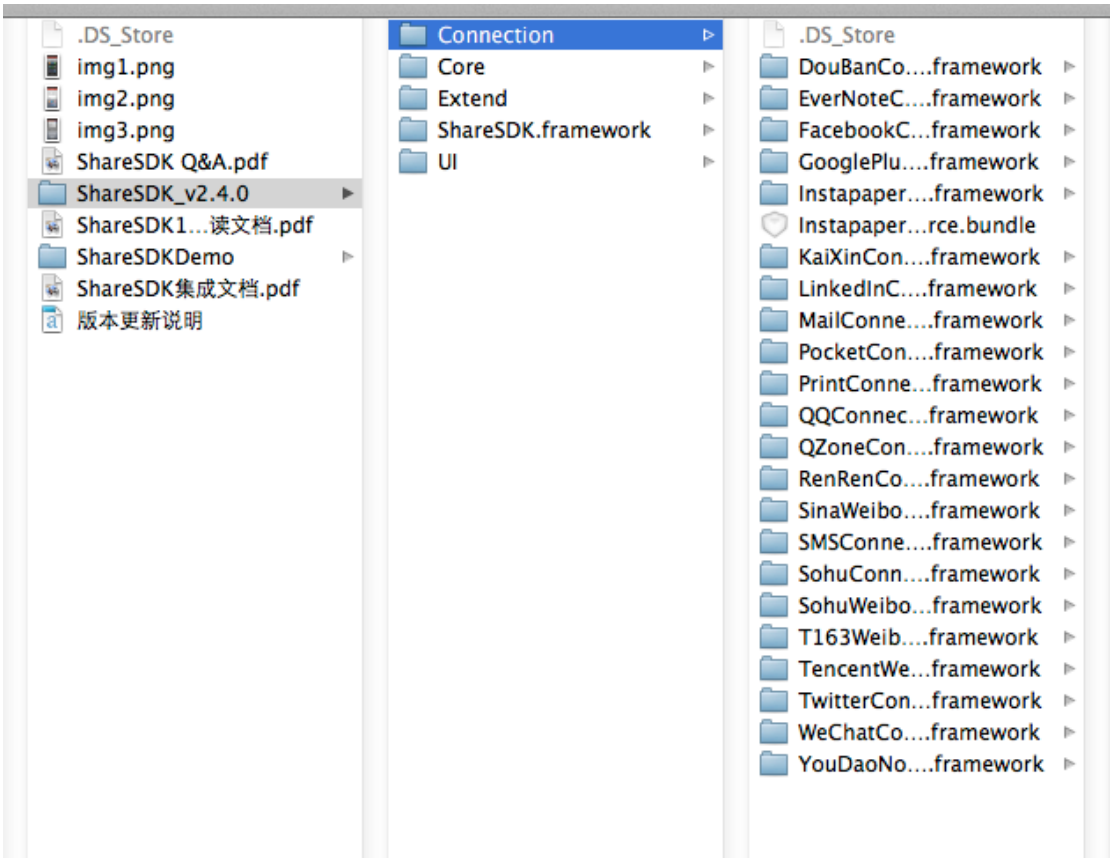
ShareSDK 集成指南

一、SDK 目录说明



Connection	包含所有分享平台框架（framework）。可以根据自己的需要删除不需要集成的平台。	
	DouBanConnection.framework	豆瓣分享平台框架
	FacebookConnection.framework	Facebook 分享平台框架。
	InstapaperConnection.framework	Instapaper 分享平台框架。
	InstapaperResource.bundle	Instapaper 资源包
	KaiXinConnection.framework	开心网分享平台框架。
	MailConnection.framework	邮件分享框架。
	PrintConnection.framework	打印分享框架。
	QQConnection.framework	QQ 好友分享框架
	QZoneConnection.framework	QQ 空间分享平台框架。
	RenRenConnection.framework	人人网分享框架。
	SinaWeiboConnection.framework	新浪微博分享框架
	SMSConnection.framework	短信分享框架。
	SohuConnection.framework	搜狐分享框架，用于搜狐随

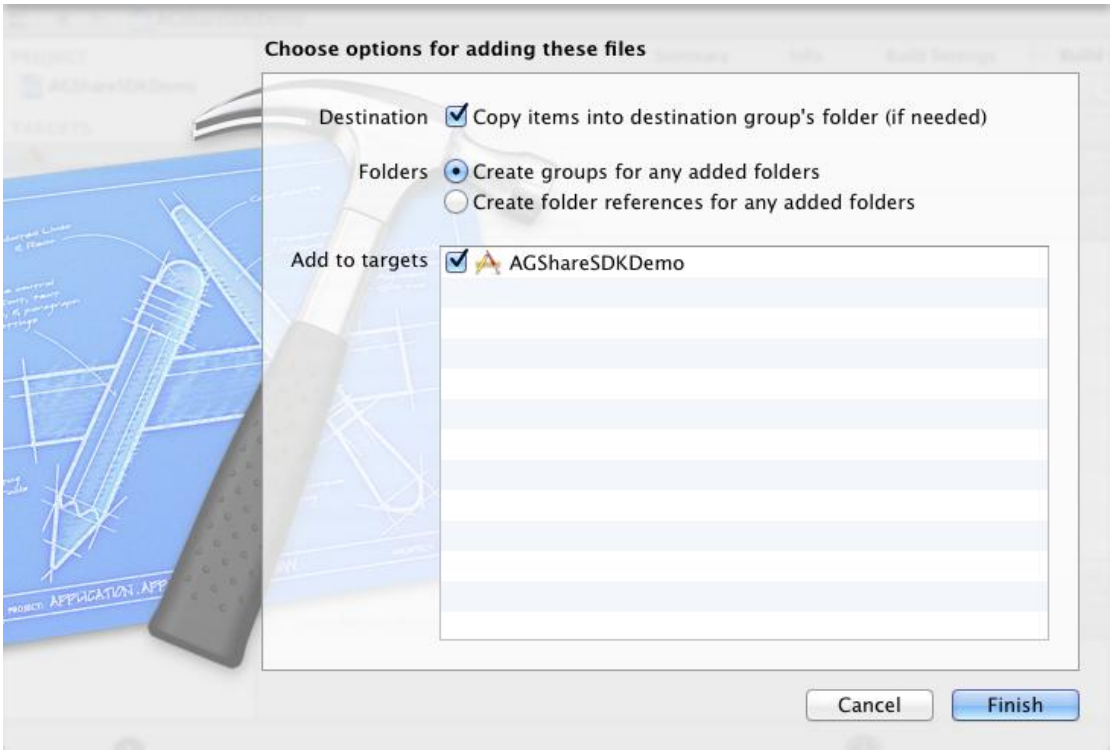
		身看。
	SohuWeiboConnection.framework	搜狐微博分享框架
	T163WeiboConnection.framework	网易微博分享框架
	TencentWeiboConnection.framework	腾讯微博分享框架
	TwitterConnection.framework	Twitter 分享框架
	WeChatConnection.framework	微信分享框架
	YouDaoNoteConnection.framework	有道云笔记分享框架
	PocketConnection.framework	Pocket 分享框架
	EverNoteConnection.framework	印象笔记分享框架
	GooglePlusConnection.framework	Google+分享框架
	LinkedInConnection.framework	LinkedIn 分享框架
Core	SDK 核心框架目录，包括了 SDK 的核心服务。所有的 Connection 必须依赖此目录中的框架（framework）。	
	AGCommon.framework	通用工具类框架。
	ShareSDKLocalizable.strings	本地化资源文件。
	Resource.bundle	通用资源包
	ShareSDKCoreService.framework	SDK 核心服务包
Extend	外部 SDK 目录，用于存在第三方 SDK，目前包括微信、QQ 好友分享、QQ 空间 SSO 以及腾讯微博 SSO 四个平台的 SDK。可以根据需要删除不集成的 SDK 包。	
	TencentOpenAPI.framework	QQ 空间 SSO 授权以及 QQ 好友分享框架。QQConnection.framework 和 QZoneConnection.framework 的 SSO 功能需要依赖此库进行分享。
	SDKExport	微信 SDK 开发包。WeChatConnection.framework 需要依赖此库进行分享。
	TencentOpenApi_IOS_Bundle.bundle	TencentOpenApi 框架所要使用的资源包。
	TencentWeiboSSO	腾讯微博的 SSO 开发包。TencentWeiboConnection.framework 需要依赖此库进行 SSO 登陆。
	RenRenSDK	人人网的 SDK 开发包，RenRenConnection.framework 的 SSO 功能需要依赖此库进行分享。
	libWeiboSDK	新浪微博 SDK 开发包，SinaWeiboConnection.framework 的客户端分享需要依赖次库进行分享

	GooglePlusSDK	Google+的 SDK 开发包，GooglePlusConnection.framework 需要依赖此库
ShareSDK.framework	SDK 的快捷分享框架，此框架提供了一组实用功能接口，使用此框架可以进行快速集成。	
UI	SDK 的内置 UI 框架目录，其命名格式为"ShareSDK + 设备类型 + 视图样式 + 视图类型"。尚未使用的 UI 框架及其资源包可以删除。	
	ShareSDKiPadDefaultShareViewUI.bundle	iPad 版默认分享视图资源包。
	ShareSDKiPadDefaultShareViewUI.framework	iPad 版默认分享视图框架。
	ShareSDKiPadSimpleShareViewUI.bundle	iPad 版简单分享视图资源包。
	ShareSDKiPadSimpleShareViewUI.framework	iPad 版简单分享视图框架。
	ShareSDKiPhoneAppRecommendShareViewUI.framework	iPhone 版应用推荐视图框架。
	ShareSDKiPhoneDefaultShareViewUI.bundle	iPhone 版默认分享视图资源包。
	ShareSDKiPhoneDefaultShareViewUI.framework	iPhone 版默认分享视图框架。
	ShareSDKiPhoneSimpleShareViewUI.bundle	iPhone 版简单分享视图资源包
	ShareSDKiPhoneSimpleShareViewUI.framework	iPhone 版简单分享视图框架。

