

# Weibo IOS SDK 文档

北京新潮讯捷信息技术有限公司

编号：WEIBO\_IOS\_SDK

版本：WEIBO\_IOS\_SDK V2.1.0

修订记录

时间	文档版本	修订人	备注
2012/7/19	1.0.0	陈行政	初稿
2013/1/30	1.0.1	陈行政	文档整合
2013/1/31	1.1.0	陈行政	文档更新
2013/3/12	2.0.0	洪涛	Sdk 升级为 2.0 版本
2013/4/16	2.1.0	唐庆杰	SDK 升级到 2.1 版本

## 目录

Weibo iOS SDK 文档 .....	1
Weibo SDK.....	3
一、    概述.....	3
名词解释.....	3
二、    流程.....	4
三、    集成步骤及示例分析.....	5
1.    新建工程，引入相关文件 .....	5
2.    设置编译选项.....	5
3.    定义所需常量.....	5
4.    实现 WeiboSDKDelegate 协议.....	6
5.    注册 appkey(clientid).....	7
6.    重写 AppDelegate 的 handleOpenURL 和 openURL 方法: .....	7
7.    场景 0: SSO 微博客户端授权认证.....	7
8.    场景 1: 从第三方方向微博发送消息.....	8
9.    场景 2: 从微博的发布界面调起第三方，向第三方请求数据。 .....	9
四、    操作过程解析.....	10
1.    SSO 登录过程 .....	10
2.    Oauth2.0 认证登录过程.....	10
3.    从第三方应用提取信息，分享微博.....	11
4.    分享微博.....	12

# Weibo SDK

## 一、概述

微博 SDK 为开发者提供访问 oauth2.0 授权认证，并集成 sso 登录功能，使第三方应用可通过新浪微博官方客户端快速通过 Oauth2.0 授权，并完成用户登录操作。提供微博分享功能，可直接通过微博客户端分享微博。

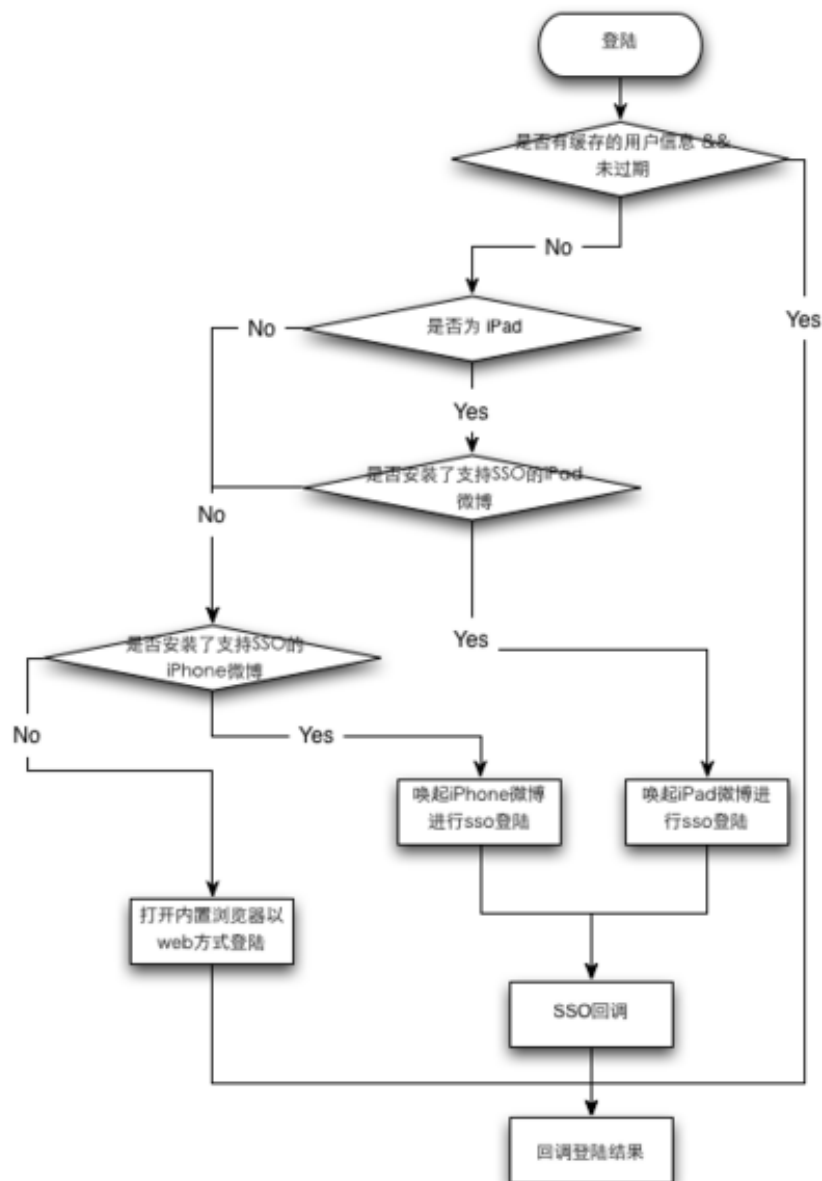
本文档将对使用 IOS SDK 时所用的一些参数、接口进行说明，并分析一个简单示例，帮助第三方更方便的使用 SDK（一些不使用的接口只做简单说明）。

