■ /挑战的用户的 openid，应用传入的 openid 必须为当前用户的 QQ 好友且对方必须为已安装用户 ■ Receiver 只能为 1 个 ,每天针对同一个好友只能挑战/炫耀一次

SocialConstants.PARAM_SEND_MSG	必传	String	炫耀/挑战中的内容描述。 ✓ 长度限制：50 个汉字（100 个字符）以内，超过限制则自动从末尾截断
SocialConstants.PARAM_IMG_URL	必传	String	炫耀/挑战场景图的 URL
SocialConstants.PARAM_SOURCE	可选	String	透传参数，由开发者自定义该参数内容。 开发者可根据这个参数的内容，判断被炫耀/挑战的用户是点击哪个好友发送的信息。 例如： 用户 A 向 C 发送炫耀/挑战信息时，source 的值为“openid=001”，用户 B 向 C 发送炫耀/挑战信息时，source 的值为“openid=002”。当用户 C 点击信息链接进入应用时，url 中会携带 app_custom 参数，如果 app_custom 的值为“openid=001”，说明 C 点击的信息是 A 发送的，如果 app_custom 的值为“openid=002”，说明 C 点击的信息是 B 发送的。

1.10 炫耀

特别声明：该接口目前仅支持白名单应用。出于信息安全的考虑，本接口目前升级中。升级完成前暂不接受开发者的授权申请。升级完成后，所有接入 QQ 登录的开发者将均可调用。 已接入过该接口的开发者，不需要更改调用的接口，将自动兼容。

通过调用 Tencent 类的 brag 函数实现向好友炫耀的功能。

调用炫耀接口的示例代码如下：

```
private void onClickBrag() {
    if (mTencent.isSessionValid() && mTencent.getOpenId() != null) {
        Bundle b = new Bundle();
        b.putString(SocialConstants.PARAM_RECEIVER, "15762FF138EE42D88FE2234D3B89C44B");

        b.putString(SocialConstants.PARAM_SEND_MSG, "向某某某炫耀一下");

        b.putString(SocialConstants.PARAM_IMG_URL,
"http://i.gtimg.cn/qzonestyle/act/qzone_app_img/app888_888_75.png");;
        b.putString(SocialConstants.PARAM_SOURCE, URLEncoder.encode(""));
        mTencent.brag(this, b, new BaseUiListener());
```

```
    }
}
```

调用炫耀接口的参数与发送挑战的参数相同。

1.11 分享消息到 QQ（无需 QQ 登录）

分享消息到 QQ 的接口，可将新闻、图片、文字、应用等分享给 QQ 好友、群和讨论组。Tencent 类的 shareToQQ 函数可直接调用，不用用户授权（使用手机 QQ 当前的登录态）。调用将打开分享的界面，用户选择好友、群或讨论组之后，点击确定即可完成分享，并进入与该好友进行对话的窗口。

本接口支持 3 种模式，每种模式的参数设置不同，下面分别进行介绍：

（1） 分享图文消息

调用分享接口的示例代码如下：

```
private void onClickShare() {
    final Bundle params = new Bundle();
    params.putInt(QQShare.SHARE_TO_QQ_KEY_TYPE, Tencent.SHARE_TO_QQ_TYPE_DEFAULT);
    params.putString(QQShare.SHARE_TO_QQ_TITLE, "要分享的标题");

    params.putString(QQShare.SHARE_TO_QQ_SUMMARY,  "要分享的摘要");
    params.putString(QQShare.SHARE_TO_QQ_TARGET_URL,  "http://www.qq.com/news/1.html");
    params.putString(QQShare.SHARE_TO_QQ_IMAGE_URL,
"http://imgcache.qq.com/qzone/space_item/pre/0/66768.gif");
    params.putString(QQShare.SHARE_TO_QQ_APP_NAME,  "测试应用 222222");

    params.putInt(QQShare.SHARE_TO_QQ_EXT_INT,  "其他附加功能");
    mTencent.shareToQQ(MainActivity.this, params, new BaseUiListener());
}
```

调用分享接口的 params 参数说明如下：

参数	是否必传	类型	参数说明
QQShare.SHARE_TO_QQ_KEY_TYPE	必填	Int	分享的类型。图文分享(普通分享)填 *Tencent.SHARE_TO_QQ_TYPE_DEFAULT*

QQShare.PARAM_TARGET_URL	必填	String	这条分享消息被好友点击后的跳转 URL。
QQShare.PARAM_TITLE	必填	String	分享的标题，最长 30 个字符。
QQShare.PARAM_SUMMARY	可选	String	分享的消息摘要，最长 40 个字。
QQShare.SHARE_TO_QQ_IMAGE_URL	可选	String	分享图片的 URL 或者本地路径
QQShare.SHARE_TO_QQ_APP_NAME	可选	String	手 Q 客户端顶部，替换“返回”按钮文字，如果为空，用返回代替
QQShare.SHARE_TO_QQ_EXT_INT	可选	int	分享额外选项，两种类型可选（默认是不隐藏分享到 QZone 按钮且不自动打开分享到 QZone 的对话框）： Tencent.SHARE_TO_QQ_FLAG_QZONE_AUTO_OPEN，分享时自动打开分享到 QZone 的对话框。 Tencent.SHARE_TO_QQ_FLAG_QZONE_ITEM_HIDE，分享时隐藏分享到 QZone 按钮

（2） 分享纯图片

调用分享接口的示例代码如下：

```
private void onClickShare() {
        Bundle params = new Bundle();
    params.putString(QQShare.SHARE_TO_QQ_IMAGE_LOCAL_URL,imageUrl.getText().toString());
        params.putString(QQShare.SHARE_TO_QQ_APP_NAME, appName.getText().toString());
        params.putInt(QQShare.SHARE_TO_QQ_KEY_TYPE, QQShare.
                SHARE_TO_QQ_TYPE_IMAGE);
        params.putInt(QQShare.SHARE_TO_QQ_EXT_INT, QQShare.
                SHARE_TO_QQ_FLAG_QZONE_AUTO_OPEN);
        mTencent.shareToQQ(MainActivity.this, params, new BaseUiListener());
}
```

参数	是否必传	类型	参数说明
QQShare.SHARE_TO_QQ_KEY_TYPE	必选	int	分享类型，分享纯图片时填写 *Tencent.SHARE_TO_QQ_TYPE_IMAGE*
QQShare.SHARE_TO_QQ_IMAGE_LOCAL_URL	必选	String	需要分享的本地图片路径
QQShare.SHARE_TO_QQ_APP_NAME	可选	String	手 Q 客户端顶部，替换“返回”按钮文字，如果为空，用返回代替
QQShare.SHARE_TO_QQ_EXT_INT	可选	int	分享额外选项，两种类型可选（默认是不隐藏分享到 QZone 按钮且不自

			动打开分享到 QZone 的对话框）： Tencent.SHARE_TO_QQ_FLAG_QZONE_AUTO_OPEN，分享时自动打开分享到 QZone 的对话框。 Tencent.SHARE_TO_QQ_FLAG_QZONE_ITEM_HIDE，分享时隐藏分享到 QZone 按钮