二、SDK 初始化

1．导入 SDK

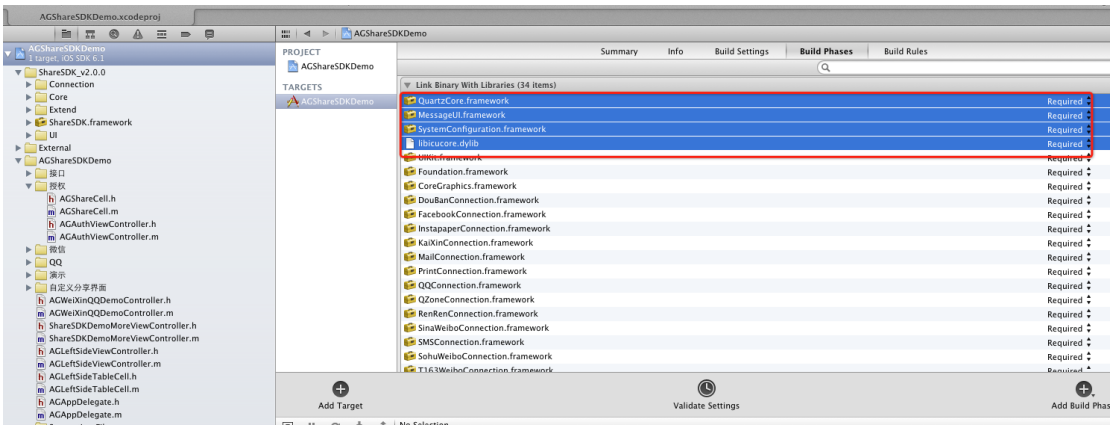
将 ShareSDK 文件夹到项目文件夹中夹并拖入项目中。



2. 添加依赖框架(Framework)

打开项目设置中的 Build Phases 页，在“Link Binary With Libraries”一栏中，点击左下角的“＋”号；在弹出窗口里面分别以下库加入到项目中：

SystemConfiguration.framework
QuartzCore.framework
MessageUI.framework (如果不集成邮件和短信可以不添加)
libicucore.dylib
libz.1.2.5.dylib
libstdc++.dylib (如果不需要 QQ 空间 SSO 登录可以不添加)
libsqlite3.dylib (如果不需要 QQ 空间 SSO 登录可以不添加)
CoreTelephony.framework (如果不需要 QQ 空间 SSO 登录可以不添加)
Security.framework (如果不使用 Google+可以不添加)



3. 导入头文件 ShareSDK.h 并初始化社交平台的 App 信息。

打开*AppDelegate.m(*代表你的工程名字)　导入文件头 ShareSDK.h：

```
#import <ShareSDK/ShareSDK.h>
```

首先要注意的是应用相关信息需要到各个平台进行注册申请（参考：分享平台注册申请应用地址对照表），如果不需要使用到的平台可以不进行设置：

初始化 SDK 有两种方式：第一种是采用本地平台配置信息初始化，第二种是采用服务器中托管的平台配置信息初始化。两种初始化方法中都需要调用 registerApp 来初始化 SDK，其中 appKey 参数是需要在 ShareSDK 官网中可以注册申请的（建议为每个应用申请一个 appKey，如果多个应用用同一个 key 会导致不同应用的统计数据混合在一起）。下面将分别讲述两种方式。

3.1 本地配置信息初始化

在- (BOOL)application: didFinishLaunchingWithOptions:方法中添加如下语句对 SDK 进行初始化并为各个分享平台的应用信息进行设置.

```
- (void)initializePlat
{
    //添加新浪微博应用

    [ShareSDK connectSinaWeiboWithAppKey:@"3201194191"
                                appSecret:@"0334252914651e8f76bad63337b3b78f"
                                redirectUri:@"http://appgo.cn"];


    //添加腾讯微博应用

    [ShareSDK connectTencentWeiboWithAppKey:@"801307650"
                                    appSecret:@"ae36f4ee3946e1cbb98d6965b0b2ff5c"
                                    redirectUri:@"http://www.sharesdk.cn"];


    //添加QQ空间应用

    [ShareSDK connectQZoneWithAppKey:@"100371282"
                            appSecret:@"aed9b0303e3ed1e27bae87c33761161d"];


    //添加网易微博应用

    [ShareSDK connect163WeiboWithAppKey:@"T5EI7BXe13vfyDuy"
                                appSecret:@"gZxwyNOvjFYpxwwlnuizHRRtBRZ2lV1j"
                                redirectUri:@"http://www.shareSDK.cn"];
```

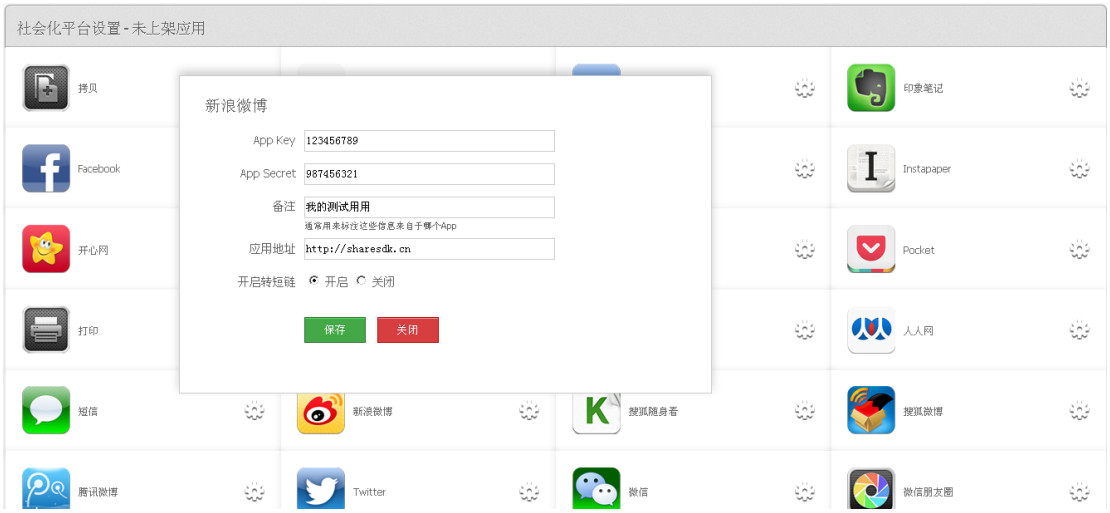
```
    //添加搜狐微博应用

    [ShareSDK connectSohuWeiboWithConsumerKey:@"SAfmTG1blxZY3HztESWx"
                               consumerSecret:@"yfTZf)!rVwh*3dqQuVJVsUL37!F)!yS9S!Orcsij"
                                  redirectUri:@"http://www.sharesdk.cn"];


    //添加豆瓣应用

    [ShareSDK connectDoubanWithAppKey:@"07d08fbfc1210e931771af3f43632bb9"
                            appSecret:@"e32896161e72be91"
                          redirectUri:@"http://dev.kumoway.com/braininference/infos.php"];


    //添加人人网应用

    [ShareSDK connectRenRenWithAppKey:@"fc5b8aed373c4c27a05b712acba0f8c3"
                            appSecret:@"f29df781abdd4f49beca5a2194676ca4"];


    //添加开心网应用

    [ShareSDK connectKaiXinWithAppKey:@"358443394194887cee81ff5890870c7c"
                            appSecret:@"da32179d859c016169f66d90b6db2a23"
                          redirectUri:@"http://www.sharesdk.cn/"];


    //添加Instapaper应用

    [ShareSDK
connectInstapaperWithAppKey:@"4rDJORmcOcSAZL1YpqGHRI605xUvrLbOhkJ07yO0wWrYrc61FA"
                          appSecret:@"GNr1GespOQbrm8nvd7rlUsyRQsIo3bolbMguAl9gfpdL0aKZWe"];


    //添加有道云笔记应用

    [ShareSDK connectYouDaoNoteWithConsumerKey:@"dcde25dca105bcc36884ed4534dab940"
                                consumerSecret:@"d98217b4020e7f1874263795f44838fe"
                                   redirectUri:@"http://www.sharesdk.cn/"];


    //添加Facebook应用

    [ShareSDK connectFacebookWithAppKey:@"107704292745179"
                              appSecret:@"38053202e1a5fe26c80c753071f0b573"];


    //添加Twitter应用

    [ShareSDK connectTwitterWithConsumerKey:@"mnTGqtXk0TYMXYTN7qUxg"
```

```
                consumerSecret:@"ROkFqr8c3m1HXqS3rm3TJ0WkAJuwBOSaWhPbZ9Ojuc"
                redirectUri:@"http://www.sharesdk.cn"];

    //添加搜狐随身看应用

    [ShareSDK connectSohuKanWithAppKey:@"e16680a815134504b746c86e08a19db0"
                             appSecret:@"b8eec53707c3976efc91614dd16ef81c"
                             redirectUri:@"http://sharesdk.cn"];

    //添加Pocket应用

    [ShareSDK connectPocketWithConsumerKey:@"11496-de7c8c5eb25b2c9fcdc2b627"
                                 redirectUri:@"pocketapp1234"];

    //添加印象笔记应用

    [ShareSDK connectEvernoteWithType:SSEverNoteTypeSandbox
                          consumerKey:@"sharesdk-7807"
                          consumerSecret:@"d05bf86993836004"];

    //添加LinkedIn应用

    [ShareSDK connectLinkedInWithApiKey:@"ejo5ibkye3vo"
                              secretKey:@"cC7B2jpxITqPLZ5M"
                              redirectUri:@"http://sharesdk.cn"];
}

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary
*)launchOptions
{
    [ShareSDK registerApp:@"520520test"];
    [self initializePlat];

    ……

    return YES;
}
```

注：初始化平台中某些平台需要传入 redirectUri 参数，则参数为回调地址，必须在申请 App 的平台上进行回调地址设置后，将此地址传入参数中。否则会导致授权页显示出错。其中腾讯微博的回调地址为 app 信息中的“应用网址”

3.2 服务器托管模式初始化

在申请相关分享平台应用信息后登录 ShareSDK 官网，在申请的应用中把得到的应用相关信息设置到对应的平台中,如下图所示：



然后在- (BOOL)application: didFinishLaunchingWithOptions:方法中添加如下语句对 SDK 进行初始化。

```
- (void)initializePlatForTrusteeship
{
    //导入QQ互联和QQ好友分享需要的外部库类型，如果不需要QQ空间SSO和QQ好友分享可以不调用
此方法
    [ShareSDK importQQClass:[QQApiInterface class]
                tencentOAuthCls:[TencentOAuth class]];

    //导入人人网需要的外部库类型,如果不需要人人网SSO可以不调用此方法
    [ShareSDK importRenRenClass:[RennClient class]];

    //导入腾讯微博需要的外部库类型，如果不需要腾讯微博SSO可以不调用此方法
    [ShareSDK importTencentWeiboClass:[WBApi class]];

    //导入微信需要的外部库类型，如果不需要微信分享可以不调用此方法
    [ShareSDK importWeChatClass:[WXApi class]];
}

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary
*)launchOptions
```

```
{
    [ShareSDK registerApp:@"api20" useAppTrusteeship:YES];
    [self initializePlatForTrusteeship];


    ……


    return YES;
}
```