## 名词解释

AppKey	分配给每个第三方应用的 app key。用于鉴权身份，显示来源等功能。
AccessToken	表示用户身份的 token，用于微博 API 的调用。
ExpirationDate	过期时间，用于判断登录是否过期。
RedirectURI	应用回调页面，可在新浪微博开放平台->我的应用->应用信息->高级应用->授权设置->应用回调页中找到。
ssoCallbackScheme	sso 回调地址，info.plist 中定义的 URL types，用于在微博客户端完成 sso 登录后进行回调。

## 二、 流程

### 1: 认证授权流程



使用带 sso 功能的 sdk 进行登录，只需调用登录接口，并完成回调方法对接收登录结果即可。SDK 中自动完成对是否进行 SSO 登录的判断，若支持，则唤起微博客户端，用户确认后返回请求的应用，SDK 对请求结果进行解析，最后交给第三方实现的回调方法进行处理；否则，SDK 将通过内置浏览器请求登录，用户输入用户名密码提交后，仍是 SDK 解析请求结果，并由第三方应用实现相应的回调方法进行最后处理。

### 2: 微博分享流程

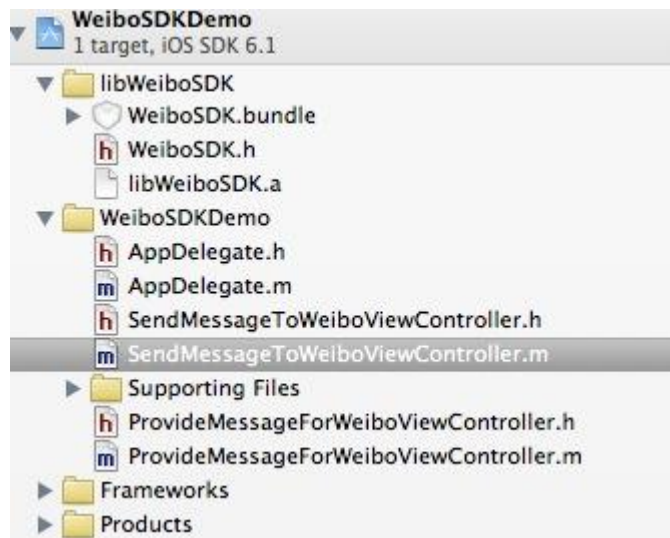
微博分享分两种场景：一是从第三方应用分享信息到微博；二是微博主动唤起第三方应用，并提取信息返回到微博客户端，进行分享。