（3） 分享音乐

音乐分享后，发送方和接收方在聊天窗口中点击消息气泡即可开始播放音乐。

调用分享接口的示例代码如下：

```
private void onClickAudioShare() {
        final Bundle params = new Bundle();
        params.putInt(QQShare.SHARE_TO_QQ_KEY_TYPE, Tencent.SHARE_TO_QQ_TYPE_AUDIO);

        params.putString(QQShare.SHARE_TO_QQ_TITLE, "要分享的标题");


        params.putString(QQShare.SHARE_TO_QQ_SUMMARY,    "要分享的摘要");

        params.putString(QQShare.SHARE_TO_QQ_TARGET_URL,    "http://www.qq.com/news/1.html");
        params.putString(QQShare.SHARE_TO_QQ_IMAGE_URL,
"http://imgcache.qq.com/qzone/space_item/pre/0/66768.gif");

        params.putString(QQShare.SHARE_TO_QQ_AUDIO_URL, "音乐链接");


        params.putString(QQShare.SHARE_TO_QQ_APP_NAME,    "测试应用 222222");

      params.putInt(QQShare.SHARE_TO_QQ_EXT_INT, QQShare.SHARE_TO_QQ_FLAG_QZONE_AUTO_OPEN
);            mTencent.shareToQQ(MainActivity.this, params, new BaseUiListener());
}
```

调用分享接口的 params 参数说明如下：

参数	是否必传	类型	参数说明
QQShare.SHARE_TO_QQ_KEY_TYPE	必填	Int	分享的类型。分享音乐填 Tencent.SHARE_TO_QQ_TYPE_AUDIO
QQShare.PARAM_TARGET_URL	必选	String	这条分享消息被好友点击后的跳转 URL。
QQShare.SHARE_TO_QQ_AUDIO_URL	必填	String	音乐文件的远程链接，以 URL 的形式传入，不支持本地音乐。
QQShare.PARAM_TITLE	必选	String	分享的标题，最长 30 个字符。
QQShare.PARAM_SUMMARY	可选	String	分享的消息摘要，最长 40 个字符。
QQShare.SHARE_TO_QQ_IMAGE_URL	可选	String	分享图片的 URL 或者本地路径

QQShare.SHARE_TO_QQ_APP_NAME	可选	String	手 Q 客户端顶部，替换“返回”按钮文字，如果为空，用返回代替
QQShare.SHARE_TO_QQ_EXT_INT	可选	int	分享额外选项，两种类型可选（默认是不隐藏分享到 QZone 按钮且不自动打开分享到 QZone 的对话框）： Tencent.SHARE_TO_QQ_FLAG_QZONE_AUTO_OPEN，分享时自动打开分享到 QZone 的对话框。 Tencent.SHARE_TO_QQ_FLAG_QZONE_ITEM_HIDE，分享时隐藏分享到 QZone 按钮

（4） 分享应用

应用分享后，发送方和接收方在聊天窗口中点击消息气泡即可进入应用的详情页。

调用分享接口的示例代码如下：

```
private void onClickAppShare() {
    final Bundle params = new Bundle();
    params.putInt(QQShare.SHARE_TO_QQ_KEY_TYPE, Tencent.SHARE_TO_QQ_TYPE_APP);

    params.putString(QQShare.SHARE_TO_QQ_TITLE, "要分享的标题");


    params.putString(QQShare.SHARE_TO_QQ_SUMMARY,  "要分享的摘要");

    params.putString(QQShare.SHARE_TO_QQ_IMAGE_URL,
"http://imgcache.qq.com/qzone/space_item/pre/0/66768.gif");

    params.putString(QQShare.SHARE_TO_QQ_AUDIO_URL, "音乐链接");


    params.putString(QQShare.SHARE_TO_QQ_APP_NAME,  "测试应用 222222");

   params.putInt(QQShare.SHARE_TO_QQ_EXT_INT, QQShare.SHARE_TO_QQ_FLAG_QZONE_AUTO_OPEN
);          mTencent.shareToQQ(MainActivity.this, params, new BaseUiListener());
}
```

调用分享接口的 params 参数说明如下：

参数	是否必传	类型	参数说明
QQShare.SHARE_TO_QQ_KEY_TYPE	必填	Int	分享的类型。分享音乐填 Tencent.SHARE_TO_QQ_TYPE_APP
QQShare.PARAM_TITLE	必选	String	分享的标题，最长 30 个字符。
QQShare.PARAM_SUMMARY	可选	String	分享的消息摘要，最长 40 个字符。
QQShare.SHARE_TO_QQ_IMAGE_URL	可选	String	分享图片的 URL 或者本地路径

QQShare.SHARE_TO_QQ_APP_NAME	可选	String	手 Q 客户端顶部，替换“返回”按钮文字，如果为空，用返回代替
QQShare.SHARE_TO_QQ_EXT_INT	可选	int	分享额外选项，两种类型可选（默认是不隐藏分享到 QZone 按钮且不自动打开分享到 QZone 的对话框）： Tencent.SHARE_TO_QQ_FLAG_QZONE_AUTO_OPEN，分享时自动打开分享到 QZone 的对话框。 Tencent.SHARE_TO_QQ_FLAG_QZONE_ITEM_HIDE，分享时隐藏分享到 QZone 按钮

1.12 分享到 QQ 空间（无需 QQ 登录）

完善了分享到 QZone 功能，分享类型参数 Tencent.SHARE_TO_QQ_KEY_TYPE，目前支持图文分享，发表说说、视频，上传图片。Tencent. shareToQzone()和 Tencent. publishToQzone()函数可直接调用，不用用户授权（使用手机 QQ 当前的登录态），调用后将打开手机 QQ 内 QQ 空间的界面进行分享或发表操作。

（1） 图文分享

示例代码如下：

```
private void shareToQzone () {
    //分享类型
    params.putString(QzoneShare.SHARE_TO_QQ_KEY_TYPE,SHARE_TO_QZONE_TYPE_IMAGE_TEXT );
    params.putString(QzoneShare.SHARE_TO_QQ_TITLE, "标题");//必填
    params.putString(QzoneShare.SHARE_TO_QQ_SUMMARY, "摘要");//选填
    params.putString(QzoneShare.SHARE_TO_QQ_TARGET_URL, "跳转 URL");//必填
    params.putStringArrayList(QzoneShare.SHARE_TO_QQ_IMAGE_URL, "图片链接 ArrayList");
    mTencent.shareToQzone(activity, params, new BaseUiListener());
}
```

params 参数说明如下：

参数	是否必传	类型	参数说明
QzoneShare.SHARE_TO_QQ_KEY_TYPE	选填	Int	SHARE_TO_QZONE_TYPE_IMAGE_TEXT（图文）
QzoneShare.SHARE_TO_QQ_TITLE	必填	Int	分享的标题，最多 200 个字符
QzoneShare.SHARE_TO_QQ_SUMMARY	选填	String	分享的摘要，最多 600 字符
QzoneShare.SHARE_TO_QQ_TARGET_URL	必填	String	需要跳转的链接，URL 字符串
QzoneShare.SHARE_TO_QQ_IMAGE_URL	选填	String	分享的图片，以 ArrayList<String>的类型传入，以便支持多张图片（注：图片最多支持 9 张图片，多余的图片会被丢弃）。