注：registerApp 方法中的 useAppTrusteeship 一定要为 YES。如果为 NO 则表示不使用托管模式，将按照本地配置方式进行初始化。

4. 为 keyWindow 设置 rootViewController 属性

请确认 didFinishLaunchingWithOptions 方法中是否有设置 window 对象的 rootViewController 属性。否则会导致无法弹出授权界面或者分享界面点击后无反映，卡死现象等。

设置类似于下面代码所示：

```
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary
*)launchOptions
{

    ……


    self.window = [[[UIWindow alloc] initWithFrame:[[UIScreen mainScreen] bounds]] autorelease];

    self.viewController = [[[UINavigationController alloc] init] autorelease];

    self.window.rootViewController  = self.viewController;
    [self.window makeKeyAndVisible];

    return YES;
}
```

注：如果您创建应用时使用 storyBoard 可以省略此步骤，系统以自动为您设置了 rootViewController。

5. 加入 handleOpenURL 处理方法

在使用 SSO 授权方式（即跳转到相应客户端进行授权的方式）或者集成微信、QQ 好友分享时，需要在处理请求 URL 的委托方法中加入 ShareSDK 的处理方法（没有以上说明情况可省略此步骤），在*AppDelegate.m(*代表你的工程名字)文

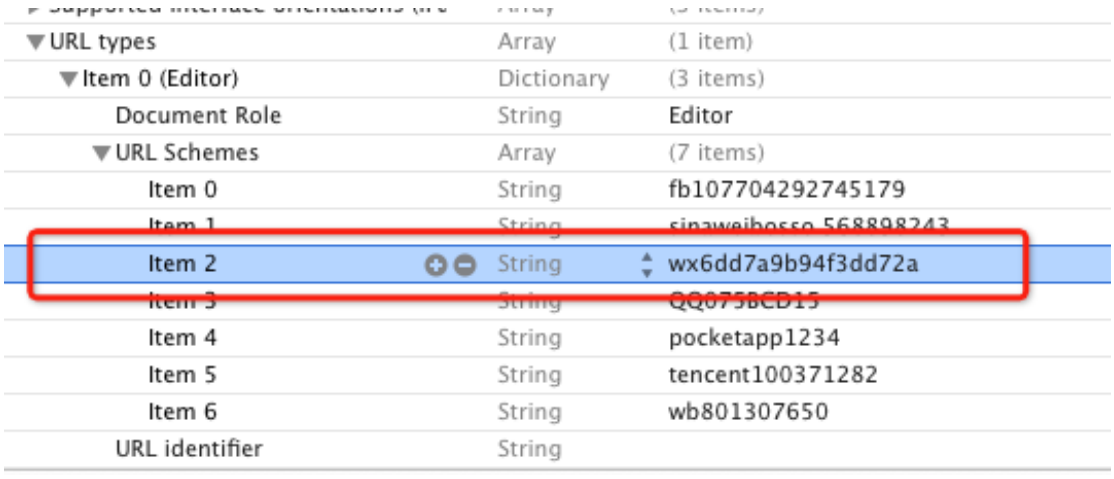
件中加入如下方法：

```
- (BOOL)application:(UIApplication *)application
          handleOpenURL:(NSURL *)url
{
      return [ShareSDK handleOpenURL:url
                                wxDelegate:self];
}

- (BOOL)application:(UIApplication *)application
              openURL:(NSURL *)url
              sourceApplication:(NSString *)sourceApplication
              annotation:(id)annotation
{
      return [ShareSDK handleOpenURL:url
                            sourceApplication:sourceApplication
                            annotation:annotation
                            wxDelegate:self];
}
```

6. 添加微信社交平台（不需要微信分享可略过此步骤）

登录微信开放平台（**http://open.weixin.qq.com/**）注册应用并取得应用的 AppID，

打开*-Info.plist（*代表你的工程名字）。找到 URL types 配置项（如果没有则新增），展开 URL types – URL Schemes，在 URL Schemes 下分别各新增一项用于微信的 Scheme（如果不添加则会导致法返回应用）。将申请的 appId 填入此配置项中。如图所示：



注：必须执行步骤 5 中的操作。

然后打开工程中的*AppDelegate.m(*代表你的工程名字) 文件，导入微信 SDK 的头文件：

```
#import "WXApi.h"
```

- **本地配置信息方式初始化时**

在- application: didFinishLaunchingWithOptions: 方法内添加如下语句：

```
//添加微信应用

[ShareSDK connectWeChatWithAppId:@"wx6dd7a9b94f3dd72a"
                       wechatCls:[WXApi class]];
```

上述代码中的 appId 参数为申请的微信 AppID。

- **服务器托管模式初始化时**

确认在初始化 SDK 后是否有调用 importWeChatClass 方法，如果没有则添加如下语句

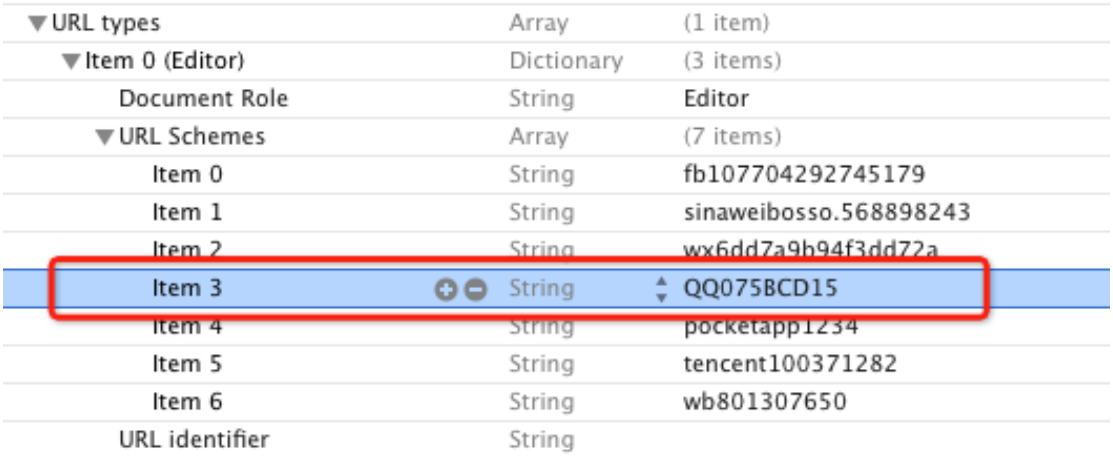
```
//导入微信需要的外部库类型，如果不需要微信分享可以不调用此方法

[ShareSDK importWeChatClass:[WXApi class]];
```

7. 添加 QQ 社交平台（不需要 QQ 好友分享可略过此步骤）

7.1 使用 QQ 互联申请的 AppID 初始化

先登录 QQ 互联（http://connect.qq.com/）注册成为开发者并登记应用取得 AppId（如果配置 QQ 空间时已申请应用 Id 可以不需要重复申请，直接使用 QQ 空间中申请的 Id）。

打开*-Info.plist（*代表你的工程名字）。找到 URL types 配置项（如果没有则新增），展开 URL types – URL Schemes，在 URL Schemes 下分别各新增一项用于 QQ 的 Scheme（如果不添加则会导致法返回应用）。以“QQ” ＋ appId 的 16 进制形式填入此配置项中（如果 appId 转换的 16 进制数不够 8 位则在前面补 0，如转换的是：5FB8B52，则最终填入为：QQ0 5FB8B52）。如图所示：



注：必须执行步骤 5 中的操作。

然后打开工程中的*AppDelegate.m(*代表你的工程名字) 文件，导入 QQSDK 的头文件：

```
#import <TencentOpenAPI/QQApiInterface.h>
#import <TencentOpenAPI/TencentOAuth.h>
```

- **本地配置信息方式初始化时**

在- (BOOL)application: didFinishLaunchingWithOptions:方法内添加如下语句：

```
//添加QQ应用

 [ShareSDK connectQQWithQZoneAppKey:@"100371282"
                            qqApiInterfaceCls:[QQApiInterface class]
                              tencentOAuthCls:[TencentOAuth class]];
```

- **服务器托管模式初始化时**

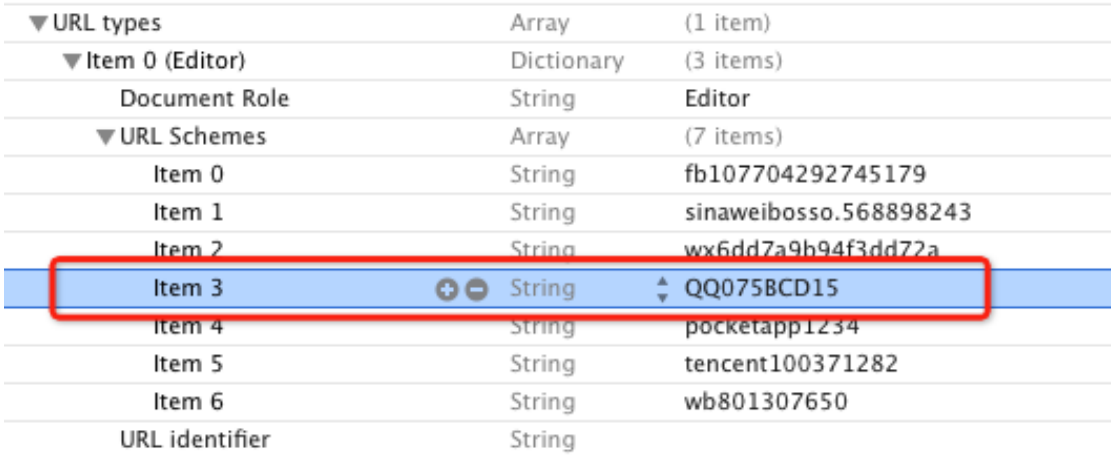
确认在初始化 SDK 后是否有调用 importQQClass 方法，如果没有则添加如下语句：

```
//导入QQ互联和QQ好友分享需要的外部库类型，如果不需要QQ空间SSO和QQ好友分享可以不调用此方

法

 [ShareSDK importQQClass:[QQApiInterface class]
                 tencentOAuthCls:[TencentOAuth class]];
```

