

目录

一、产品特点.....	2
二、SDK接口介绍	2
Renren接口说明	2
RenrenPay接口说明	4
三、基本功能使用说明.....	5
使用iOS SDK的准备工作	5
使用iOS SDK提供的页面授权方法	6
使用iOS SDK提供的用户名密码授权方式	7
使用iOS SDK调用人人API接口 （以取得用户信息为例）	7
使用iOS SDK完成一键上传照片	8
使用iOS SDK提供的widget dialog （以发布状态为例）	9
使用iOS SDK退出授权状态	10
四、支付功能使用说明.....	10
取得人人支付的对象.....	10
支付流程.....	11
查询本地保存的订单.....	12
删除本地保存的订单.....	12
使用测试接口	12
校验订单.....	13
五、功能示范.....	13
一键上传图片	13
使用常用功能接口查询当前登录用户信息	13
使用通用API接口发布状态	14
使用Widget Dialog发布自定义新鲜事.....	14
使用支付接口.....	15

人人网开放平台 iOS SDK V3.0

一、产品特点


- 支付功能：SDK 提供支付、充值、本地订单查询、本地订单修复和本地订单删除等接口支持，用户消费一气呵成。
- Navigation 页面：本 SDK 新增了 Navigation 形式的页面展示，使得 SDK 页面与你的应用契合度更高。
- 弹层页面改善：弹层页面添加关闭“X”按钮，修复页面加载失败时无法关闭页面的问题。
- 保障应用的信息安全，去除 secret 宏定义，需要使用 secret 的接口，由开发者传入 secret。

二、SDK 接口介绍

Renren 接口说明

表 2-1 接口方法

序号	类型	功能	接口	状态
1	登录、授权与登出	授权页面方式登录——弹层页面	-(void)authorizationWithPermissions:(NSArray *)permissions andDelegate:(id<RenrenDelegate>)delegate	原有
2		授权页面方式登录——Navigation 页面	-(void)authorizationInNavigationWithPermissions:(NSArray *)permissions andDelegate:(id<RenrenDelegate>)delegate;	新增
3		用户名密码方式登录	-(void)passwordFlowAuthorizationWithParam:(RORPasswordFlowRequestParam *)param andDelegate:(id<RenrenDelegate>)delegate;	原有
4		登出	-(void)logout:(id<RenrenDelegate>)delegate;	原有
5	一键发布	一键上传照片	-(void)publishPhotoSimplyWithI	原有

			 image caption: (NSString*)caption;	
6	常用功能	获取用户信息	-(void) getUsersInfo: (ROUserInfoRequestParam *)param andDelegate: (id<RenrenDelegate>) delegate;	原有
7		获取好友列表	-(void) getFriends: (ROGetFriendsRequestParam *)param andDelegate: (id<RenrenDelegate>) delegate;	原有
8		获取好友详细信息列表	-(void) getFriendsInfo: (ROGetFriendsInfoRequestParam *)param andDelegate: (id<RenrenDelegate>) delegate;	原有
9		获取相册信息	-(void) getAlbums: (ROAlbumsInfoRequestParam *)param andDelegate: (id<RenrenDelegate>) delegate;	原有
10		上传照片	-(void) uploadPhoto: (ROUploadPhotoRequestParam *)param andDelegate: (id<RenrenDelegate>) delegate;	原有
11		新建相册	-(void) createAlbum: (ROCreateAlbumRequestParam *)param andDelegate: (id<RenrenDelegate>) delegate;	原有
12	通用方法	通用 API 接口调用	-(RORequest *) requestWithParams: (NSMutableDictionary *)params andDelegate: (id<RenrenDelegate>) delegate;	原有
13		Widget Dialog 调用——弹层页面	-(void) dialog: (NSString *)action andParams: (NSMutableDictionary *)params andDelegate: (id<RODialogDelegate>) delegate;	原有
14		Widget Dialog 调用——Navigation 页面	-(void) dialogInNavigation: (NSString *)action andParams: (NSMutableDictionary *)params andDelegate: (id<RenrenDelegate>) delegate;	新增
15		判断用户登录后授权的	-(BOOL) isSessionValid;	原有

