

微博iOS平台SDK文档

编号：WEIBO_IOS_SDK

版本：WEIBO_IOS_SDK V3.2.5

修订记录：

时间	文档版本	修订人	备注
2012/07/19	1.0.0	陈行政	初稿
2013/01/30	1.0.1	陈行政	文档整合
2013/01/31	1.1.0	陈行政	文档更新
2013/03/12	2.0.0	洪涛	SDK升级到2.0版本
2013/04/16	2.1.0	唐庆杰	SDK升级到2.1版本
2013/09/09	2.3.0	唐庆杰	新增登陆登出按钮、好友邀请功能
2013/10/20	2.3.2	洪涛	重写文档，添加完整SDK指引
2014/11/17	3.0.0	邱文杰	SDK升级到3.0版本
2015/02/04	3.1.0	李靖宇	新增短信注册通道
2016/05/11	3.1.4	李靖宇	新增私信分享，支持ip v 6 - only
2017/4/19	3.2.0	李靖宇	SDK升级到3.2版本
2017/7/31	3.2.1	李靖宇	SDK新增多图，视频分享以及分享到微博story
2017/9/18	3.2.5	王美美	替换UIWebView，并优化SDK

目录

Weibo SDK	3
一.SDK接入设置	3
1.注册成为开发者，创建移动应用.....	3
2.设定授权回调页.....	3
3.设定Apple ID 和 Bundle ID.....	4
4.设置工程回调URL Scheme	5
5.添加SDK文件到工程	5
6.在工程中引入静态库之后,需要在编译时添加 -objc 编译选项.....	5
7.添加FrameWork文件到工程	6
8.定义应用 SSO 登录或者 Oauth2.0 认证所需的几个常量	6
9.注册 appkey(clientid)	6
10.重写AppDelegate 的handleOpenURL和openURL方法	7
二、应用场景代码示例.....	8
1.SSO 微博客户端授权认证	8
2.从第三方应用向微博发送请求.....	8
3.从微博的发布界面调起第三方,向第三方请求数据.....	8
4.用户取消对应用的授权.....	9
5.aid 相关 aid 是设备唯一的移动设备指纹.....	9
6.SDK连接到微博	10
7.SDK分享多图，视频修改，新增SDK分享单图，视频到story	10

Weibo SDK

一.SDK接入设置

1.注册成为开发者，创建移动应用



如果你还不是一名开发者，请先注册成为开发者，具体参考新手指南：<http://open.weibo.com/wiki/%E6%96%B0%E6%89%8B%E6%8C%87%E5%8D%97>

创建应用时，开发者需要谨慎选择应用对应平台，不同的平台建议使用不同APPKEY开发。

应用平台：	<input type="checkbox"/> iPhone	<input type="checkbox"/> Android	<input type="checkbox"/> BlackBerry
	<input type="checkbox"/> Windows Phone	<input type="checkbox"/> Symbian	<input type="checkbox"/> WebOS
	<input type="checkbox"/> Other		

本文档读者请选择iPhone

2.设定授权回调页

请在“我的应用 - 应用信息 - 高级信息”中填写您的应用回调页，这样才能使OAuth2.0授权正常进行。如果您的APPSECRET发生泄露，您也可以通过该页面中的重置按钮对其重置，如下图所示：



注意：iOS应用推荐使用默认授权回调页！地址为：
<https://api.weibo.com/oauth2/default.html>

3. 设定 Apple ID 和 Bundle ID

请在“我的应用 - 应用信息 - 基本信息”中填写您的Apple ID 和 Bundle ID，这样您的应用才能正常使用微博iOS SDK授权和回调。（更改设置有延时，建议退出账号重新登录后再测试）

应用基本信息

应用类型：普通应用 - 客户端

应用名称：客户端sdk测试使用 该名称也用于来源显示，不超过10个汉字或20个字符

应用平台：手机 查看移动客户端接入指南

☒ iPhone ☐ Android ☐ BlackBerry
☐ Windows Phone ☐ Symbian
☐ WebOS ☐ Other

* Apple ID: 如何查找Apple ID

* Bundle ID:

注：Apple ID如果没有的话，先随意填写，当获取了合法的Apple ID之后请马上到这个页面修改为正式版本。而Bundle ID需要和工程设置保证一致，在XCODE5下Bundle的截图如下：