7.2 使用手机 QQ 开发者平台申请的 AppID 初始化

如果申请的 appID 是在手机 QQ 开发者平台（http://mobile.qq.com/api/）中登记的，可以通过以下步骤进行集成：

打开*-Info.plist（*代表你的工程名字）。找到 URL types 配置项（如果没有则新增），展开 URL types – URL Schemes，在 URL Schemes 下分别各新增一项用于 QQ 的 Scheme（如果不添加则会导致法返回应用）。将申请的 appId 填入此配置项中。如图所示：



注：必须执行步骤 5 中的操作。

然后打开工程中的*AppDelegate.m(*代表你的工程名字) 文件，导入 QQSDK 的头文件：

```
#import<QQApi/QQApi.h>
```

注：如果你使用 QQ 空间 SSO 授权方式，那么不能引入以上头文件，否则会导致名称空间冲突。应修改为导入以下头文件：

```
#import<TencentOpenAPI/QQApi.h>
```

- **本地配置信息方式初始化时**

在- (BOOL)application: didFinishLaunchingWithOptions:方法内添加如下语句：

```
//添加QQ应用

 [ShareSDK connectQQWithAppId:@"QQ0F0A941E" qqApiCls:[QQApi class]];
```

- **服务器托管模式初始化时**

确认在初始化 SDK 后是否有调用 importQQClass 方法，如果没有则添加如下语句：

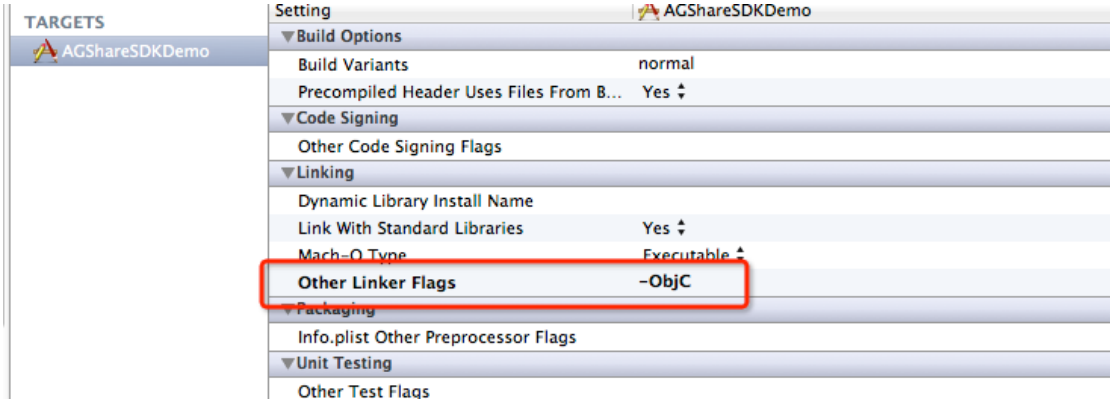
```
//导入QQ互联和QQ好友分享需要的外部库类型，如果不需要QQ空间SSO和QQ好友分享可以不调用此方法
 [ShareSDK importQQClass:[QQApi class]
         tencentOAuthCls:nil];
```

8. 添加 Google+社交平台（不需要 Google+分享可略过此步骤）

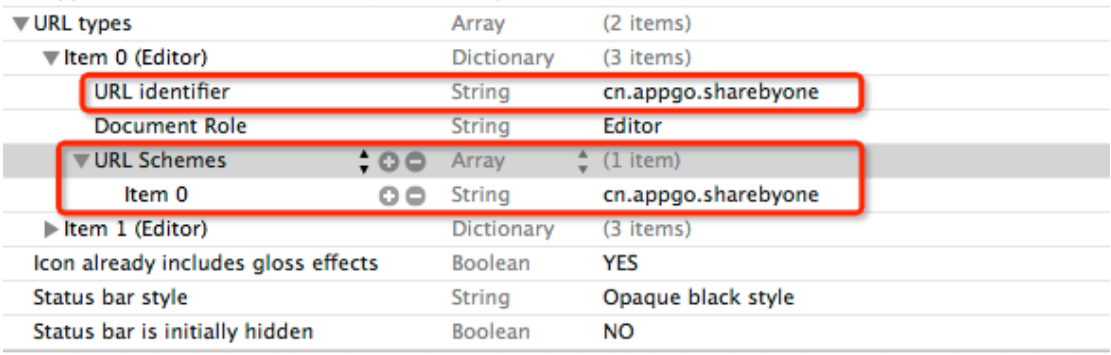
登录 Google+开发者平台（https://code.google.com/apis/console/）注册应用并取得应用的 ClientID。

然后确认 GoogleOpenSource.framework 和 GooglePlus.framework 是否已经加入到工程中，如果没有则从 ShareSDK 包中的 Extend 目录中把 GooglPlusSDK 目录拖入到工程中。

打开 Build Settings，在 Other linker flags 中加入-ObjC 标识。如图：



打开*-Info.plist（*代表你的工程名字）。找到 URL types 配置项（如果没有则新增），展开 URL types – URL identifier 中填入 Bundle ID，在 URL types –URL Schemes 下新增一项用于 Google+的 Scheme（如果不添加则会导致法返回应用）。将应用的 Bundle ID 填入此配置项中。如图所示：



注：必须执行步骤 5 中的操作。

然后打开工程中的*AppDelegate.m(*代表你的工程名字) 文件，导入Google+SDK 的头文件：

```
#import <GoogleOpenSource/GoogleOpenSource.h>
#import <GooglePlus/GooglePlus.h>
```

- **本地配置信息方式初始化时**

在- application: didFinishLaunchingWithOptions: 方法内添加如下语句：

```
//初始化Google+
[ShareSDK connectGooglePlusWithClientId:@"232554794995.apps.googleusercontent.com"
                            clientSecret:@"PEdFgtrMw97aCvf0joQj7EMk"
                            redirectUri:@"http://localhost"
                            signInCls:[GPPSignIn class]
                            shareCls:[GPPShare class]];
```

上述代码中的 clientId 参数为申请的 Google+的 ClientID。

- **服务器托管模式初始化时**

确认在初始化 SDK 后是否有调用 importGooglePlusClass 方法，如果没有则添加如下语句

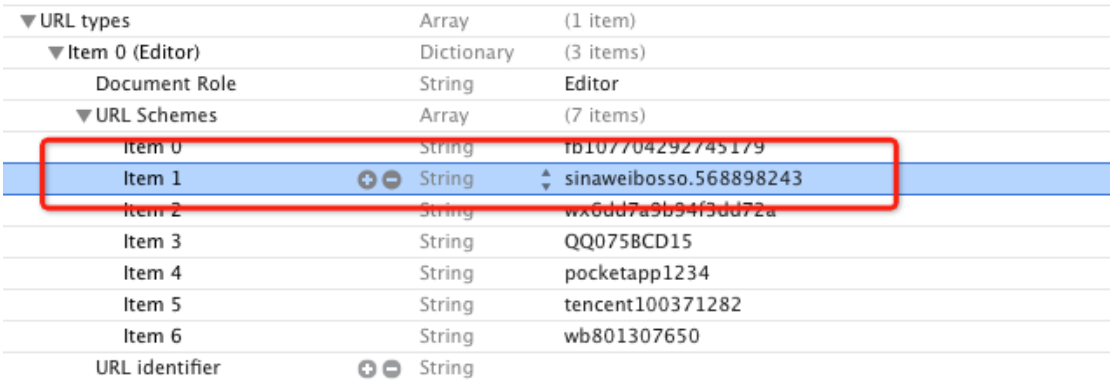
```
[ShareSDK importGooglePlusClass:[GPPSignIn class]
                                shareClass:[GPPShare class]];
```

9. 配置 SSO 授权（没有使用 SSO 授权可略过此步骤）

注：在执行该步骤时必须完成步骤 5 中的操作。

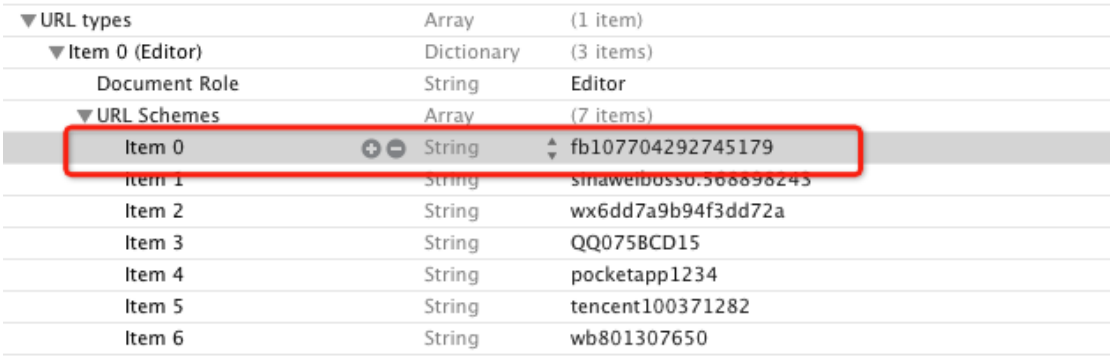
9.1 新浪微博

打开*-Info.plist（*代表你的工程名字）。找到 URL types 配置项（如果没有则新增），展开 URL types – URL Schemes，在 URL Schemes 下分别各新增一项用于新浪微博的 Scheme（如果不添加则会导致法返回应用）。其填写格式为：sinaweibosso.+appKey（你在新浪微博申请的 AppKey），如：sinaweibosso.2279784657。如图所示：