		周期是否有效		
16	扩展功能	获得人人支付功能对象 RenrenPay	-(RenrenPay *)getRenrenPayWithSecret:(NSString *)secret andLocalMem:(BOOL)isUsed;	新增

表 2-2 代理方法

序号	代理方法	说明	状态
1	-(void)renren:(Renren *)renrenrequestDidReturnResponse:(ROResponse*)response;	API 接口请求成功时调用，返回服务器给出的请求结果数据对象。（一键上传和登录登出相关功能不适用）	原有
2	-(void)renren:(Renren *)renrenrequestFailWithError:(ROResponse*)error;	API 接口请求失败时调用，返回表示错误原因的错误对象。（一键上传和登录登出相关功能不适用）	原有
3	-(void)renrenDialogDidCancel:(Renren *)renren;	Dialog 页面被用户或 Renren 对象取消并关闭时调用。（仅 Dialog 功能适用）	原有
4	-(void)renrenDidLogin:(Renren*)renren;	登录成功时调用。（仅登录相关功能适用）	原有
5	-(void)renrenDidLogout:(Renren*)renren;	登出成功时调用。（仅登出功能适用）	原有
6	-(void)renren:(Renren *)renrenloginFailWithError:(ROResponse*)error;	登录失败时调用，返回 返回表示错误原因的错误对象。（仅登录相关功能适用）	原有

RenrenPay 接口说明

表 2-3 接口方法

序号	类型	功能	接口	状态
1	支付	获取订单号	-(NSString*)getOrderNumber;	新增
2		生成订单	-(ROResponse*)makePayOrderWithOrderNum:(NSString *)orderNum andAmount:(NSInteger)amount andDescription:(NSString *)description andPayment:(NSString*)payment;	新增
3		提交订单——弹层页面	-(void)submitPayOrderWithOrder:(ROResponse*)order andPermissions:(NSArray*)permissions;	新增

			ssions andDelegate: (id<RenrenPayDeleg ate>)) delegate;	
4		提交订单—— Navigation 页面	-(void) submitPayOrderInNavigat ionWithOrder: (ROPayOrderInfo*) order andPermissions: (NSArray*) permi ssions andDelegate: (id<RenrenPayDeleg ate>)) delegate;	新增
5	查询订单	查询订单——弹层页面	-(void) queryOrderListWithDeleg ate: (id<RenrenPayDelegate>)) del egate;	新增
6		查询订单—— Navigation 页面	-(void) queryOrderListInNavigat ionWithDelegate: (id<RenrenPayD elegate>)) delegate;	新增
7	删除订单	删除订单	-(void) deleteOrderList;	新增
8	校验订单	校验订单	-(NSString*) getPayResultEncode WithOrder: (ROPayOrderInfo*) ord er andAppPayPassword: (NSString*) p assword;	新增

表 2-4 代理方法

序号	代理方法	说明	状态
1	-(void) payDidSuccessWithOrder: (RO PayOrderInfo*) order;	提交订单，支付成功	新增
2	-(void) payDidFailWithError: (ROPay Error*) error;	提交订单，支付失败	新增
3	-(void) repairOrderDidSuccessWithO rder: (ROPayOrderInfo*) order;	订单修复成功	新增
4	-(void) repairOrderDidFailWithErro r: (ROPayError*) error;	订单修复失败	新增

三、基本功能使用说明

使用 iOS SDK 的准备工作

1. 请将 JSON，FMDB 和 Renren 三个文件夹复制到你的 Xcode 工程下

2. 在 Xcode 下右键点击你的 Xcode 工程代码 Group，从弹出的菜单中选择 Add -> Existing Files，加入前面提到过的 3 个文件夹中的代码。

3. 需要设置包含路径，请在工程的 XXX-Prefix.pch 文件中引入 ROConnect.h 文件

```
#import "ROConnect.h"
```