注意:
QZone接口暂不支持发送多张图片的能力，若传入多张图片，则会自动选入第一张图片作为预览图。多图的能力将会在以后支持。

（2） 发表说说、视频或上传图片

示例代码如下：

```
private void publishToQzone () {
    //分享类型
    params.putString(QzoneShare.SHARE_TO_QZONE_KEY_TYPE, PUBLISH_TO_QZONE_TYPE_PUBLISHMOOD );
    params.putString(QzoneShare.SHARE_TO_QQ_SUMMARY, "摘要");

    params.putStringArrayList(QzoneShare.SHARE_TO_QQ_IMAGE_URL, "图片链接 ArrayList");

    params.putString(QzonePublish.PUBLISH_TO_QZONE_VIDEO_PATH,"本地视频地址");
    Bundle extParams = new Bundle();
    extParams.putString (QzonePublish. HULIAN_EXTRA_SCENE, "分享场景" );

    extParams.putString (QzonePublish. HULIAN_CALL_BACK, "回调信息" );
    params.putBundle(QzonePublish.PUBLISH_TO_QZONE_EXTMAP, extParams);
    mTencent.publishToQzone(activity, params, new BaseUiListener());
}
```

params 参数说明如下：

参数	是否必传	类型	参数说明
PUBLISH_TO_QZONE_KEY_TYPE	必填	Int	QzonePublish.PUBLISH_TO_QZONE_TYPE_PUBLISHMOOD（发表说说、上传图片） QzonePublish.PUBLISH_TO_QZONE_TYPE_PUBLISHVIDEO（发表视频）
PUBLISH_TO_QZONE_SUMMARY	选填	String	说说正文（传图和传视频接口会过滤第三方传过来的自带描述，目的为了鼓励用户自行输入有价值信息）
PUBLISH_TO_QZONE_IMAGE_URL	选填	ArrayList	说说的图片,以 ArrayList<String>的类型传入，以便支持多张图片（注：<=9 张图片为发表说说，>9 张为上传图片到相册），只支持本地图片
PUBLISH_TO_QZONE_VIDEO_PATH	选填	String	发表的视频，只支持本地地址，发表视频时必填；上传视频的大小最好控制在 100M 以内（因为 QQ 普通用户上传视频必须在 100M 以内，黄钻用户可上传 1G 以内视频，大于 1G 会直接报错。）
HULIAN_EXTRA_SCENE	选填	String	区分分享场景，用于异化 feeds 点击行为和小尾巴展示
HULIAN_CALL_BACK	选填	String	游戏自定义字段，点击分享消息回到游戏时回传给游戏

1.13 更改获取用户信息接口

调用获取用户信息接口的示例代码如下：

```
UserInfo info = new UserInfo(this, MainActivity.mQQAuth.getQQToken());
info.getUserInfo(new BaseUIListener(this,"get_simple_userinfo"));
```

1.14 发布带图微博

用于发送一个带有图片的微博。

示例代码如下:

```
private void onClickUserInfo() {

Bundle bundle = new Bundle();
                bundle.putString("format", "json");
                bundle.putString("content", "test add pic with url");

                 // 把 bitmap 转换为 byteArray，用于发送请求

                Bitmap bitmap = BitmapFactory.decodeResource(getResources(), R.drawable.ic_launcher);
                ByteArrayOutputStream baos = new ByteArrayOutputStream();
```

```
                bitmap.compress(Bitmap.CompressFormat.JPEG, 40, baos);
                byte[] buff = baos.toByteArray();
                // Log.v(TAG, "length: " + buff.length);
                bundle.putByteArray("pic", buff);
                mTencent.requestAsync(SocialConstants.GRAPH_ADD_PIC_T, bundle,
                        SocialConstants.HTTP_POST, new BaseApiListener("add_pic_t", false), null);
                bitmap.recycle();

                mProgressDialog.show();
}
```

调用发送带图微博接口的 bundle 参数说明请参阅 http://wiki.connect.qq.com/add_pic_t

1.15 语音 API(新增)

特别声明：该接口目前仅支持白名单应用。出于信息安全的考虑，本接口目前升级中。升级完成前暂不接受开发者的授权申请。升级完成后，所有接入 QQ 登录的开发者将均可调用。 已接入过该接口的开发者，不需要更改调用的接口，将自动兼容。

调用语音 API 的 params 参数说明如下：

V2.0 新增语音 API，可以向 QQ 好友发送一段录制的语音。

示例代码如下：

```
    private void onClickVoice() {
        Log.i("sample", "onClickVoice");
            if (ready()) {
                if(voiceBundle == null){
                    voiceBundle = new Bundle();
                    // TODO keywords.
                    voiceBundle.putString(SocialSocialConstants.PARAM_APP_ICON,
                            "http://imgcache.qq.com/qzone/space_item/pre/0/66768.gif");
                    voiceBundle.putString(SocialSocialConstants.PARAM_APP_DESC,
                            "AndroidSdk_2_0: invite description!");

                    voiceBundle.putString(SocialSocialConstants.PARAM_ACT, "进入应用");

                }
                mTencent.voice(MainActivity.this, voiceBundle, new BaseUiListener());
        }
```

参数	是否必传	类型	参数说明
SocialConstants.PARAM_RECEIVER	必传	String	若 receiver 包含 OpenID，则表示使用模式一（应用指定接收者模式）;若 receiver 为空，代表使用模式二（用户指定接收者模式） 3) 传入的 OpenID 必须为应用内关系链，即为已安装该应用的 QQ 好友 4) 传入个数不得超过 10 个，若超过 10 个则自动截取前 10 个，剩余舍弃 授权使用应用内关系链的应用才可使用模式一（应用指定接收者）
SocialConstants.PARAM_IMG_DATA	必传	String	语音配图的二进制文件，若应用直接传入图片文件，则发送语音时附带上传。应用传入图片 size 不超过 320*320，否则会被等比压缩
SocialConstants.PARAM_IMG_URL	必传	String	语音配图的 url 连接，该参数与 imgData 参数二选一。 如果 imgData 和 url 都传了，则优先选择 imgData
SocialConstants.PARAM_EXCLUDE	可选	String	在用户自己指定好友的场景中，如果开发者不希望某些用户显示在好友选择器中，可传入这些用户的 openid，多个 openid 之间用“,”分隔（“exclude”和“specified”一共最多可传入 50 个 openid），好友选择器则不显示这些 openid 对应的用户。
SocialConstants.PARAM_SPECIFIED	可选	String	在用户自己指定好友的场景中，如果开发者需要在好友选择器中显示指定的用户，可传入这些用户的 openid，多个 openid 之间用“,”分隔（“exclude”和“specified”一共最多可传入 50 个 openid)，好友选

			择器会显示这些 openid 对应的用户。
SocialConstants.PARAM_ONLY	可选	String	仅当设置了 PARAM_SPECIFIED，需要传入该参数，用于标识是否在好友选择器中只显示 PARAM_SPECIFIED 指定的用户。默认值为“0”。 “0”：显示 PARAM_SPECIFIED 指定的用户，同时显示好友列表和已安装好友。 “1”：只显示“specified”指定的用户，无好友列表和已安装好友。此时必须保证 PARAM_SPECIFIED 有可用的 openid 传入，否则会报错。
SocialConstants.PARAM_SOURCE	可选	String	由开发者自定义该参数内容，用于判断用户接收的哪个好友的礼物或请求。 例如： 用户 A 向 C 赠送礼物/发送好友请求时，source 的值为“openid=001”，用户 B 向 C 赠送礼物/发送好友请求时，source 的值为“openid=002”。当用户 C 点击链接进入免费礼物或好友请求的查看页面时，url 中会携带 app_custom 参数，app_custom 的值为“openid=001”，说明 C 响应的是 A 的请求，如果 app_custom 的值为“openid=002”，说明 C 响应的是 B 的请求。