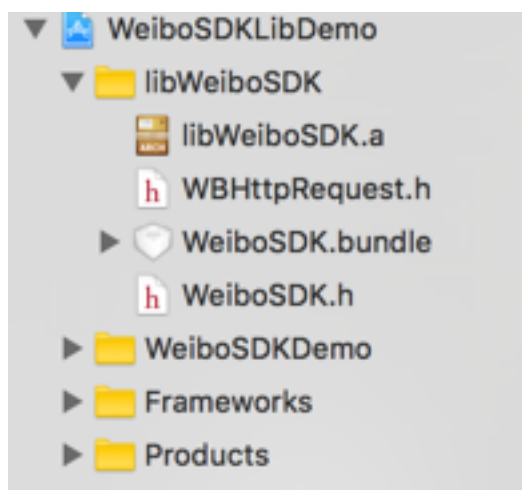
4.设置工程回调URL Scheme

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



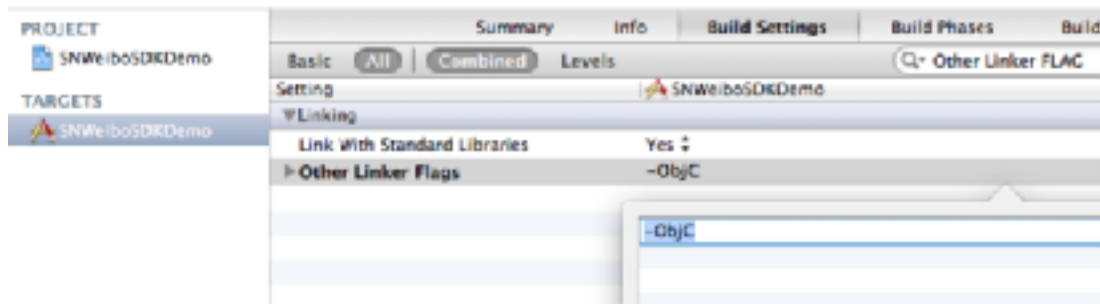
5.添加SDK文件到工程

将从GitHub上下载的libWeiboSDK文件夹添加至工程，其中包含WeiboSDK.h、WBHttpRequest.h这两个.h文件以及libWeiboSDK.a和WeiboSDK.bundle，统共4个文件。



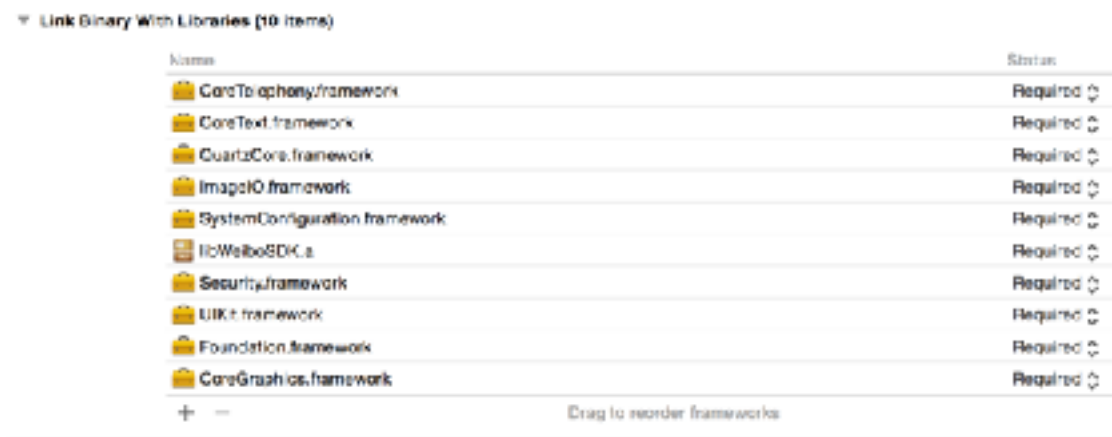
6.在工程中引入静态库之后,需要在编译时添加 -objC 编译选项

避免静态库中类加载 不全造成程序崩溃。方法:程序 Target->Buid Settings->Linking 下 Other Linker Flags 项添加-ObjC。