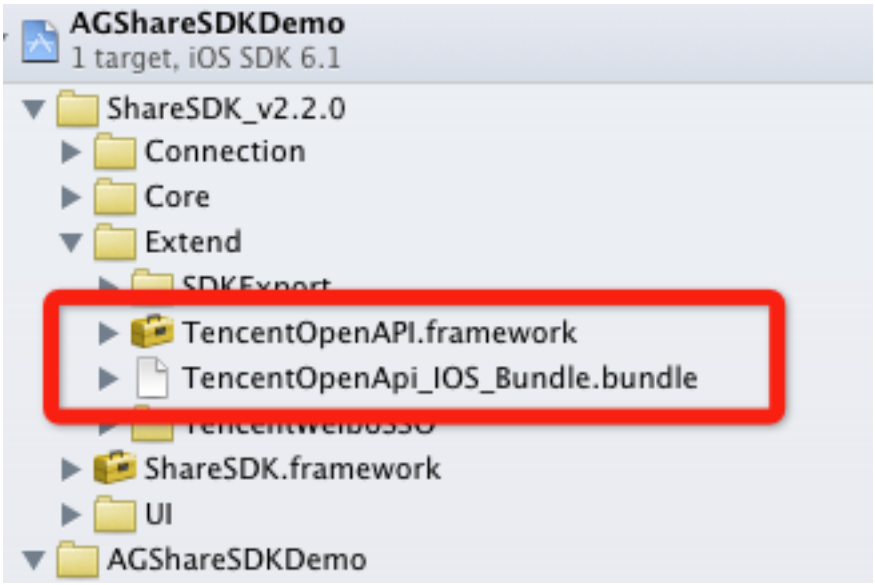
9.2 Facebook

打开*-Info.plist（*代表你的工程名字）。找到 URL types 配置项（如果没有则新增），展开 URL types – URL Schemes，在 URL Schemes 下分别各新增一项用于 Facebook 的 Scheme（如果不添加则会导致法返回应用）。其填写格式为：fb+appKey（你在 Facebook 申请的 AppKey），如：fb107704292745179。如图所示：

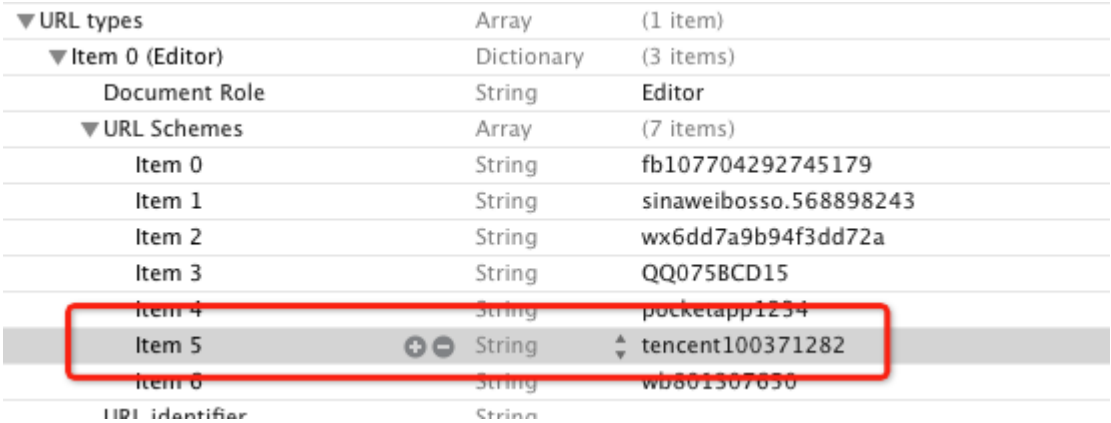


9.3 QQ 空间

先确认是否已把 TencentOpenAPI.framework 以及对应的资源包导入到项目中。



打开*-Info.plist（*代表你的工程名字）。找到 URL types 配置项（如果没有则新增），展开 URL types – URL Schemes，在 URL Schemes 下分别各新增一项用于 QQ 空间的 Scheme（如果不添加则会导致法返回应用）。其填写格式为：tencent+appID（你在 QQ 空间中申请的 AppId），如：tencent100371282。如图所示：



打开*AppDelegate.h(*代表你的工程名字) 文件，引入头文件：

```
#import<TencentOpenAPI/QQApiInterface.h>
#import<TencentOpenAPI/TencentOAuth.h>
```

- **本地配置信息方式初始化时**

打开*AppDelegate.m(*代表你的工程名字) 文件，修改初始化 QQ 空间的代码，由原来的

```
//添加QQ空间应用
```

```
[ShareSDK connectQZoneWithAppKey:@"100371282"
                                appSecret:@"aed9b0303e3ed1e27bae87c33761161d"];
```

改为

```
//添加QQ空间应用

[ShareSDK connectQZoneWithAppKey:@"100371282"
                                appSecret:@"aed9b0303e3ed1e27bae87c33761161d"
                        qqApiInterfaceCls:[QQApiInterface class]
                          tencentOAuthCls:[TencentOAuth class]];
```

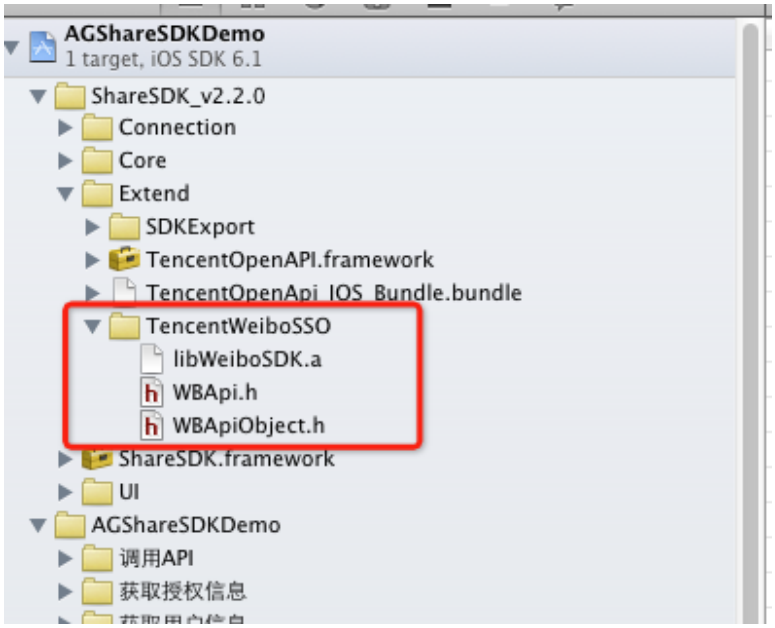
- **服务器托管模式初始化时**

确认在初始化 SDK 后是否有调用 importQQClass 方法，如果没有则添加如下语句：

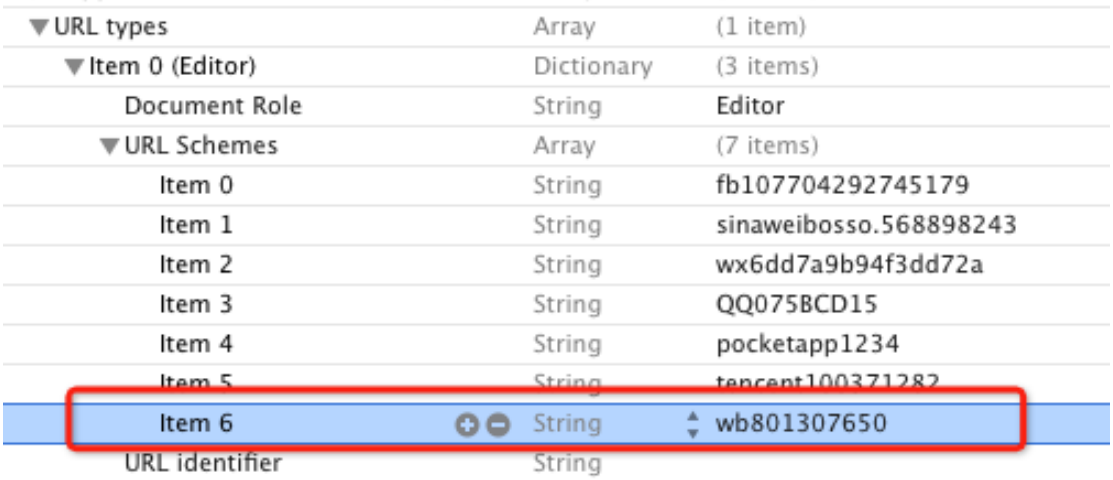
```
//导入QQ互联和QQ好友分享需要的外部库类型，如果不需要QQ空间SSO和QQ好友分享可以不调用此方法

[ShareSDK importQQClass:[QQApiInterface class]
                tencentOAuthCls:[TencentOAuth class]];
```

9.4 腾讯微博

先确认是否已把 libWeiboSDK.a 以及对应的头文件导入到项目中。



打开*-Info.plist（*代表你的工程名字）。找到 URL types 配置项（如果没有则新增），展开 URL types – URL Schemes，在 URL Schemes 下分别各新增一项用于腾讯微博的 Scheme（如果不添加则会导致法返回应用）。其填写格式为：wb + AppKey（你在腾讯微博中申请的 AppKey）， 如：wb801307650。如图所示：



然后打开*AppDelegate.h(*代表你的工程名字) 文件，引入头文件：

```
#import "WBApi.h"
```

- **本地配置信息方式初始化时**

打开*AppDelegate.m(*代表你的工程名字) 文件，修改初始化腾讯微博的代码，由原来的

```
//添加腾讯微博应用

[ShareSDK connectTencentWeiboWithAppKey:@"801307650"
                                 appSecret:@"ae36f4ee3946e1cbb98d6965b0b2ff5c"
                                 redirectUri:@"http://www.sharesdk.cn"];
```

改为

```
//添加腾讯微博应用

[ShareSDK connectTencentWeiboWithAppKey:@"801307650"
                                 appSecret:@"ae36f4ee3946e1cbb98d6965b0b2ff5c"
                                 redirectUri:@"http://www.sharesdk.cn"
                                 wbApiCls:[WBApi class]];
```

- **服务器托管模式初始化时**

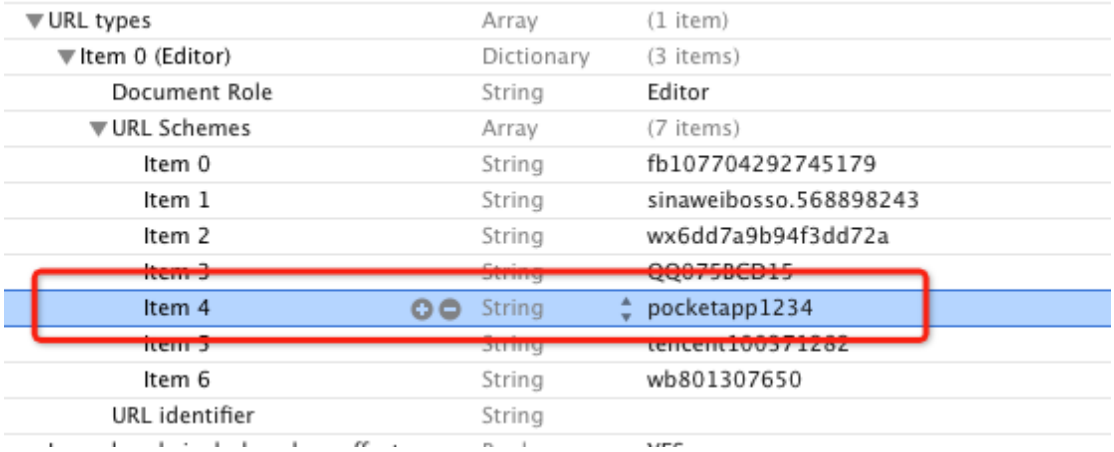
确认在初始化 SDK 后是否有调用 importTencentWeiboClass 方法，如果没有