### 三、 集成步骤及示例分析

以实际例子分析第三方应用引入 WeiboSDK 的过程。具体代码参考 WeiboSDKDemo 工程。

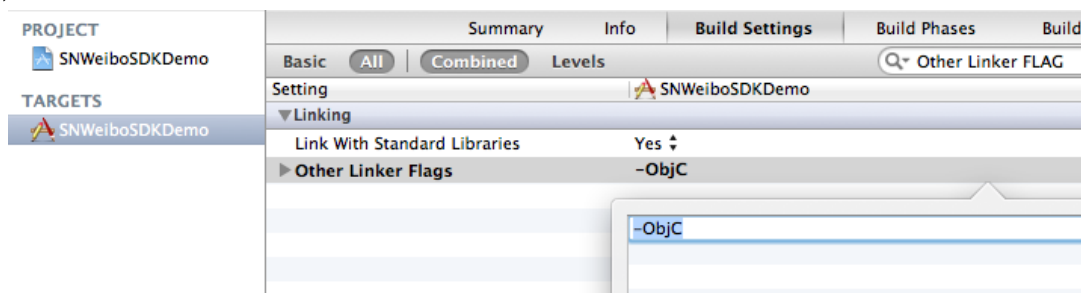
#### 1. 新建工程，引入相关文件

创建新的工程，将图中 libWeiboSDK 文件拷贝并添加至工程中。



#### 2. 设置编译选项

在工程中引入静态库之后，需要在编译时添加 `-objc` 编译选项，避免静态库中类加载不全造成程序崩溃。方法：程序 Target->Build Settings->Linking 下 Other Linker Flags 项添加 `-ObjC`。如图：



#### 3. 定义所需常量

定义应用 SSO 登录或者 OAuth2.0 认证所需的几个常量

- AppKey: 第三方应用申请的 appkey，用来身份鉴证、显示来源等；
- AppRedirectURL: 应用回调页，在进行 OAuth2.0 登录认证时所用。对于 Mobile 客

户端应用来说，是不存在 Server 的，故此处的应用回调页地址只要与新浪微博开放平台->我的应用->应用信息->高级应用->授权设置->应用回调页中的 url 地址保持一致就可以了，如图所示：



注：appkey 和 redirect\_uri 在开放平台（<https://open.weibo.cn/>）上获取。

- URL types: 修改 info.plist 文件 URL types 项为自己的 sso 回调地址，“WB[你的应用程序的 Appkey]”，例如:wb204543436852。

▼ URL types	Array	(1 item)
▼ Item 0 (Editor)	Dictionary	(3 items)
Document Role	String	Editor
URL identifier	String	com.weibo
▼ URL Schemes	Array	(1 item)
Item 0	String	wb2045436852

- Bundle id: 向开放平台注册应用程序的 bundle id，目前只能邮件给相关人员进行人工注册。联系方式：蒋斌([jiangbin1@staff.sina.com.cn](mailto:jiangbin1@staff.sina.com.cn))。  
邮件内容：  
1: appkey  
2: bundle id

#### 4. 实现 WeiboSDKDelegate 协议

第三方应用需要实现自己的 WeiboSDKDelegate，来处理用户登录、退出以及取消登录的回调。需要实现如下方法：

```
/**
 * 收到一个来自微博客户端程序的请求
 *
 * 收到微博的请求后，第三方应用应该按照请求类型进行处理，处理完后必须通过
 * [WeiboSDK sendResponse:] 将结果回传给微博
 * @param request 具体的请求对象
 */
- (void)didReceiveWeiboRequest:(WBBaseRequest *)request;
```

```
/**
收到一个来自微博客户端程序的响应

收到微博的响应后，第三方应用可以通过响应类型、响应的数据和 [WBBaseResponse
userInfo] 中的数据完成自己的功能
@param response 具体的响应对象
*/
- (void)didReceiveWeiboResponse:(WBBaseResponse *)response;
```

