

微博 iOS 平台 SDK 文档

编号: WEIBO_IOS_SDK

版本: WEIBO_IOS_SDK V3.0.0

修订记录:

WEIBO iOS SDK 文档
Weibo SDK.....	2
一、 SDK 接入设置 (请仔细阅读本节)	2
二、 应用场景代码示例.....	7

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	邱文杰	SDK 升级到 3.0 版本

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： 该名称也用于来源显示，不超过10个汉字或20个字母

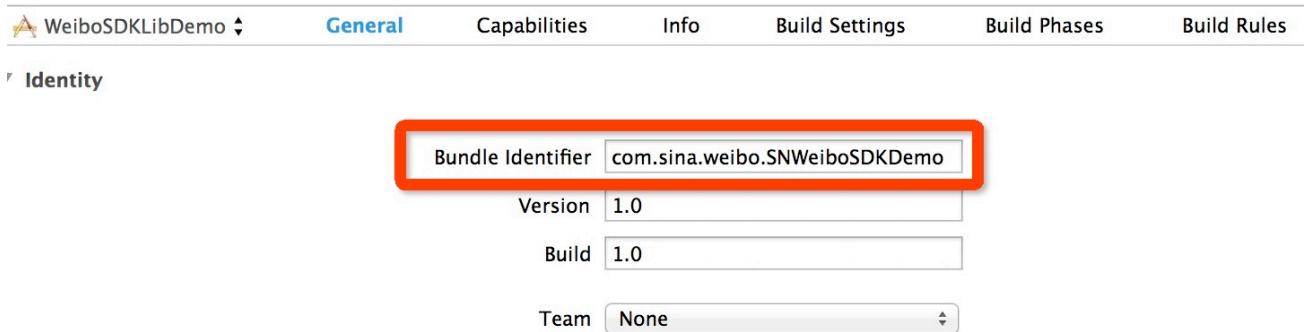
应用平台： [查看移动客户端接入指南](#)

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

▼ URL Types (1)

com.weibo	Identifier <input type="text" value="com.weibo"/>	URL Schemes <input type="text" value="wb2045436852"/>
	Icon <input type="text" value="None"/>	Role <input type="text" value="Editor"/>
▼ Additional url type properties (0)		
Key	Type	Value
Click here to add additional url type properties		

5) 添加 SDK 文件到工程

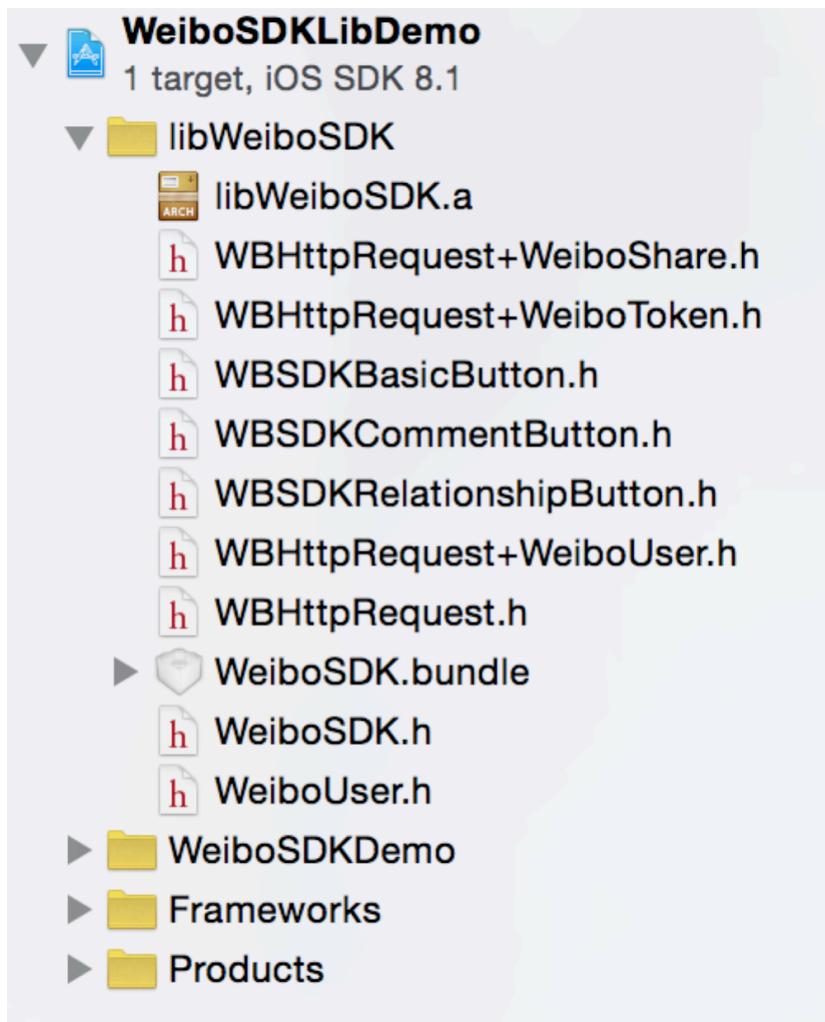
将从 GitHub 上下载的 libWeiboSDK 文件夹添加至工程，其中包含 WeiboSDK.h、