则添加如下语句：

```
//导入腾讯微博需要的外部库类型，如果不需要腾讯微博SSO可以不调用此方法

[ShareSDK importTencentWeiboClass:[WBApi class]];
```

9.5 Pocket

打开*-Info.plist（*代表你的工程名字）。找到 URL types 配置项（如果没有则新增），展开 URL types – URL Schemes，在 URL Schemes 下分别各新增一项用于 Pocket 的 Scheme（如果不添加则会导致法返回应用）。其填写格式为：pocketapp1234，后面的数字可以为任意值。如图所示：

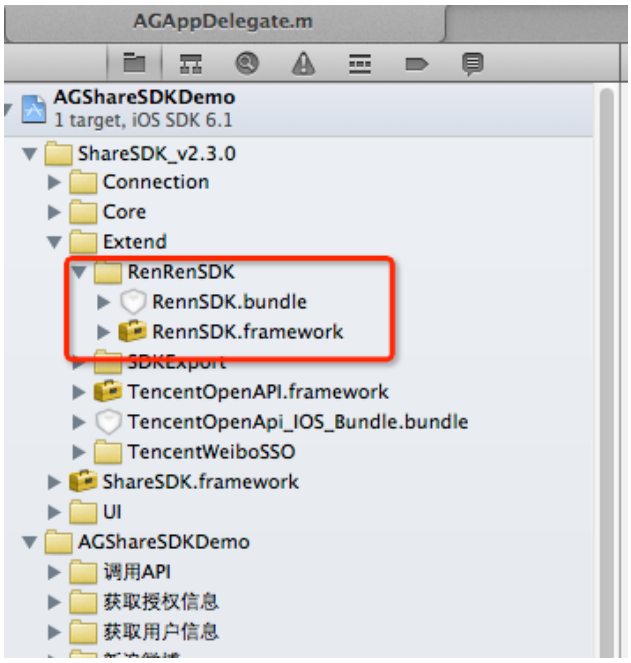


注：初始化中的回调地址必须与填写的 url scheme 一致。如：

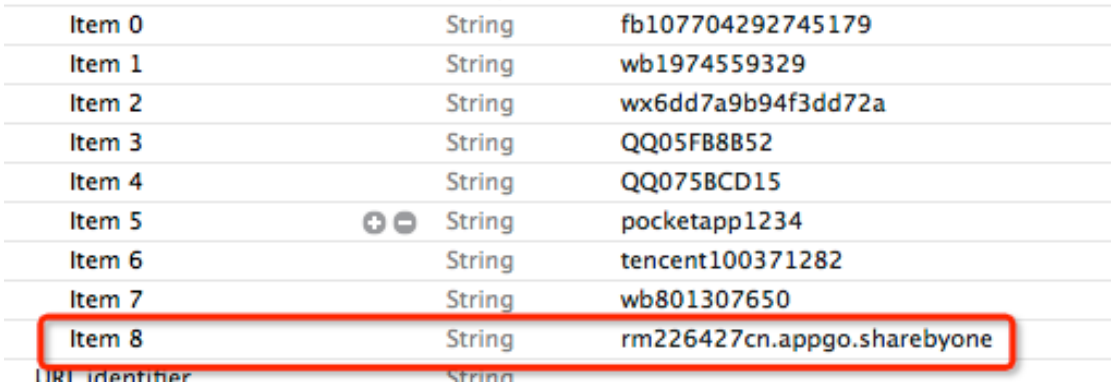
```
//初始化Pocket

[ShareSDK connectPocketWithConsumerKey:@"11496-de7c8c5eb25b2c9fcdc2b627"
                                redirectUri:@"pocketapp1234"];
```

9.6 人人网

先确认是否已把 RennSDK.framework 以及对应的头文件导入到项目中。



打开*-Info.plist（*代表你的工程名字）。找到 URL types 配置项（如果没有则新增），展开 URL types – URL Schemes，在 URL Schemes 下分别各新增一项用于人人网的 Scheme（如果不添加则会导致法返回应用）。其填写格式为：rm + appID（你在人人申请应用的 AppId） + BundleID 。 如：rm226427cn.appgo.sharebyone，如图所示：



然后打开*AppDelegate.h(*代表你的工程名字) 文件，引入头文件：

```
#import<RennSDK/RennSDK.h>
```

- **本地配置信息方式初始化时**

打开*AppDelegate.m(*代表你的工程名字) 文件，修改初始化人人网的代码，由原来的

```
//添加人人网应用
```

```
[ShareSDK connectRenRenWithAppKey:@"fc5b8aed373c4c27a05b712acba0f8c3"
                        appSecret:@"f29df781abdd4f49beca5a2194676ca4"];
```

改为

```
//添加人人网应用

[ShareSDK connectRenRenWithAppId:@"226427"
                          appKey:@"fc5b8aed373c4c27a05b712acba0f8c3"
                          appSecret:@"f29df781abdd4f49beca5a2194676ca4"
                          renrenClientClass:[RennClient class]];
```

- **服务器托管模式初始化时**

确认在初始化 SDK 后是否有调用 importRenRenClass 方法，如果没有则添加如下语句：

```
//导入人人网需要的外部库类型,如果不需要人人网SSO可以不调用此方法

[ShareSDK importRenRenClass:[RennClient class]];
```

三、接口使用

下面部分将介绍 ShareSDK 中各种功能接口的使用。

注：在使用服务器托管配置信息初始化时，由于从服务器获取信息有一定的时间延迟，因此为保证可以在正确初始化平台后调用相关功能，SDK 中提供了一个 waitAppSettingComplete 的方法，用于等待设置 App 信息完成后执行相关操作。其用法如下：

```
[ShareSDK waitAppSettingComplete:^{
    NSString *imagePath = [[NSBundle mainBundle] pathForResource:@"ShareSDK"
                                                          ofType:@"jpg"];


    //构造分享内容

    id<ISSContent> publishContent = [ShareSDK content:@"分享内容"

                                       defaultContent:@"默认分享内容，没内容时显示"

                                       image:[ShareSDK imageWithPath:imagePath]
                                       title:@"ShareSDK"
                                       url:@"http://www.sharesdk.cn"
```

```
                                                description:@"这是一条测试信息"

                                                mediaType:SSPublishContentMediaTypeNews];


    [ShareSDK showShareActionSheet:nil
                             shareList:nil
                             content:publishContent
                             statusBarTips:YES
                             authOptions:nil
                             shareOptions: nil
                             result:^(ShareType type, SSPublishContentState state, id<ISSStatusInfo>
statusInfo, id<ICMErrorInfo> error, BOOL end) {
                                        if (state == SSPublishContentStateSuccess)
                                        {

                                            NSLog(@"分享成功");

                                        }
                                        else if (state == SSPublishContentStateFail)
                                        {

                                            NSLog(@"分享失败,错误码:%d,错误描述:%@", [error

errorCode], [error errorDescription]);
                                        }
                                    }];
}];
```

此方法同样在本地信息初始化 SDK 的时候可用，会立即进行返回。

10．分享内容

在分享内容部分 ShareSDK 提供了三种接口来满足不同需求的分享功能实现。

10.1 菜单方式分享：

使用此方式进行分享会首先弹出菜单供用户选择分享的目标平台，然后再显示内容编辑界面供用户进行分享内容编辑，最后进行分享。调用该方式的接口如下：

```
NSString *imagePath = [[NSBundle mainBundle] pathForResource:@"ShareSDK"
                                                ofType:@"jpg"];


//构造分享内容
```

```
id<ISSContent> publishContent = [ShareSDK content:@"分享内容"
                                        defaultContent:@"默认分享内容，没内容时显示"
                                        image:[ShareSDK imageWithPath:imagePath]
                                        title:@"ShareSDK"
                                        url:@"http://www.sharesdk.cn"
                                        description:@"这是一条测试信息"
                                        mediaType:SSPublishContentMediaTypeNews];

[ShareSDK showShareActionSheet:nil
                    shareList:nil
                    content:publishContent
                    statusBarTips:YES
                    authOptions:nil
                    shareOptions: nil
                    result:^(ShareType type, SSPublishContentState state, id<ISSStatusInfo>
statusInfo, id<ICMErrorInfo> error, BOOL end) {
                        if (state == SSPublishContentStateSuccess)
                        {
                            NSLog(@"分享成功");
                        }
                        else if (state == SSPublishContentStateFail)
                        {
                            NSLog(@"分享失败,错误码:%d,错误描述:%@", [error
errorCode], [error errorDescription]);
                        }
                    }];
```

以上代码中主要使用[ShareSDK showShareActionSheet: shareList: content: statusBarTips: authOptions: shareOptions: esult]方法进行分享。

第一个参数用于指定菜单显示在哪个视图控制器上。如果指定为 nil，则表示使用显示在最顶层的 UIViewController 对象。(注 :如果需要适配 iPad 版本 ,需要调用 container 对象中的相关方法设置 iPad 容器，否则无法在 iPad 中正常显示。)

第二个参数则用于指定菜单弹出的分享平台列表，传入 nil 则表示显示所有分享平

台，如果开发者需要自己定制显示列表则通过此参数进行控制。ShareSDK 提供了 getShareListWithType 方法来进行分享列表的构造。

第三个参数为实现了 ISSContent 协议的对象，需要调用了 ShareSDK 的+(id<ISSContent>)content: defaultContent: image: title: url: description: mediaType:;方法来构造对象。（注：id<ISSContent>对象提供了对部分平台，如：微信、QQ、QQ 空间、邮件、短信、有道云笔记、Instapaper 平台的内容定制，通过定制可以针对不同的平台发送不同的内容。详细请参考 Demo 程序）