1.16 应用评价

应用评价的基本流程为：

（1）判断用户是否已经登录，且应用是否获取了 openid；

（2）创建一个 Bundle，并填入必要参数；

（3）调用 SDK 中 Tencent 类的 grade 接口，启动评价界面。

调用应用评价接口的示例代码如下：

```
private void onClickAppGrade() {
    if (ready()) {
        Bundle params = new Bundle();
        params.putString("comment", "亲，给个好评吧");
        mTencent.grade(MainActivity.this, params, new BaseUiListener());
    }
}
```

1.17 好友召回（新增）

特别声明：该接口目前仅支持白名单应用。出于信息安全的考虑，本接口目前升级中。升级完成前暂不接受开发者的授权申请。升级完成后，所有接入 QQ 登录的开发者将均可调用。 已接入过该接口的开发者，不需要更改调用的接口，将自动兼容。

通过调用 Tencent 类的 reactive 函数发起好友召回。除了接受者能获得礼物以外，发送者也接收相应礼物。

调用发送请求接口的示例代码如下：

```
private void onClickReactive() {
    if (mTencent.isReady()) {
        Bundle params = new Bundle();
        params.putString(SocialConstants.PARAM_TITLE, "title字段测试");
        params.putString(SocialConstants.PARAM_SEND_MSG, "msg字段测试");
params.putString(SocialConstants.PARAM_IMG_URL,"http://i.gtimg.cn/qzonestyle/act/qzone_app_img/app888_888_75.
```

```
png");
        params.putString(SocialConstants.PARAM_REC_IMG,
"http://i.gtimg.cn/qzonestyle/act/qzone_app_img/app888_888_75.png");
        params.putString(SocialConstants.PARAM_REC_IMG_DESC, "发送者获取礼物描述");
        mTencent.reactive(this, params, new BaseUIListener(SocialApiActivity.this));
    }
}
```

调用发送请求接口的参数说明如下：

参数	是否必传	类型	参数说明
SocialConstants.PARAM_TITLE	必传	String	应用奖励给发起召回的用户的奖品的名称
SocialConstants.PARAM_SEND_MSG	必传	String	召回老友时的默认赠言，用户可修改，默认赠言请尽量简洁生动
SocialConstants.PARAM_IMG_URL	必传	String	赠送给好友的礼物的图片 url
SocialConstants.PARAM_REC_IMG	必传	String	赠送给发送者的礼物图片 url
SocialConstants.PARAM_REC_IMG_DESC	必传	String	赠送给发送者的礼物的描述信息
SocialConstants.PARAM_EXCLUDE	可选	String	在用户自己指定好友的场景中，如果开发者不希望某些用户显示在好友选择器中，可传入这些用户的 openid，多个 openid 之间用“,”分隔（“exclude”和“specified”一共最多可传入 50 个 openid），好友选择器则不显示这些 openid 对应的用户。
SocialConstants.PARAM_SPECIFIED	可选	String	在用户自己指定好友的场景中，如果开发者需要在

			好友选择器中显示指定的用户，可传入这些用户的 openid，多个 openid 之间用“,”分隔（“exclude”和“specified”一共最多可传入 50 个 openid），好友选择器会显示这些 openid 对应的用户。
SocialConstants.PARAM_ONLY	可选	String	仅当设置了“specified”，需要传入该参数，用于标识是否在好友选择器中只显示“specified”指定的用户。默认值为“0”。 “0”：显示“specified”指定的用户，同时显示好友列表和已安装好友。 “1”：只显示“specified”指定的用户，无好友列表和已安装好友。此时必须保证“specified”有可用的 openid 传入，否则会报错。
SocialConstants.PARAM_SOURCE	可选	String	由开发者自定义该参数内容，用于判断用户接收的哪个好友的礼物或请求。 例如： 用户 A 向 C 发送好友召回时，source 的值为“openid=001”，用户 B 向 C 发送好友召回时，source 的值为“openid=002”。当用户 C 点击链接进入请求的查看页面时，url 中会携带 app_custom 参数，app_custom 的值为“openid=001”，说明 C 响应的是 A 的请求，如果 app_custom 的值为“openid=002”，说明 C

			响应的是 B 的请求。

1.18 查询附近的人（新增）

特别声明：出于信息安全的考虑，本接口仅对可信赖的合作应用开放。已经成功接入“QQ 登录”的应用需提交申请以获取访问本接口的权限。

查询附近都在玩此应用的人，通过调用 Tencent 类的 searchNearby 函数查询附近都在玩此应用的人

使用此接口前需要在配置文件配置如下权限信息：

```
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_LOCATION_EXTRA_COMMANDS" />
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
```

调用发送请求接口的示例代码如下：

```
private void onClickSearchNearby() {
        if (mTencent.isReady()) {
            BaseUIListener l = new BaseUIListener(SocialApiActivity.this);
            mTencent.searchNearby(SocialApiActivity.this, null, l);
        }
    }
```

调用接口返回 json 字段说明：

key	value
total	int 型，返回附近人个数
openids	Json 数组，所有的附近人数组
nickname	String 类型，所属 openids 组，昵称
gender	int 型，所属 openids 组，性别，男为1，女为2，-1为未知

distance	int 型，所属 openids 组，用户离当前调用者的距离，单位“米”
page	int 型，查询附近的人以 page 为单位，每页做多100个，默认 page 为1，要显示更多信息，可在 params 参数指定 page=2等等

1.19 删除用户位置信息（新增）

特别声明：出于信息安全的考虑，本接口仅对可信赖的合作应用开放。已经成功接入“QQ 登录”的应用需提交申请以获取访问本接口的权限。

查询过位置信息的用户，后台保留了位置信息，添加入口给用户删除自己上传的位置信息，保护隐私。通过调用 Tencent 类的 deleteLocation 函数清除用户位置信息

示例代码如下：

```
private void onClickDeleteLocation() {
        if (mTencent.isReady()) {
            BaseUIListener l = new BaseUIListener(SocialApiActivity.this);
            mTencent.deleteLocation(SocialApiActivity.this, null, l);
        }
    }
```

1.20 启动应用吧

通过调用 Tencent 类的 startAppbar 函数启动应用吧相关界面，主要包含如下页面：应用吧详情页、我的消息页及发帖页面，通过参数标识跳转页面，参数如下：

AppbarAgent.TO_APPBAR_DETAIL //打开应用吧详情页

AppbarAgent.TO_APPBAR_NEWS //打开我的消息页

AppbarAgent.TO_APPBAR_SEND_BLOG //打开发帖页

使用此接口前需要在配置文件配置如下信息：

```
<activity
        android:name="com.tencent.open.yyb.AppbarActivity"
        android:configChanges="orientation|keyboardHidden|screenSize"
        android:screenOrientation="portrait"
        android:theme="@android:style/Theme.Translucent.NoTitleBar" />
```

调用接口的示例代码如下：

```
mTencent = Tencent.createInstance(APPID, SocialAppbarActivity.this);
mTencent.startAppbar(SocialAppbarActivity.this, AppbarAgent.TO_APPBAR_DETAIL);
```

1.21 打开应用吧标签页

通过调用 Tencent 类的 startAppbarLabel 函数启动应用吧标签页面，参数如下：

Label：需要跳转的标签值

和 1.21 一样需要配置 AppbarActivity 属性，详见 1.21 说明，调用接口的实例代码如下

```
mTencent = Tencent.createInstance(APPID, SocialAppbarActivity.this);
mTencent.startAppbarLabel(SocialAppbarActivity.this, label);
```