7.添加Framework文件到工程

在工程中修改Other Linker Flags后，需要修改编译步骤的链接库设置，避免链接阶段由于库的设置错误导致程序崩溃。方法:程序 Target->Build Phases->Link Binary With Libraries下添加以下Framework至工程中。需要添加的Frameworks为: QuartzCore.framework、ImageIO.framework、SystemConfiguration.framework、Security.framework、CoreTelephony.framework、CoreText.framework、UIKit.framework、Foundation.framework、CoreGraphics.framework、libz.dylib、libsqlite3.dylib、WebKit.framework。



8.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:

```
#define kAppKey @"2045436852"
#define kRedirectURI @"http://www.sina.com"
```

9.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 enableDebugMode:YES];
    [WeiboSDK registerApp:kAppKey];
}

```

10.重写AppDelegate 的handleOpenURL和openURL方法

```

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

- (BOOL)application:(UIApplication *)application handleOpenURL:(NSURL *)url
{
    return [WeiboSDK handleOpenURL:url delegate:self];
}

```

二、应用场景代码示例

1.SSO 微博客户端授权认证

```
{
    WBAuthorizeRequest *request = [WBAuthorizeRequest request];
    request.redirectURI = kRedirectURI;
    request.scope = @"all";
    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.

此中UserInfo内容为用户自定义(可不填写), 微博回调Response中会通过 requestUserInfo包含原 request.userInfo 中的所有数据,用于第三方自定义操作或request区分。

2.从第三方应用向微博发送请求

代码示例如:

```
WBSendMessageToWeiboRequest *request = [WBSendMessageToWeiboRequest requestWithMessage:[self
    messageToShare]];
request.userInfo = @{@"ShareMessageFrom": @"SendMessageToWeiboViewController",
                    @"Other_Info_1": [NSNumber numberWithInt:123],
                    @"Other_Info_2": @{@"obj1": @"obj2"},
                    @"Other_Info_3": @{@"key1": @"obj1", @"key2": @"obj2"}};
[WeiboSDK sendRequest:request];
```

同上此中UserInfo内容为用户自定义(可不填写), 微博回调Response中会通过 requestUserInfo包含原 request.userInfo 中的所有数据,用于第三方自定义操作或request区分。

3.从微博的发布界面调起第三方,向第三方请求数据

第三程序需要实现 WeiboSDKDelegate 中的 didReceiveWeiboRequest 的方法,当用户选择了第三方提供的数据以后,将数据封装成 WBMessageObject

```
-(WBMessageObject *)messageToShare
{
    WBMessageObject *message = [WBMessageObject message];
    message.text = @"测试使用";
    return message;
}
```

构造WBProvideMessageForWeiboResponse对象如:

```
WBProvideMessageForWeiboResponse *response = [WBProvideMessageForWeiboResponse responseWithMessage:
    [self messageToShare]];
[WeiboSDK sendResponse:response];
```


通过sendResponse方法回传数据给微博客户端完成此次任务。

4.用户取消对应用的授权

调用[WeiboSDK logOutWithToken: delegate:];方法即可

调用此接口后，微博SDK会发起网络请求使token失效

@param token 第三方应用之前申请的Token

@param delegate WBHttpRequestDelegate对象，用于接收微博SDK对于发起的接口请求的请求的响应

应用可实现WBHttpRequestDelegate中的

-(void)request:(WBHttpRequest*)request didReceiveResponse:(NSURLResponse *)response;

-(void)request:(WBHttpRequest *)request didFailWithError:(NSError *)error;

-(void)request:(WBHttpRequest *)request didFinishLoadingWithResult:(NSString *)result;

方法用于监听此次Http请求，如：

```
- (void)request:(WBHttpRequest *)request didFinishLoadingWithResult:(NSString *)result
{
    NSString *title = nil;
    UIAlertView *alert = nil;

    title = @"收到网络回调";
    alert = [[UIAlertView alloc] initWithTitle:title
                                       message:[NSString stringWithFormat:@"%s",result]
                                       delegate:nil
                                       cancelButtonTitle:@"确定"
                                       otherButtonTitles:nil];

    [alert show];
    [alert release];
}

- (void)request:(WBHttpRequest *)request didFailWithError:(NSError *)error;
{
    NSString *title = nil;
    UIAlertView *alert = nil;

    title = @"请求异常";
    alert = [[UIAlertView alloc] initWithTitle:title
                                       message:[NSString stringWithFormat:@"%s",error]
                                       delegate:nil
                                       cancelButtonTitle:@"确定"
                                       otherButtonTitles:nil];

    [alert show];
    [alert release];
}
```