第四个参数为是否在状态栏上显示提示信息（包括成功和错误提示）。

第五个参数用于指定在分享过程中，需要授权时的一些信息配置。如：是否需要自动弹出授权界面然后授权、授权时是否回调服务器以及授权视图的委托等。需要说明的是在设置是否需要自动授权标志为 NO 时，则 SDK 对于需要授权的状态不会弹出授权界面，需要用户自行处理。

第六个参数用于指定分享时的一些信息配置。如：分享视图标题、一键分享列表的定制、微信、QQ 分享按钮的隐藏以及分享视图委托等。

第七个参数用于处理分享返回后续的处理方法。返回方法中包括了分享状态、分享成功后的状态信息（StatusInfo）、分享失败时的错误信息、以及本次分享会话是否结束标识。

10.2 分享内容视图方式

使用此方式直接弹出内容分享编辑框，用户编辑完内容后点击发布直接分享出去。其调用代码如下：

```
//创建分享内容

    NSString *imagePath = [[NSBundle mainBundle] pathForResource:@"image" ofType:@"jpg"];
    id<ISSContent> publishContent = [ShareSDK content:@"content"
                                       defaultContent:@""
                                       image:[ShareSDK imageWithPath:imagePath]
                                       title:nil
                                       url:nil
                                       description:nil
                                       mediaType:SSPublishContentMediaTypeText];


    //创建弹出菜单容器

    id<ISSContainer> container = [ShareSDK container];
    [container setIPadContainerWithView:sender
                                arrowDirect:UIPopoverArrowDirectionUp];


    //显示分享菜单

    [ShareSDK showShareViewWithType:ShareTypeSinaWeibo
                            container:container
                            content:publishContent
                            statusBarTips:YES
                            authOptions:nil
                            shareOptions:nil
                            result:^(ShareType type, SSPublishContentState state, id<ISSStatusInfo>
statusInfo, id<ICMErrorInfo> error, BOOL end) {
                                if (state == SSPublishContentStateSuccess)
                                {

                                    NSLog(@"发表成功");

                                }
                                else if (state == SSPublishContentStateFail)
                                {

                                    NSLog(@"发布失败!error code == %d, error code == %@",

[error errorCode], [error errorDescription]);
                                }
                            }];
```

以上代码主要使用主要使用[ShareSDK showShareViewWithType: container: content: statusBarTips: authOptions: shareOptions: result:]方法进行分享。

第一个参数为分享的目标平台类型。其他参数与菜单方式中的 content 参数含义相同。其显示效果如上面编辑分享内容视图截图。

10.3 直接分享方式

此方式不需要通过任何分享界面直接发送到指定的平台。其提供了两种不同的方法来满足不同的功能，第一种分享到单个平台，调用代码如下：

```
//创建分享内容

NSString *imagePath = [[NSBundle mainBundle] pathForResource:@"image" ofType:@"jpg"];
id<ISSContent> publishContent = [ShareSDK content:@"content"
                                          defaultContent:@""
                                          image:[ShareSDK imageWithPath:imagePath]
                                          title:nil
                                          url:nil
                                          description:nil
                                          mediaType:SSPublishContentMediaTypeText];

[ShareSDK shareContent:publishContent
                    type:ShareTypeSinaWeibo
                    authOptions:nil
                    statusBarTips:YES
                    result:^(ShareType type, SSPublishContentState state, id<ISSStatusInfo>
statusInfo, id<ICMErrorInfo> error, BOOL end) {
          if (state == SSPublishContentStateSuccess)
          {

                NSLog(@"分享成功");

          }
          else if (state == SSPublishContentStateFail)
          {

                NSLog(@"分享失败,错误码:%d,错误描述:%@", [error errorCode], [error

errorDescription]);
          }
     }];
```

以上代码中主要使用[ShareSDK shareContent: type: authOptions: statusBarTips: result:]方法进行分享。其参数说明与上面带分享界面的功能接口参数说明相同。

另外一种方法就是可以通过传入多个平台类型来进行一键分享信息。其调用代码如下：

```
//创建分享内容

NSString *imagePath = [[NSBundle mainBundle] pathForResource:@"image" ofType:@"jpg"];
    id<ISSContent> publishContent = [ShareSDK content:@"content"
                                              defaultContent:@""
                                              image:[ShareSDK imageWithPath:imagePath]
                                              title:nil
                                              url:nil
                                              description:nil
                                              mediaType:SSPublishContentMediaTypeText];

    [ShareSDK oneKeyShareContent:publishContent
                        shareList:[NSArray defaultOneKeyShareList]
                        authOptions:nil
                        statusBarTips:YES
                        result:^(ShareType type, SSPublishContentState state, id<ISSStatusInfo>
statusInfo, id<ICMErrorInfo> error, BOOL end) {
                                if (state == SSPublishContentStateSuccess)
                                {

                                    NSLog(@"分享成功");

                                }
                                else if (state == SSPublishContentStateFail)
                                {

                                    NSLog(@"分享失败,错误码:%d,错误描述:%@", [error errorCode],
[error errorDescription]);

                                }
                            }];
```

以上代码主要使用[ShareSDK oneKeyShareContent: shareList: authOptions: statusBarTips: result:]方法进行分享，其中shareList参数表示需要将内容分享到的目标平台类型列表，[NSArray defaultOneKeyShareList]表示分享到所有支持一键分享的平台中。其余参数与其他分享接口意义相同。

10.4 定制分享

对于需要定义分享视图的样式、标题以及部分属性（如控制默认分享视图中的QQ、微信分享按钮显示）或者调整分享视图时，可以在各个分享接口的shareOptions参数中进行设定。如：

```
//创建分享内容
```

```
NSString *imagePath = [[NSBundle mainBundle] pathForResource:@"image" ofType:@"jpg"];
id<ISSContent> publishContent = [ShareSDK content:@"content"
                                          defaultContent:@""
                                          image:[ShareSDK imageWithPath:imagePath]
                                          title:nil
                                          url:nil
                                          description:nil
                                          mediaType:SSPublishContentMediaTypeText];


//创建分享选项

id<ISSShareOptions> shareOptions =

        [ShareSDK defaultShareOptionsWithTitle:@"内容分享"

                                  oneKeyShareList:[NSArray defaultOneKeyShareList]
                                  qqButtonHidden:YES
                                  wxSessionButtonHidden:YES
                                  wxTimelineButtonHidden:YES
                                  showKeyboardOnAppear:NO
                                  shareViewDelegate:nil
                                  friendsViewDelegate:nil
                                  picViewerViewDelegate:nil];


    //弹出分享菜单

    [ShareSDK showShareActionSheet:nil
                           shareList:nil
                           content:publishContent
                           statusBarTips:YES
                           authOptions:nil
                           shareOptions:shareOptions
                           result:^(ShareType type,  SSPublishContentState state, id<ISSStatusInfo>
statusInfo, id<ICMErrorInfo> error, BOOL  end){
                                  if (state == SSPublishContentStateSuccess)
                                  {

                                      NSLog(@"分享成功");

                                  }
                                  else if (state == SSPublishContentStateFail)
                                  {

                                      NSLog(@"分享失败,错误码:%d,错误描述:%@", [error
errorCode], [error errorDescription]);
                                  }
```

```
                                        }];
```

ShareSDK提供了三种构造ShareOptions对象的方法，分别对应默认、简单以及应用推荐三种分享视图样式，其分别是：

+ defaultShareOptionsWithTitle: oneKeyShareList: qqButtonHidden: wxSessionButtonHidden: wxTimelineButtonHidden: showKeyboardOnAppear: shareViewDelegate: friendsViewDelegate: picViewerViewDelegate:;

+ simpleShareOptionsWithTitle: shareViewDelegate:;

+ appRecommendShareOptionsWithTile: shareViewDelegate:;

title参数表示了设置分享视图的标题，oneKeyShareList参数控制一键分享列表的显示（在默认分享视图底部显示的平台列表），qqButtonHidden、wxSessionButtonHidden、wxTimelineButtonHidden则分别是控制默认分享视图下的QQ、微信好友、微信朋友圈按钮的显示。showKeyboardOnAppear参数控制分享视图显示时是否弹出键盘。shareViewDelegate参数表示分享视图委托，friendsViewDelegate参数表示好友列表视图委托。picViewerViewDelegate参数表示照片查看视图委托。

10.5 定制属于自己的分享 UI

结合+ shareContent: type: authOptions: statusBarTips: result:;和+ authWithType: options: result:;以及+ hasAuthorizedWithType:;方法来打造属于自己的分享UI。下面代码使用简单的UITextView和UIButton来搭建简单的新浪微博分享界面。代码如下：

```
- (void)viewDidLoad
{
    [super viewDidLoad];

    _contentTextView = [[UITextView alloc] initWithFrame:CGRectMake(0.0, 0.0, 320.0, 200.0)];
    [self.view addSubview:_contentTextView];
    [_contentTextView release];

    UIButton *sendButton = [UIButton buttonWithType:UIButtonTypeRoundedRect];
    sendButton.frame = CGRectMake(0.0, 205.0, 100.0, 45.0);

    [sendButton setTitle:@"分享" forState:UIControlStateNormal];

    [sendButton addTarget:self
                   action:@selector(sendButtonClickHandler:)
                   forControlEvents:UIControlEventTouchUpInside];
    [self.view addSubview:sendButton];
```

```
}

- (void)sendButtonClickHandler:(id)sender
{
    if ([ShareSDK hasAuthorizedWithType:ShareTypeSinaWeibo])
    {
        //构造分享内容

        id<ISSContent> publishContent =
          [ShareSDK content:_contentTextView.text
                    defaultContent:nil
                    image:nil
                    title:nil
                    url:nil
                    description:nil
                    mediaType:SSPublishContentMediaTypeNews];

        [ShareSDK shareContent:publishContent
                    type:ShareTypeSinaWeibo
                    authOptions:nil
                    statusBarTips:YES
                    result:^(ShareType type, SSPublishContentState state, id<ISSStatusInfo>
statusInfo, id<ICMErrorInfo> error, BOOL end) {
                    if (state == SSPublishContentStateSuccess)
                    {

                        NSLog(@"分享成功");

                    }
                    else if (state == SSPublishContentStateFail)
                    {

                        NSLog(@"分享失败,错误码:%d,错误描述:%@", [error errorCode], [error
errorDescription]);

                    }
                }];
    }
    else
    {
        [ShareSDK authWithType:ShareTypeSinaWeibo
                    options:nil
                    result:^(SSAuthState state, id<ICMErrorInfo> error) {
                    if (state == SSPublishContentStateSuccess)
                    {
```

```
                                //构造分享内容

                                id<ISSContent> publishContent =
                                    [ShareSDK content:_contentTextView.text
                                              defaultContent:nil
                                              image:nil
                                              title:nil
                                              url:nil
                                              description:nil
                                              mediaType:SSPublishContentMediaTypeNews];


                                 [ShareSDK shareContent:publishContent
                                           type:ShareTypeSinaWeibo
                                           authOptions:nil
                                           statusBarTips:YES
                                           result:^(ShareType type, SSPublishContentState state,
id<ISSStatusInfo> statusInfo, id<ICMErrorInfo> error, BOOL end) {
                                               if (state == SSPublishContentStateSuccess)
                                               {

                                                        NSLog(@"分享成功");

                                               }
                                               else if (state == SSPublishContentStateFail)
                                               {

                                                        NSLog(@"分享失败,错误码:%d,错误描述:%@", [error
errorCode], [error errorDescription]);
                                                }
                                           }];
                                    }
                                }];
        }
}
```