1.22 一键加群

通过调用 Tencent 类的 joinQQGroup 调用加群接口，参数如下：

Key：需要添加群的 ID（申请 ID 请到 qun.qq.com 申请）

调用接口的示例代码如下：

```
mTencent = Tencent.createInstance(APPID, this);
mTencent.joinQQGroup(this, key);
```

1.23 一键绑群

通过调用 Tencent 类的 bindQQGroup 接口将公会与 QQ 群绑定，参数如下：

Params：Bundle 类型调用参数，必须包含：unionid, union_name, zoneid, signature 字段

参数含义如下：

参数	value
GameAppOperation.GAME_UNION_ID	String 型，公会 ID（一般为数字）
GameAppOperation.GAME_UNION_NAME	String 型，公会名称
GameAppOperation.GAME_ZONE_ID	String 型，区域 ID（一般为数字）
GameAppOperation.GAME_SIGNATURE	String 型，游戏盟主身份验证签名，从游戏后台获取

调用接口的示例代码如下：

```
Bundle params = new Bundle();
params.putString(GameAppOperation.GAME_UNION_ID, tvGameUnionId.getText() + "");
params.putString(GameAppOperation.GAME_ZONE_ID, tvGameZoneId.getText() + "");
params.putString(GameAppOperation.GAME_SIGNATURE, tvGameSignature.getText() + "");
params.putString(GameAppOperation.GAME_UNION_NAME, tvGameUnionName.getText() + "");

mTencent = Tencent.createInstance(APPID, this);
mTencent.bindQQGroup(activity, params);
```

1.24 添加好友

通过调用 Tencent 类的 makeFriend 调用添加好友接口，参数如下：

Params：Bundle 类型调用参数，必须包含：openid, label, message 字段

参数含义如下：

参数	value
Opened	String 型，要添加好友的 openid
Label	String 型，要添加好友的备注
Message	String 型，验证信息

调用接口的示例代码如下：

```
Bundle params = new Bundle();
params.putString(GameAppOperation.GAME_FRIEND_OPENID, fopenid.getText() + "");
```

```
params.putString(GameAppOperation.GAME_FRIEND_LABEL, label.getText() + "");
params.putString(GameAppOperation.GAME_FRIEND_ADD_MESSAGE, message.getText() + "");
mTencent.makeFriend(GameLogicActivity.this, params);
```

1.25 收藏信息到 QQ 我的收藏（无需 QQ 登录）

收藏信息到 QQ 我的收藏的接口，可将文本、图文、咨询连接（含视频连接）、音乐连接等发送到 QQ【我的收藏】。

Tencent 类的 addToQQFavorites 函数可直接调用，不用用户授权（使用手机 QQ 当前的登录态）。调用该接口将会有一个左图右文的弹窗让用户确认。

调用本接口,需要配置 AndroidManifest，在 com.tencent.tauth.AuthActivity 中添加 android:theme 属性：

```
<activity
        android:name="com.tencent.tauth.AuthActivity"
        android:noHistory="true"
       android:theme="@android:style/Theme.Translucent"
        android:launchMode="singleTask" >
```

本接口支持 4 种模式，每种模式的参数设置不同，下面分别进行介绍：

（1） 文本类信息收藏

调用文本信息收藏接口的示例代码如下：

```
final Bundle params = new Bundle();
params.putString(GameAppOperation.QQFAV_DATALINE_APPNAME, appName.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_TITLE, title.getText().toString());
params.putInt(GameAppOperation.QQFAV_DATALINE_REQTYPE, GameAppOperation.QQFAV_DATALINE_TYPE_TEXT);
params.putString(GameAppOperation.QQFAV_DATALINE_DESCRIPTION, description.getText().toString());
mTencent.addToQQFavorites (activity, params, new BaseUiListener());
```

调用文本信息收藏接口的 params 参数说明如下：

参数	是否必传	类型	参数说明
GameAppOperation.QQFAV_DATALINE_APPNAME	必填	String	应用的名称，SDK 会通过传入的 activity 得到应用的名称，如果通过 activity 得不到应用名称，则会使用传入的这个 AppName。
GameAppOperation.QQFAV_DATALINE_TITLE	可选	String	收藏信息的标题。
GameAppOperation.QQFAV_DATALINE	必填	int	收藏的类型，纯文本类型为：

_REQTYPE			GameAppOperation.QQFAV_DATALINE_TYPE_TEXT
GameAppOperation.QQFAV_DATALINE_DESCRIPTION	必填	String	收藏信息的正文内容。

（2） 图文类信息收藏

调用图文信息收藏接口的示例代码如下：

```
final Bundle params = new Bundle();
params.putString(GameAppOperation.QQFAV_DATALINE_APPNAME, appName.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_TITLE, title.getText().toString());
params.putInt(GameAppOperation.QQFAV_DATALINE_REQTYPE,GameAppOperation.QQFAV_DATALINE_TYPE_IMAGE_TEXT);
params.putString(GameAppOperation.QQFAV_DATALINE_DESCRIPTION, description.getText().toString()
                                                + GameAppOperation.PIC_SYMBOLE);
params.putStringArrayList(GameAppOperation.QQFAV_DATALINE_FILEDATA, fileDataList);
mTencent.addToQQFavorites (activity, params, new BaseUiListener());
```

调用图文信息收藏接口的 params 参数说明如下：

参数	是否必传	类型	参数说明
GameAppOperation.QQFAV_DATALINE_APPNAME	必填	String	应用的名称，SDK 会通过传入的 activity 得到应用的名称，如果通过 activity 得不到应用名称，则会使用传入的这个 AppName。
GameAppOperation.QQFAV_DATALINE_TITLE	可选	String	收藏信息的标题。
GameAppOperation.QQFAV_DATALINE_REQTYPE	必填	int	收藏的类型，图文类型为：GameAppOperation.QQFAV_DATALINE_TYPE_IMAGE_TEXT
GameAppOperation.QQFAV_DATALINE_DESCRIPTION	可选	String	图片的描述信息，如果有多图，则每个图片对应的描述信息使用 GameAppOperation.PIC_SYMBOLE 来隔开
GameAppOperation.QQFAV_DATALINE_FILEDATA	必填	ArrayList <String>	图片路径 List：包含本地图片和网络图片，可以有多个。

（3） 音乐类信息收藏

音乐收藏后，在手机 QQ【我的手藏】里面点击消息气泡即可开始播放音乐。

调用音乐类信息收藏接口的示例代码如下：

```
final Bundle params = new Bundle();
params.putString(GameAppOperation.QQFAV_DATALINE_APPNAME, appName.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_TITLE, title.getText().toString());
params.putInt(GameAppOperation.QQFAV_DATALINE_REQTYPE,GameAppOperation.QQFAV_DATALINE_TYPE_AUDIO);
params.putString(GameAppOperation.QQFAV_DATALINE_DESCRIPTION, description.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_IMAGEURL, imageUrl.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_URL, targetUrl.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_AUDIOURL, audioUrl.getText().toString());
mTencent.addToQQFavorites (activity, params, new BaseUiListener());
```