WeiboUser.h 、 WBHttpRequest.h 、 WBHttpRequest+WeiboUser.h 、

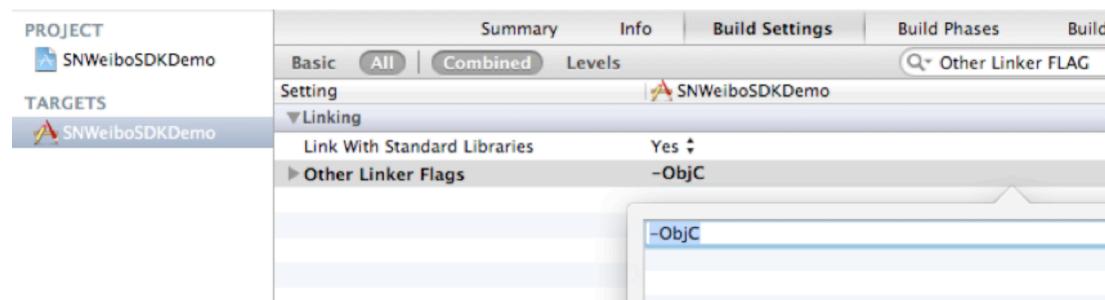
WBHttpRequest+WeiboShare.h 、 WBHttpRequest+WeiboToken.h 、

WBSDKBasicButton.h、 WBSDKRelationshipButton.h、 WBSDKCommentButton.h

这 9 个.h 文件以及 libWeiboSDK.a 和 WeiboSDK.bundle , 统共 11 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->Build Settings->Linking 下 Other Linker Flags 项添加-ObjC。



7) 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"
```

8) 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];  
}
```

9) 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

第三方程序需要实现 `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 didFailWithError:(NSError *)error;

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:@"%@",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:@"%@",error]
                                         delegate:nil
                                    cancelButtonTitle:@"确定"
                                    otherButtonTitles:nil];
    [alert show];
    [alert release];
}

```

注：向好友发送应用邀请、以及请求 OpenApi 为同样使用方法，具体源码见 Demo 源代码。

5、请求 openAPI

调用如下两方法均可：

```

+ (void)requestWithURL:(NSString *)url
                    httpMethod:(NSString *)httpMethod
                    params:(NSDictionary *)params
delegate:(id<WBHttpRequestDelegate>)delegate;

```

```

+ (void)requestWithAccessToken:(NSString *)accessToken
                        url:(NSString *)url
                    httpMethod:(NSString *)httpMethod
                    params:(NSDictionary *)params
delegate:(id<WBHttpRequestDelegate>)delegate;

```

参数说明如下：

@param accessToken 应用获取到的 accessToken , 用于身份验证
 @param url 请求 url 地址
 @param httpMethod 支持"GET" "POST"
 @param params 向接口传递的参数结构
 @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

应用同样可实现 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 ,

此外,为了方便使用,SDK 还封装了部分请求 Open API 请求,范例方法如下: +