10.6 iPad 版本分享界面集成

如果需要在iPad中显示分享，则需要使用+ container;功能接口创建一个容器对象并调用容器对象的setIPadContainer系列方法来设置iPad版显示容器。其实现代码如下：

```
//创建弹出菜单容器
```

```
id<ISSContainer> container = [ShareSDK container];
 [container setIPadContainerWithView:sender
                                 arrowDirect:UIPopoverArrowDirectionUp];


//弹出分享菜单

[ShareSDK showShareActionSheet:container
                                shareList:nil
                                content:publishContent
                                statusBarTips:YES
                                authOptions:authOptions
                                shareOptions:shareOptions
                                result:^(ShareType type, SSPublishContentState state, id<ISSStatusInfo>
statusInfo, id<ICMErrorInfo> error, BOOL end) {
                                    if (state == SSPublishContentStateSuccess)
                                    {

                                        NSLog(@"分享成功");

                                    }
                                    else if (state == SSPublishContentStateFail)
                                    {

                                        NSLog(@"分享失败,错误码:%d,错误描述:%@", [error

errorCode], [error errorDescription]);
                                    }
                                }];
```

11. 激活 / 禁止 SSO 授权。

用于控制平台是否使用SSO方式进行授权(目前只支持新浪微博、Facebook、QQ空间、腾讯微博、Pocket。默认情况下是激活SSO授权方式。)，代码如下：

激活SSO授权

```
[ShareSDK ssoEnabled:YES];
```

禁止SSO授权

```
[ShareSDK ssoEnabled:NO];
```

12. 获取当前用户信息

获取当前授权用户信息，其调用方式如下：

```
[ShareSDK getUserInfoWithType:ShareTypeSinaWeibo
                  authOptions:nil
                  result:^(BOOL result, id<ISSUserInfo> userInfo, id<ICMErrorInfo> error) {
                      if (result)
                      {
                          NSLog(@"成功");
                      }
                      else
                      {
                          NSLog(@"失败");
                      }
                  }];
```

其中第一个参数为平台类型，用于指定获取哪个平台的授权用户信息。如果指定平台的用户尚未进行授权则弹出授权界面。第二个参数为授权选项参数。第三个参数为授权返回的处理方法。

13. 关注用户

关注指定用户（此接口目前仅支持新浪微博和腾讯微博），其调用方式如下：

```
//关注用户

[ShareSDK followUserWithType:ShareTypeSinaWeibo
                  field:@"ShareSDK"
                  fieldType:SSUserFieldTypeTypeName
                  authOptions:nil
                  viewDelegate:nil
                  result:^(SSResponseState state, id<ISSUserInfo> userInfo, id<ICMErrorInfo>
error) {
                      NSString *msg = nil;
                      if (state == SSResponseStateSuccess)
                      {
                          NSLog(@"关注成功");
                      }
                      else if (state == SSResponseStateFail)
                      {
                          NSLog(@"%@", [NSString stringWithFormat:@"关注失
```

```
败:%@", error.errorDescription]);
                                        }
                                    }];
```

其中第一个参数为关注用户所属平台类型，第二个参数为关注用户的昵称或者用户ID（具体指示昵称还是用户ID由第三个参数控制），第三个参数为标识第二个参数的字段类型）。第四个参数为授权选项配置，第五个参数为视图委托，主要用于Facebook关注时弹出的添加好友界面的委托。

14 获取关注列表

获取当前授权用户的关注列表（此接口目前仅支持新浪微博、腾讯微博、Facebook、Twitter），其调用方式如下：

```
id<ISSPage> page = [ShareSDK pageWithPageNo:1 pageSize:0];

[ShareSDK getFriendsWithType:ShareTypeSinaWeibo
                        page:page
                        authOptions:nil
                        result:^(BOOL result, NSArray *users, id<ISSPage> currPage, id<ISSPage>
prevPage, id<ISSPage> nextPage, BOOL hasNext, id<ICMErrorInfo> error) {
                            if (result)
                            {

                                NSLog(@"成功");

                            }
                            else
                            {

                                NSLog(@"失败");

                            }
                    }];
```

其中第一个参数为平台类型，用于指定获取哪个平台的授权用户关注列表，如果指定平台的用户尚未进行授权则弹出授权界面。第二个参数为取关注列表的分页对象，用于指定要获取那一页的好友列表数据；ShareSDK中提供了两种构造分页对象的方法+ pageWithCursor:;和+ pageWithPageNo: pageSize:;。第三个参数为授权选项设置，第四个参数为关注列表返回的处理方法。

15 用户授权

要想操作相关平台提供的方法必须要先取得用户授权。ShareSDK提供了单独的授权接口来实现用户的登录授权。其代码如下：

```
[ShareSDK authWithType:ShareTypeSinaWeibo options:nil result:^(SSAuthState state,
id<ICMErrorInfo> error) {
        if (state == SSAuthStateSuccess)
        {
            NSLog(@"成功");
        }
        else if (state == SSAuthStateFail)
        {
            NSLog(@"失败");
        }
    }];
```

第一个参数为需要进行授权的平台类型。第二个参数为授权选项设置，如果需要定制授权视图的样式需要指定此参数。第三个参数则是授权返回结果的处理方法。需要注意的是ShareSDK中提供的各种需要进行授权的接口（包括分享、获取用户信息等）都已经在内部进行授权检测判断，不需要开发者单独进行授权。

授权界面效果如下所示：

授权界面

16. 用户取消授权

如果需要取消授权可以调用ShareSDK的cancelAuthWithType接口，如下：

```
[ShareSDK cancelAuthWithType:ShareTypeSinaWeibo];
```

17. 从外部授权信息中设置 ShareSDK 平台授权。

如果代码中已实现某平台的授权，那么可以通过ShareSDK中的+ setCredential:;接口设置平台的授权信息。如下：

```
id<ISSCredential> newCredential = [ShareSDK credentialWithSourceData:sourceData
type:ShareTypeSinaWeibo];


//设置新浪微博使用新的授权凭证

[ShareSDK setCredential:newCredential type:ShareTypeSinaWeibo];
```

其中sourceData为授权返回的数据。

四、附录

分享平台注册申请应用地址对照表

新浪微博	http://open.weibo.com
腾讯微博	http://dev.t.qq.com
QQ空间	http://connect.qq.com/intro/login/
网易微博	http://open.t.163.com
搜狐微博	http://open.t.sohu.com
豆瓣	http://developers.douban.com
人人网	http://dev.renren.com

开心网	http://open.kaixin001.com
Instapaper	http://www.instapaper.com/main/request_oauth_consumer_token
有道云笔记	http://note.youdao.com/open/developguide.html#app
Facebook	https://developers.facebook.com
Twitter	https://dev.twitter.com
搜狐随身看	https://open.sohu.com
QQ好友分享	http://mobile.qq.com/api/
微信	http://open.weixin.qq.com
Pocket	http://getpocket.com/developer/
印象笔记	http://dev.yinxiang.com
LinkedIn	https://www.linkedin.com/secure/developer
Google+	https://code.google.com/apis/console

本地错误码对照表

注：此表格不包含各平台调用接口后返回错误码。如果需要了解接口返回错误码请登录相关开放平台网站进行查看。

错误码	**错误描述**
-100	尚未初始化SDK
-101	传入参数无效
-102	不支持此功能
-103	尚未授权
-104	该功能不支持iOS %@以下系统版本
-105	尚未集成该平台!
-106	设备尚未连接网络!
-999	未知错误
-20001	该设备不允许进行打印

-18001	该设备不支持邮件分享或尚未设置邮件帐号
-19001	该设备不支持短信分享
-10001	取消添加好友
-11001	此方法已过时，请使用getFriendsWithPage方法代替
-22001	尚未集成微信接口
-22002	未知的消息发送类型
-22003	尚未安装微信
-22004	当前微信版本不支持该功能
-22005	尚未设置微信的URL Scheme
-24001	发送失败
-24002	尚未安装QQ
-24003	当前QQ版本不支持该功能
-24004	未知的消息发送类型
-24005	尚未集成QQ接口
-24006	尚未设置QQ的URL Scheme

微信分享内容说明

对于微信分享需要指定分享类型，其中类型包括：文本（Text）、图像（Image）、新闻（News）、音乐（Music）、视频（Video）、应用（App）以及 Gif。在进行 ISSContent 内容构造时通过 mediaType 参数来指定是分享那种类型，此参数主要为微信和 QQ 好友分享提供。注：不同的分享类型，要求填写的参数也不一样，下表描述了分享类型与参数之间的关系：

类型	字段	是否必填

SSPublishContentMediaTypeText	content	是
SSPublishContentMediaTypeImage	content	否
	image	是
SSPublishContentMediaTypeNews	content	是
	image	否
	url	是
	title	否
SSPublishContentMediaTypeMusic	content	是
	image	否
	url	是
	title	否
	musicFileUrl	否
SSPublishContentMediaTypeVideo	content	是
	image	否
	url	是
	title	否
SSPublishContentMediaTypeApp	content	是
	image	否
	url	是
	title	否
	fileData	否
	extInfo	否
SSPublishContentMediaTypeNonGif	image	是
	emoticonData	是