调用音乐类信息收藏接口的 params 参数说明如下：

参数	是否必传	类型	参数说明
GameAppOperation.QQFAV_DATALINE_APPNAME	必填	String	应用的名称，SDK 会通过传入的 activity 得到应用的名称，如果通过 activity 得不到应用名称，则会使用传入的这个 AppName。
GameAppOperation.QQFAV_DATALINE_TITLE	可选	String	收藏信息的标题。
GameAppOperation.QQFAV_DATALINE_REQTYPE	必填	int	收藏的类型，音乐类型为：GameAppOperation.QQFAV_DATALINE_TYPE_AUDIO
GameAppOperation.QQFAV_DATALINE_DESCRIPTION	可选	String	音乐的描述信息。
GameAppOperation.QQFAV_DATALINE_IMAGEURL	必填	String	预览图地址：可以为本地图片地址或网络图片地址
GameAppOperation.QQFAV_DATALINE_URL	必填	String	详情页地址
GameAppOperation.QQFAV_DATALINE_AUDIOURL	必填	String	音乐播放地址

（4） 资讯类（包含视频）收藏

资讯类（包含视频）收藏后，在手机 QQ【我的手藏】里面点击消息气泡即可进入资讯类的详情页。

调用资讯类（包含视频）收藏接口的示例代码如下：

```
final Bundle params = new Bundle();
params.putString(GameAppOperation.QQFAV_DATALINE_APPNAME, appName.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_TITLE, title.getText().toString());
params.putInt(GameAppOperation.QQFAV_DATALINE_REQTYPE,GameAppOperation.QQFAV_DATALINE_TYPE_DEFAULT)
;
params.putString(GameAppOperation.QQFAV_DATALINE_DESCRIPTION, description.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_IMAGEURL, imageUrl.getText().toString());
```

```
params.putString(GameAppOperation.QQFAV_DATALINE_URL, targetUrl.getText().toString());
mTencent.addToQQFavorites (activity, params, new BaseUiListener());
```

调用资讯类（包含视频）收藏接口的 params 参数说明如下：

参数	是否必传	类型	参数说明
GameAppOperation.QQFAV_DATALINE_APPNAME	必填	String	应用的名称，SDK 会通过传入的 activity 得到应用的名称，如果通过 activity 得不到应用名称，则会使用传入的这个 AppName。
GameAppOperation.QQFAV_DATALINE_TITLE	可选	String	收藏信息的标题。
GameAppOperation.QQFAV_DATALINE_REQTYPE	必填	int	收藏的类型，资讯类型为：GameAppOperation. QQFAV_DATALINE_TYPE_DEFAULT
GameAppOperation.QQFAV_DATALINE_DESCRIPTION	可选	String	音乐的描述信息。
GameAppOperation.QQFAV_DATALINE_IMAGEURL	必填	String	预览图地址：可以为本地图片地址或网络图片地址
GameAppOperation.QQFAV_DATALINE_URL	必填	String	详情页地址

1.26 发送信息到 QQ 我的电脑（无需 QQ 登录）

发送信息到 QQ 我的电脑的接口，可将文本、图文、咨询连接（含视频连接）、音乐连接等发送到手机 QQ【我的电脑】。Tencent 类的 sendToMyComputer 函数可直接调用，不用用户授权（使用手机 QQ 当前的登录态）。

建议显示给用户的入口名称为【发送到电脑】。

本接口支持 4 种模式，每种模式的参数设置不同，下面分别进行介绍：

（1） 文本类信息

调用文本信息接口的示例代码如下：

```
final Bundle params = new Bundle();
params.putString(GameAppOperation.QQFAV_DATALINE_APPNAME, appName.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_TITLE, title.getText().toString());
params.putInt(GameAppOperation.QQFAV_DATALINE_REQTYPE, GameAppOperation.QQFAV_DATALINE_TYPE_TEXT);
params.putString(GameAppOperation.QQFAV_DATALINE_DESCRIPTION, description.getText().toString());
mTencent.sendToMyComputer(activity, params, new BaseUiListener());
```

调用文本信息接口的 params 参数说明如下：

参数	是否必传	类型	参数说明
GameAppOperation.QQFAV_DATALINE_APPNAME	必填	String	应用的名称，SDK 会通过传入的 activity 得到应用的名称，如果通过 activity 得不到应用名称，则会使用传入的这个 AppName。
GameAppOperation.QQFAV_DATALINE_TITLE	可选	String	信息的标题。
GameAppOperation.QQFAV_DATALINE_REQTYPE	必填	int	类型，纯文本类型为：GameAppOperation.QQFAV_DATALINE_TYPE_TEXT
GameAppOperation.QQFAV_DATALINE_DESCRIPTION	必填	String	信息的正文内容。

（2） 图文类信息

调用图文信息接口的示例代码如下：

```
final Bundle params = new Bundle();
params.putString(GameAppOperation.QQFAV_DATALINE_APPNAME, appName.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_TITLE, title.getText().toString());
params.putInt(GameAppOperation.QQFAV_DATALINE_REQTYPE,GameAppOperation.QQFAV_DATALINE_TYPE_IMAGE_TEXT);
params.putString(GameAppOperation.QQFAV_DATALINE_DESCRIPTION, description.getText().toString()
                                        + GameAppOperation.PIC_SYMBOLE);
params.putStringArrayList(GameAppOperation.QQFAV_DATALINE_FILEDATA, fileDataList);
mTencent.sendToMyComputer(activity, params, new BaseUiListener());
```

调用图文信息接口的 params 参数说明如下：

参数	是否必传	类型	参数说明
GameAppOperation.QQFAV_DATALINE_APPNAME	必填	String	应用的名称，SDK 会通过传入的 activity 得到应用的名称，如果通过 activity 得不到应用名称，则会使用传入的这个 AppName。
GameAppOperation.QQFAV_DATALINE_TITLE	可选	String	信息的标题。
GameAppOperation.QQFAV_DATALINE_REQTYPE	必填	int	类型，图文类型为：GameAppOperation.QQFAV_DATALINE_TYPE_IMAGE_TEXT
GameAppOperation.QQFAV_DATALINE_DESCRIPTION	可选	String	图片的描述信息，如果有多图，则每个图片对应的描述信息使用 GameAppOperation.PIC_SYMBOLE 来隔开

GameAppOperation.QQFAV_DATALINE_FILEDATA	必填	ArrayList<String>	图片路径 List：包含本地图片和网络图片，可以有多个。

（3） 音乐类信息

调用音乐类信息接口的示例代码如下：

```
final Bundle params = new Bundle();
params.putString(GameAppOperation.QQFAV_DATALINE_APPNAME, appName.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_TITLE, title.getText().toString());
params.putInt(GameAppOperation.QQFAV_DATALINE_REQTYPE,GameAppOperation.QQFAV_DATALINE_TYPE_AUDIO);
params.putString(GameAppOperation.QQFAV_DATALINE_DESCRIPTION, description.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_IMAGEURL, imageUrl.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_URL, targetUrl.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_AUDIOURL, audioUrl.getText().toString());
mTencent.sendToMyComputer (activity, params, new BaseUiListener());
```

调用音乐类信息接口的 params 参数说明如下：

参数	是否必传	类型	参数说明
GameAppOperation.QQFAV_DATALINE_APPNAME	必填	String	应用的名称，SDK 会通过传入的 activity 得到应用的名称，如果通过 activity 得不到应用名称，则会使用传入的这个 AppName。
GameAppOperation.QQFAV_DATALINE_TITLE	可选	String	信息的标题。
GameAppOperation.QQFAV_DATALINE_REQTYPE	必填	int	类型，音乐类型为：GameAppOperation.QQFAV_DATALINE_TYPE_AUDIO
GameAppOperation.QQFAV_DATALINE_DESCRIPTION	可选	String	音乐的描述信息。
GameAppOperation.QQFAV_DATALINE_IMAGEURL	必填	String	预览图地址：可以为本地图片地址或网络图片地址
GameAppOperation.QQFAV_DATALINE_URL	必填	String	详情页地址
GameAppOperation.QQFAV_DATALINE_AUDIOURL	必填	String	音乐播放地址