## 5. 注册 appkey(clientid)

程序启动时，在代码中向微博终端注册你的 Appkey，如果首次集成微博 SDK,建议打开调试选项以便输出调试信息。

```
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:
(NSDictionary *)launchOptions
{
[ WeiboSDK registerApp:"wb204543436852" ];
[ WeiboSDK enableDebugMode:YES ];
}
```

## 6. 重写 AppDelegate 的 handleOpenURL 和 openURL 方法:

用于实现微博客户端向第三方应用返回的处理结果。

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

## 7. 场景 0: SSO 微博客户端授权认证

```

WBAuthorizeRequest *request = [WBAuthorizeRequest request];
request.redirectURI = kRedirectURI;
request.scope = @"email,direct_messages_write";
request.userInfo = @{@"SSO_From": @"SendMessageToWeiboViewController",
                    @"Other_Info_1": [NSNumber numberWithInt:123],
                    @"Other_Info_2": @[@"obj1", @"obj2"],
                    @"Other_Info_3": @{@"key1": @"obj1", @"key2": @"obj2"}};
[WeiboSDK sendRequest:request];

```

调用 `sendRequest` 的方法后会跳转到微博程序。如果当前微博客户端没有账号，则进入登录界面；如果当前微博客户端已经有账户，则进入账户管理界面，选择要向第三方授权的账户。当授权完成后会回调给第三方应用程序，第三方实现 `WeiboSDKDelegate` 的 `didReceiveWeiboResponse` 方式监听此次请求的 `response`。

```

- (void)didReceiveWeiboRequest:(WBBaseResponse *) response
{
    if ([response isKindOfClass:WBAuthorizeResponse.class])
    {
        NSString *title = @"认证结果";
        NSString *message = [NSString stringWithFormat:@"响应状
态: %d\nresponse.userId: %@\nresponse.accessToken: %@\n响应UserInfo数据: %@\n
原请求UserInfo数据: %@",
                           response.statusCode, [(WBAuthorizeResponse
*)response userID], [(WBAuthorizeResponse *)response accessToken],
                           response.userInfo, response.requestUserInfo];

        ...
    }
}

```

## 8. 场景 1：从第三方方向微博发送消息

```

- (void)sendWebContent
{
    WBWebpageObject *pageObject = [ WBWebpageObject object ];
    PageObject.objectID = @"identifier1";
    pageObject.thumbnailData =[ UIImage imageNamed:@"1.jpg" ];
    pageObject.title = @"Sample Title";
    PAGE 1 OF 2
    pageObject.description = @"Sample Description";
    pageObject.webpageUrl = @"http://www.weibo.com";
    WBMessageObject *message = [ [ WBMessageObject alloc ] message ];
    message.text = @"This is a test";
    message.mediaObject = pageObject;
    WBSendMessageToWeiboRequest *req = [ [ WBSendMessageToWeiboRequest

```



```

alloc ] init ] autorelease ];
req.message = message;
[ WeiboSDK sendRequest:req ];
}

```

调用 `sendRequest` 方法后会跳转到微博程序的发布界面，当用户完成消息发送以后会回调给第三方应用程序，第三方实现 `WeiboSDKDelegate` 的 `didReceiveWeiboResponse` 方式监听此次请求的 `response`。

```

- (void)didReceiveWeiboRequest:(WBBaseResponse *) response
{
if ( [ response isKindOfClass: [ WBSendMessageToWeiboResponse class ] ] )
{
NSString *strMsg = [ NSString stringWithFormat:"发送结果代码: %d",
response.statusCode ];
NSLog(@"打印消息中的设置信息%@",response.userInfo);
...
}
}

```

## 9. 场景 2：从微博的发布界面调起第三方，向第三方请求数据。

第三方案程序需要实现 `WeiboSDKDelegate` 中的 `didReceiveWeiboRequest` 的方法。

```

- (void)didReceiveWeiboRequest:(WBBaseRequest *)request
{
if ( [ request isKindOfClass:[WBProvideMessageForWeiboRequest class ] ] )
{
//展示本地数据，供用户选择分享到微博。
ProvideMessageForWeiboViewController *controller =
[ [ ProvideMessageForWeiboViewController alloc ] initWithArray:dataArray ];
...
}
}