5.aid 相关 aid 是设备唯一的移动设备指纹

如果您的应用需要使用一个设备唯一标识符来区分设备，可以使用aid值。

使用方法见以下方法：

+ (NSString*)getWeiboAid;

方法声明在 WeiboSDK.h 中,说明如下：

1)该方法返回当前设备的aid值。返回的aid值可能为nil,当值为nil 时会尝试向服务器获取aid值。

2)当获取成功时(aid值变为有效值)时,SDK会发出名为WeiboSDKGetAidSucessNotification的通知,通知中带有aid值。

3)当获取失败时,SDK会发出名为WeiboSDKGetAidFailNotification的通知， 通知中带有NSError 对象。

6.SDK连接到微博

在sdk3.2版本中添加了连接到微博的功能，即呼起微博客户端或打开微博H5页面，SDK自动检测是否安装微博客户端，当调用SDK相关方法时：

- 有的话呼起微博客户端定位到对应界面；
- 没有的话打开 webView 加载相应的微博H5页面；

具体的功能分为8项，如下图所示

```
linkToUserButton = [self setupButtonWithTitle:@"连接到指定用户的微博个人主页" andSelector:
    @selector(linkToUser:)];
linkToSingleBlogButton = [self setupButtonWithTitle:@"连接到指定的单条微博详情页" andSelector:
    @selector(linkToSingleBlog:)];
linkToArticleButton = [self setupButtonWithTitle:@"连接到指定的微博头条文章页" andSelector:
    @selector(linkToArticle:)];
shareToWeiboButton = [self setupButtonWithTitle:@"分享到微博" andSelector:@selector
    (shareToWeibo:)];
commentToWeiboButton = [self setupButtonWithTitle:@"评论指定的微博" andSelector:@selector
    (commentToWeibo:)];
linkToSearchButton = [self setupButtonWithTitle:@"连接到微博搜索内容流" andSelector:@selector
    (linkToSearch:)];
linkToTimelineButton = [self setupButtonWithTitle:@"连接到我的微博消息流" andSelector:
    @selector(linkToTimeline:)];
linkToProfileButton = [self setupButtonWithTitle:@"连接到我的微博个人主页" andSelector:
    @selector(linkToProfile:)];
```

注：以上场景均可在Demo源码中找到相应代码片段

7.SDK分享多图，视频修改，新增SDK分享单图，视频到story

原有WBMessageObject对象新增WBNewVideoObject属性，表示分享时的视频对象使用新版分享由于使用了相册作为载体，存在异步通信，所以需要第三方开发者实现WBMediaTransferProtocol这个协议，并且在SDK写入图片或视频时应该给用户等待的提示（加载界面），协议目前只有两种回调：

原有图片对象WBImageObject新增两个方法

```
555
556 /**
557  图片对象添加图片数组
558  */
559 - (void)addImages:(NSArray<UIImage *>*)imageArray;
560
561 /**
562  图片对象添加照片数组
563  */
564 - (void)addImageAssets:(NSArray<PHAsset *>*)assetArray;
565
```

以上方法是为新图片对象添加内容（ps：这版旧的图片数据还留着，两种都有的话，新版微博优先识别新的数据）。

新增的WBNewVideoObject对象，也有两个添加内容的方法

第一个方法的参数需要是一个视频的本地URL

```
/**
 视频对象添加视频
 */
-(void)addVideo:(NSURL*)videoUrl;

/**
 视频对象添加视频资源
 */
-(void)addVideoAsset:(PHAsset*)videoAsset;
```

WBNewVideoObject和WBImageObject都有一个isShareToStory的属性，设为YES后表示分享到story。

图片，多媒体，视频对象两两不能共存，文字和多媒体对象无法分享到story，如果和图片或视频分享到story时会被丢弃，没有图片或者视频，默认会呼起发布器分享。

具体调用示例参见demo