```
(WBHttpRequest *)requestForFriendsListOfUser:(NSString*)currentUserID  
withAccessToken:(NSString*)accessToken  
andOtherProperties:(NSDictionary*)otherProperties  
queue:(NSOperationQueue*)queue  
withCompletionHandler:(WBRequestHandler)handler;
```

这个请求调用 Open API 的 friendships/friends 接口,接口返回相应的关注人列表。

参数说明如下:

@param currentUserID 当前授权用户所对应的 UserID.

@param accessToken 应用获取到的 accessToken, 用于身份验证

@param otherProperties 向接口传递的额外参数,

@param queue 发起请求的线程, 默认为主线程,

@param handler 用于接收接口请求的请求的响应 Block。

```
/*
 *!
 * @method
 *
 * @abstract
 * Creates a request representing a Open API call to the "friendships/friends".
 *
 * @discussion
 * Simplifies preparing a request and sending request to retrieve the user's friends.
 *
 * A successful Open API call will return an NSDictionary of objects which contains an array of <WeiboUser> objects representing the user's friends.
 *
 * You can see more details about this API in http://open.weibo.com/wiki/2/friendships/friends/en
 *
 * @param currentUserID      should be the current User's UserID which has been authorized.
 * @param accessToken        The token string.
 * @param otherProperties    Any additional properties for the Open API Request.
 * @param queue              specify the queue that you want to send request on, if this param is nil, the request will be start on MainQueue( [NSOperationQueue mainQueue] ).
 * @param handler            the completion block which will be executed after received response from Open API server.
 */
+ (WBHttpRequest *)requestForFriendsListofUser:(NSString*)currentUserID
                                         withAccessToken:(NSString*)accessToken
                                         andOtherProperties:(NSDictionary*)otherProperties
                                         queue:(NSOperationQueue*)queue
                                         withCompletionHandler:(WBRequestHandler)handler;
```

比如:

```
+ (WBHttpRequest*)requestForFollowersListOfUser:(NSString*)currentUserI
```

D

```
withAccessToken:(NSString*)accessToken
```

```
andOtherProperties:(NSDictionary*)otherProperties
```

```
queue:(NSOperationQueue*)queue
```

```
withCompletionHandler:(WBRequestHandler)handler;
```

这个请求调用 Open API 的 friendships/followers 接口, 接口返回相应的粉丝列表。

参数说明如下:

@param currentUserID 当前授权用户所对应的 UserID.

@param accessToken 应用获取到的 accessToken, 用于身份验证

@param otherProperties 向接口传递的额外参数,

@param queue 发起请求的线程, 默认为主线程,

@param handler 用于接收接口请求的请求的响应 Block。

你可以在 WBHttpRequest+WeiboUser.h、WBHttpRequest+WeiboShare.h、

WBHttpRequest+WeiboToken.h 里找到类似上述两个例子的接口请求详细说明

和使用方法。

```
/*!
@method

@brief
Creates a request representing a Open API call to the "friendships/followers".

@discussion
Simplifies preparing a request and sending request to retrieve the user's followers.

A successful Open API call will return an NSDictionary of objects which contains an array of <WeiboUser> objects representing the user's followers.