（4） 资讯类（包含视频）

调用资讯类（包含视频）接口的示例代码如下：

```
final Bundle params = new Bundle();
```

```
params.putString(GameAppOperation.QQFAV_DATALINE_APPNAME, appName.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_TITLE, title.getText().toString());
params.putInt(GameAppOperation.QQFAV_DATALINE_REQTYPE,GameAppOperation.QQFAV_DATALINE_TYPE_DEFAULT)
;
params.putString(GameAppOperation.QQFAV_DATALINE_DESCRIPTION, description.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_IMAGEURL, imageUrl.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_URL, targetUrl.getText().toString());
mTencent.sendToMyComputer(activity, params, new BaseUiListener());
```

调用资讯类（包含视频）接口的 params 参数说明如下：

参数	是否必传	类型	参数说明
GameAppOperation.QQFAV_DATALINE_APPNAME	必填	String	应用的名称，SDK 会通过传入的 activity 得到应用的名称，如果通过 activity 得不到应用名称，则会使用传入的这个 AppName。
GameAppOperation.QQFAV_DATALINE_TITLE	可选	String	信息的标题。
GameAppOperation.QQFAV_DATALINE_REQTYPE	必填	int	类型，资讯类型为：GameAppOperation. QQFAV_DATALINE_TYPE_DEFAULT
GameAppOperation.QQFAV_DATALINE_DESCRIPTION	可选	String	音乐的描述信息。
GameAppOperation.QQFAV_DATALINE_IMAGEURL	必填	String	预览图地址：可以为本地图片地址或网络图片地址
GameAppOperation.QQFAV_DATALINE_URL	必填	String	详情页地址

1.27 分享信息到 QQ 群部落（无需 QQ 登录）

分享信息到 QQ 群部落接口，可将文本、本地图片等信息分享到 QQ 群部落。Tencent 类的 shareToTroopBar 函数可直接调用，不需要用户授权（使用手机 QQ 当前的登录态）。调用该接口会唤起手机 QQ 发送至群部落界面。该接口仅支持手机 QQ5.3 及以上版本。

调用分享信息到 QQ 群部落的示例代码如下：

```
final Bundle params = new Bundle();
params.putString(GameAppOperation.QQFAV_DATALINE_APPNAME, appName.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_TITLE, title.getText().toString());
params.putString(GameAppOperation.QQFAV_DATALINE_DESCRIPTION, description.getText().toString());
params.putString(GameAppOperation.TROOPBAR_ID, troopbarId.getText().toString());
```

```
params.putStringArrayList(GameAppOperation.QQFAV_DATALINE_FILEDATA, fileDataList);
mTencent.shareToTroopBar(activity, params, new UiListener());
```

调用分享信息到 QQ 群部落接口的 params 参数说明如下：

参数	是否必传	类型	参数说明
GameAppOperation.QQFAV_DATALINE_APPNAME	选填	String	应用的名称，SDK 会通过传入的 activity 得到应用的名称，如果通过 activity 得不到应用名称，则会使用传入的这个 AppName。
GameAppOperation.QQFAV_DATALINE_TITLE	必填	String	信息的标题。(字数限制在 4~25 字之间)
GameAppOperation.QQFAV_DATALINE_DESCRIPTION	必填	String	信息的正文内容。(字数限制在 10~700 字之间)
GameAppOperation.TROOPBAR_ID	选填	String	部落 ID，一般为纯数字。
GameAppOperation.QQFAV_DATALINE_FILEDATA	选填	ArrayList<String>	配图地址信息，只支持本地图片绝对路径地址,最多可填 9 张图片。

1.28 Server-Side 登录模式

通过调用 Tencent 类的 loginServerSide 函数发起 Server-Side 模式登录。

调用该接口，会启动一个交互界面，完成用户登录和授权的交互流程，通过回调得到 server 返回的 code 和 openId。其中 code 的值保存在 access_code 字段里面。

当安装了手机 QQ 时，SDK 会启用手机 QQ 的特定 Activity，通过此 Activity 完成登录和授权功能。 当没有找到此 Activity 时，SDK 会执行 Oauth2.0 的 User-Agent 流程，即显示一个包含 WebView 的对话框， 通过加载登录授权网页来完成登录和授权的交互流程。

调用 Server-Side 模式登录接口的示例代码如下：

```
private void doLogin() {
    IUiListener listener = new BaseUiListener() {
        @Override
        protected void doComplete(JSONObject values) {
            updateLoginButton();
        }
    };
    mTencent.loginServerSide(this, SCOPE, listener);
}
```

调用登录接口的参数说明如下：

参数	参数说明
activity	调用者 activity。应用使用 SDK 时，会从应用自己的 Activity 跳转到 SDK 的 Activity，应用调用 SDK 的 Activity 即为这里的调用者 activity。
scope	应用需要获得哪些接口的权限，由“，”分隔。例如： SCOPE = “get_simple_userinfo,add_topic”；所有权限用“all”
listener	回调接口，IUiListener 实例。

注：在某些低端机上调用登录后，由于内存紧张导致 APP 被系统回收，登录成功后无法成功回传数据。

解决办法如下

在调用 login 的 Activity 或者 Fragment 重写 onActivityResult 方法，示例代码如下：

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    Tencent.onActivityResultData(requestCode,resultCode,data,loginListener);
    super.onActivityResult(requestCode, resultCode, data);
}
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    If (requestCode == Constants.REQUEST_LOGIN) {
        Tencent.onActivityResultData(requestCode,resultCode,data,loginListener);
    }
    super.onActivityResult(requestCode, resultCode, data);
}
```

2. WPA 接口

对于 WPA 用户，无需加其为好友就能和其进行会话，这里我们提供 2 个接口,一个是获取 WPA 用户的在线状态，一个是发起会话.

2.1 查询 WPA 用户在线状态

```
mTencent.getWPAUserOnlineState(uin, new BaseUIListener());
```

参数 uin 是对方的 QQ 号码。

在 onComplete 里返回的 response 是个 String，如果在线是{"online":1}，如果不在线是{"online":0}

2.2 发起 WPA 会话

发起会话:

```
int ret = Tencent.startWPAConversation(WPAActivity.this, uin, "");
```

参数 uin 是对方的 QQ 号码。

ret 为 0 表示打开了手 Q 的会话窗口. 其他则为错误.

2.3 发起群会话

发起群会话:

```
int ret = Tencent.startWPAConversation(WPAActivity.this, WPA.CHAT_TYPE_GROUP, uin, "");
```

ret 为 0 表示打开了手 Q 的群会话窗口. 其他则为错误.

参数	参数说明
activity	调用者 activity。应用使用 SDK 时，会从应用自己的 Activity 跳转到 SDK 的 Activity，应用调用 SDK 的 Activity 即为这里的调用者 activity。
chatType	会话类型，群会话为：WPA.CHAT_TYPE_GROUP
text	附加消息

3. OpenApi 接口调用说明

3.1 调用说明

上面介绍的功能接口，都是需要 SDK 提供 UI 交互来完成后续操作，在 SDK 中，还有一些功能不需要 SDK 提供 UI 交互的 OpenApi 接口，如获取用户信息、获取用户相册列表、发送分享（addshare）、发表说说、上传图片、创建相册等，更多功能接口请查看《API 列表》。

这些功能统一通过调用 Tencent 类的 request 或 requestAsync 方法来实现。request 和 requestAsync 这两个接口的功能相同，区别是一个是同步调用，一个是异步调用。

- request(String graphPath, Bundle params, String httpMethod)　　发送同步调用请求访问腾讯提供的 OpenAPI。
- requestAsync(String graphPath, Bundle params, String httpMethod, IRequestListener listener, Object state)　发送异步调用请求访问腾讯提供的 OpenAPI。

参数说明如下：

参数	参数说明
graphPath	要调用的接口名称，通过 SDK 中的 Constant 类获取宏定义。
params	以 K-V 组合的字符串参数。Params 是一个 Bundle 类型的参数，里面以键值对（Key-value）的形式存储数据，应用传入的邀请分享等参数就是通过这种方式传递给 SDK，然后由 SDK 发送到后台。
httpMethod	使用的 http 方式，如 SocialConstants.HTTP_GET，SocialConstants.HTTP_POST。
listener	回调接口，IUiListener 实例。
state	状态对象，将在回调时原样传回给 listener，供应用识别异步调用。SDK 内部不访问该对象。

其中的 graphPath 参数用于区别不同的功能接口。

详细的参数值与功能接口的对应，请参照下面的表格：

参数值	接口说明
SocialConstants.GRAPH_SIMPLE_USER_INFO	获取用户信息，返回用户昵称、头像 URL、是否黄钻用户、黄钻等级等。
SocialConstants.GRAPH_ADD_ALBUM	创建一个 QQ 空间相册。
SocialConstants.GRAPH_LIST_ALBUM	获取用户 QQ 空间相册列表，返回用户相册列表以及每个相册的详细信息。
SocialConstants.GRAPH_ADD_SHARE	发送分享，不支持@好友功能。
SocialConstants.GRAPH_ADD_TOPIC	发表一条说说到 QQ 空间。
SocialConstants.GRAPH_UPLOAD_PIC	上传一张照片到 QQ 空间相册。

3.2 调用示例

这里以发表说说接口的调用实例，来说明通过 requestAsync 调用兼容接口的方法：

```
private void onClickAddTopic() {
    if (mTencent.isSessionValid() && mTencent.getOpenId() != null)  {
        Bundle params = new Bundle();
        params = new Bundle();
        params.putString("richtype", "2");// 发布心情时引用的信息的类型。1 表示图片；2 表示网页； 3 表示视频。
        params.putString("richval",
                ("http://www.qq.com" + "#" + System.currentTimeMillis()));// 发布心情时引用的信息的值。有 richtype 时必须有 richval。
        params.putString("con", "腾讯 QQ 登录测试：心情不错！");// 发布的心情的内容。
```

```
        params.putString("lbs_nm", "广东省深圳市南山区高新科技园腾讯大厦");// 用户发布心情时的所在地。

        params.putString("lbs_x", "+90");// 经度。-180.0 到+180.0，+表示东经。

        params.putString("lbs_y", "-60");// 纬度。纬度。-90.0 到+90.0，+表示北纬。


        mTencent.requestAsync(SocialConstants.GRAPH_ADD_TOPIC, params,
                        SocialConstants.HTTP_POST, new BaseApiListener(), null);
        mProgressDialog.show();
    }
}
```

在上面的调用中，传入了 BaseApiListener 的实例，此实例的示例代码在《Andriod_SDK 使用说明》的“实现回调”一节中已说明。

4. 开发者作为内容提供者给手机 QZone 提供内容的接口

这是针对开发者作为数据提供的应用程序，如果要使用互联 SDK 来作为数据内容的提供者，比如提供图片或者视频，需要按以下步骤来实现：

1. 需要一个 Activity 来接受 QQ 应用的调用，对这个 Activity，对应的 AndroidManifest.xml 的配置如下：

```
<activity android:name="com.sample.DataProviderActivity">
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:scheme="tencent222222.data" />
    </intent-filter>
</activity>
```

这里 scheme 的格式是"tencent" + appid + ".data"。

2. 在 DataProviderActivity 的 onCreate 函数里，必须添加如下代码：

```
public class DataProviderActivity extends Activity {
    private CallbackManager mCalledManager;
    ...
```

```
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        mCalledManager = new CallbackManager(this); //构建 CallbackManager 对象

       ...
    }
...
```

3. 如果要将生成的图片，视频内容发回到 QQ 的应用里，按照下方的代码来实现：

```
public void onClick(View v) {
    switch (v.getId()) {

    case R.id.sendimagepathbth: //发送文本和图片路径


        int errorCode = mCalledManager.sendTextAndImagePath ("图片说明文本",

            "/sdcard/DCIM/Camera/101020.jpg");
        If (errorCode == 0) {
            finish();
        } else {
            ...
        }
        break;
        case R.id.sendvideopathbtn:

            int errorCode = mCalledManager.sendTextAndVideoPath("视频说明文本",

                "/sdcard/DCIM/Camera/101021.mp4");
            If (errorCode == 0) {
                finish();
            } else {
                ...
            }
            break;
        case R.id.sendtextonlybtn:

            int errorCode = mCalledManager.sendTextOnly("文本内容");

            If (errorCode == 0) {
                finish();
            } else {
                ...
            }
            break;
    }
```

这里支持 3 种内容格式：1 是文本和图片路径；2 是文本和视频路径；3 是纯文本。

路径支持本地路径和 url 地址，本地路径必须是在 sd 卡上的路径。

对应的错误代码有以下值：

```
public final class ErrorCode {
	public static final int Success = 0;
	public static final int NotSupportThisDataType = -1;
	public static final int NotFromTencentApp = -2;
	public static final int NotFoundReturnActivity = -3;
	public static final int NotFoundTargetApp = -4;
	public static final int FileNotInSdCard = -5;
	public static final int FileSizeTooLarge = -6;
	public static final int PathIsNull = -7;
	public static final int FileNotExist = -8;
	public static final int FileIsEmpty = -9;
	public static final int SdCardNotExist = -10;
}
```

4. 第三方应用可以检查 QQ 调用方请求的数据类型。

```
mCalledManager.isSupportType(DataType.TEXT_AND_IMAGE_PATH)
```

如果返回 true 则表示支持，否则不支持。这里 DataType 的定义如下：

```
public final class DataType {

		public static final int TEXT_AND_IMAGE_PATH = 0x1; //文本和图片路径

		public static final int TEXT_AND_VIDEO_PATH = 0x2; //文本和视频路径

		public static final int TEXT_ONLY    = 0x4;                    //纯文本
}
```

5. 在调用下面这三个方法后，如果返回值是 0， 建议 finish 掉 Activity。

mCalledManager.sendTextAndImagePath(...)

mCalledManager.sendTextAndVideoPath(...)

mCalledManager.sendTextOnly(...)

5. 返回码说明

使用 SDK 时，所有结果都会通过回调返回给应用。在回调的结果中，会包含每次调用结果的返回码。正常情况下返回码为 0，表示调用成功。如果返回码不为 0，说明调用出错，需要根据返回码的值来定位错误原因。

6. 登录常见错误码信息

110201：未登陆

110405：登录请求被限制

110404：请求参数缺少 appid

110401：请求的应用不存在

110407：应用已经下架

110406：应用没有通过审核

100044：错误的 sign

110500：获取用户授权信息失败

110501：获取应用的授权信息失败

110502：设置用户授权失败

110503：获取 token 失败

110504：系统内部错误