```

当用户选择了第三方提供的的数据以后，将数据封装成 `WBMessageObject` 传回给微博。

```

- (void) provideMessageToWeibo:(WBMessageObject *)message
{
WBProvideMessageForWeiboResponse *rep =
[ [ [ WBProvideMessageForWeiboResponse alloc ] init ] autorelease ] ;
rep.message = message;
[ WeiboSDK sendResponse:rep ];
}

```

## 四、 操作过程解析

### 1. SSO 登录过程

下图绘出了第三方应用通过 iOS SDK 进行 SSO 登录的基本流程。

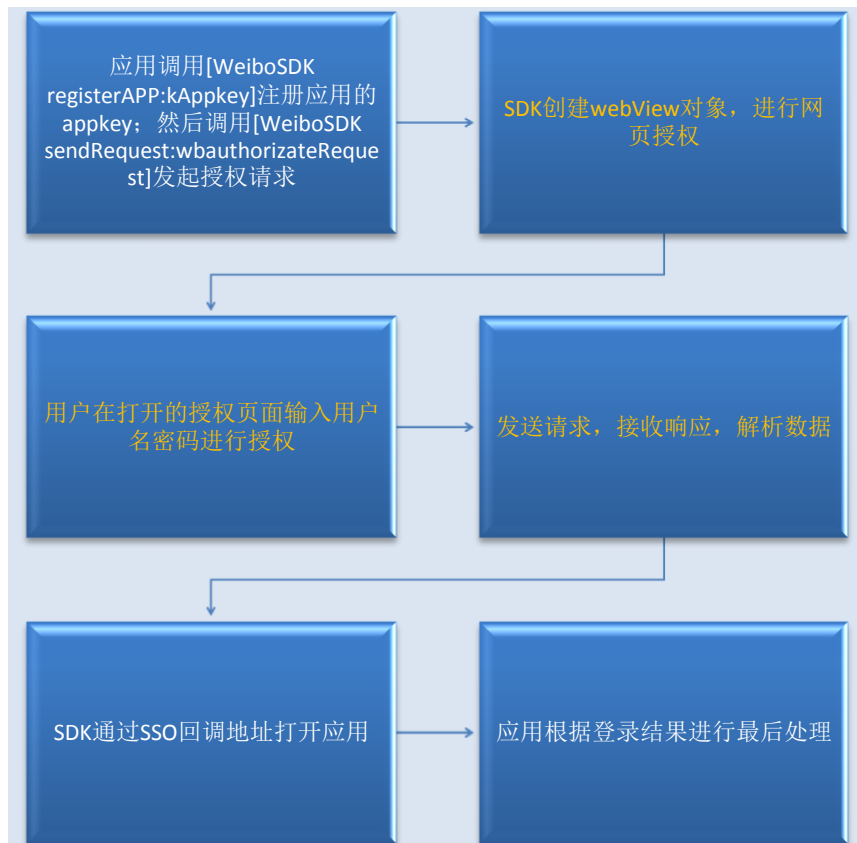
注:

- a) 图中黄色标注的文字描述的为微博 SDK 或微博客户端处理的流程，第三方应用需要实现的是白色文字描述的内容；
- b) 若用户在微博客户端登录时(未完成)，手动唤起应用到前台，应用需要在 [AppDelegate applicationDidBecomeActive] 中调用 [SinaWeibo applicationDidBecomeActive]以取消当前登录。



### 2. Oauth2.0 认证登录过程

若没有 SSO 登录的环境(未安装微博官方客户端或版本不支持)，则进行 Oauth2.0 认证登录。



### 3. 从第三方应用提取信息，分享微博



#### 4. 分享微博