You can see more details about this API in http://open.weibo.com/wiki/2/friendships/followers/en

@param currentUserID      should be the current User's UserID which has been authorized.
@param accessToken        The token string.
@param otherProperties    Any additional properties for the Open API Request.
@param queue              specify the queue that you want to send request on, if this param is nil, the request will be start on MainQueue( [NSOperationQueue mainQueue] ).
@param handler            the completion block which will be executed after received response from Open API server.
*/
+ (WBHttpRequest *)requestForFollowersListOfUser:(NSString*)currentUserID
                                         withAccessToken:(NSString*)accessToken
                                         andOtherProperties:(NSDictionary*)otherProperties
                                         queue:(NSOperationQueue*)queue
                                         withCompletionHandler:(WBRequestHandler)handler;
```

6、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 WeiboSDKGetAidSuccessNotification 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。

7、社会化评论组件

WBSDKCommentButton.h 中有如下方法：

```
- (id)initWithFrame:(CGRect)frame  
    accessToken:(NSString*)accessToken  
    keyword:(NSString*)keyWord  
    urlString:(NSString*)urlString  
    category:(NSString*)category  
    completionHandler:(WBSDKButtonHandler)handler;
```

这个方法的作用是初始化一个社会化评论按钮。

参数说明如下

@param frame 按钮的 frame 值

@param accessToken 用户授权后获取的 Token

@param keyWord 社会化评论的热点词

@param urlString 社会化评论链接，可传空

@param category 领域 ID, 此参数为必选参数。

@param handler 回调函数，当用户点击按钮，进行完与社会化评论组件相关的交互之后，回调的函数。

```
@interface WBSDKCommentButton : WBSDKBasicButton

/**
 * 初始化一个社会化评论按钮
 * @param frame 按钮的frame值
 * @param accessToken 用户授权后获取的Token
 * @param keyWord 社会化评论的热点词
 * @param urlString 社会化评论链接，可传空
 * @param category 领域ID，此参数为必选参数。
 * @param handler 回调函数，当用户点击按钮，进行完与社会化评论组件相关的交互之后，回调的函数。
 */
- (id)initWithFrame:(CGRect)frame
    accessToken:(NSString*)accessToken
    keyword:(NSString*)keyWord
    urlString:(NSString*)urlString
    category:(NSString*)category
    completionHandler:(WBSDKButtonHandler)handler;

@property (nonatomic, retain)NSString* keyWord;
@property (nonatomic, retain)NSString* accessToken;
@property (nonatomic, retain)NSString* urlString;
@property (nonatomic, retain)NSString* category;

@end
```

8、关注组件

WBSDKRelationshipButton.h 中有如下方法：

```
- (id)initWithFrame:(CGRect)frame
    accessToken:(NSString*)accessToken
    currentUser:(NSString*)currentUserID
    followUser:(NSString*)followerUserID
    completionHandler:(WBSDKButtonHandler)handler;
```

这个方法的作用是初始化一个关注组件按钮

参数说明如下

@param frame 按钮的 frame 值
@param accessToken 用户授权后获取的 Token
@param currentUserID 当前用户的 uid 值
@param followerUserID 希望当前用户加关注的用户 uid 值
@param handler 回调函数，当用户点击按钮，进行完关注组件相关的交互之后，

回调的函数。

```
@interface WBSDKRelationshipButton : WBSDKBasicButton

/**
 * 初始化一个关注组件按钮
 * @param frame 按钮的frame值
 * @param accessToken 用户授权后获取的Token
 * @param currentUserID 当前用户的uid值
 * @param followerUserID 希望当前用户加关注的用户uid值
 * @param handler 回调函数，当用户点击按钮，进行完关注组件相关的交互之后，回调的函数。
 */
- (id)initWithFrame:(CGRect)frame
    accessToken:(NSString*)accessToken
    currentUser:(NSString*)currentUserID
    followUser:(NSString*)followerUserID
    completionHandler:(WBSDKButtonHandler)handler;

@property (nonatomic, retain)NSString* accessToken;
@property (nonatomic, retain)NSString* currentUserID;
@property (nonatomic, retain)NSString* followerUserID;

@property (nonatomic, assign)WBSDKRelationshipButtonState currentRelationShip;

/**
 * 获取最新的关注状态
 * 该方法会调用OpenApi，获取当前用户与目标用户之间的关注状态，并将按钮的状态改变为正确的状态。
 */
- (void)checkCurrentRelationship;

@end
```

注：以上场景均可在 Demo 源码中找到相应代码片段