4. 为工程添加 libsqlite3.0.dylib 的静态库。

使用 iOS SDK 提供的页面授权方法

1. 首先，打开 Renren 文件夹下找到 Renren.h 文件：

```
#define kAPP_ID      @"YOUR APP ID"  
#define kAPI_Key     @"YOUR API KEY"
```

将上面宏定义的内容配置为你在人人开发平台上申请的应用信息。

2. 实例化一个 Renren 对象，你可以在应用程序委托 application:didFinishLaunchingWithOptions: 方法或是在视图加载的 viewDidLoad 方法时，创建一个 Renren 对象。

```
Renren *renren = [Renren sharedRenren];
```

3. [renren isSessionValid]

这个方法可以判断当前用户是否登录状态

4. [renren authorizationWithPermisson:nil andDelegate:self];

通过这个方法，用户可以进行登录验证授权

authorizationWithPermisson:[授予权限] 如 publish_feed photo_upload 等

参数默认可以不写，如有多个权限的话，可以这样

```
NSArray *permissions=[NSArray arrayWithObjects:@"photo_upload", "publish_feed",nil];  
[renren authorizationWithPermisson:permissions andDelegate:self];
```

5. 你还需要在你的应用中实现 Renren 的代理 RenrenDelegate，主要实现下面两个方法：

```
/**  
 * 授权登录成功时被调用，第三方开发者实现这个方法  
 * @param renren 传回代理授权登录接口请求的 Renren 类型对象。  
 */  
- (void)renrenDidLogin:(Renren *)renren;  
  
/**
```

```
* 授权登录失败时被调用，第三方开发者实现这个方法  
* @param renren 传回代理授权登录接口请求的 Renren 类型对象。  
*/  
- (void)renren:(Renren *)renren loginFailWithError:(ROError*)error;
```

使用 iOS SDK 提供的用户名密码授权方式

1. 请参照《使用 iOS SDK 提供的页面授权方式》的 1, 2, 3 步

2. 创建 ROPasswordFlowRequestParam 的对象并初始化

```
ROPasswordFlowRequestParam *requestParam = [[[ROPasswordFlowRequestParam alloc]  
init] autorelease];
```

设置 requestParam 中的 userName 和 passWord 属性，如下

```
requestParam.userName = @"user";  
requestParam.passWord = @"123456";  
requestParam.secretKey = @"Your APP Secret";
```

设置权限的种类(只获得默认权限可以不用设置此属性)

```
requestParam.scope = @"operate_like";
```

调用下面的方法完成授权

```
[renren passwordFlowAuthorizationWithParam:requestParam andDelegate:self];
```

3. 实现 RenrenDelegate 中代理方法，完成授权成功和失败的处理，可以参照《使用 iOS SDK 提供的页面授权方式》的第 5 步

使用 iOS SDK 调用人人 API 接口（以取得用户信息为例）

1. 请参照《使用 iOS SDK 提供的页面授权方式》的 1, 2。

2. 完成授权后，创建 RUserInfoRequestParam 的对象并初始化

```
RUserInfoRequestParam *requestParam = [[[RUserInfoRequestParam alloc] init]  
autorelease];
```

3. 设置 requestParam 中的 fields(返回用户那些字段信息)属性，如下

```
requestParam.fields = @"uid,name,sex";
```

调用下面的方法完成授权

```
[renren getUserInfo:requestParam andDelegate:self];
```

4. 需要实现 RenrenDelegate 中代理方法

```
/**
 * 接口请求成功，第三方开发者实现这个方法
 * @param renren 传回代理服务器接口请求的 Renren 类型对象。
 * @param response 传回接口请求的响应。
 */
- (void)renren:(Renren *)renren requestDidReturnResponse:(ROResponse*)response;

/**
 * 接口请求失败，第三方开发者实现这个方法
 * @param renren 传回代理服务器接口请求的 Renren 类型对象。
 * @param response 传回接口请求的错误对象。
 */
- (void)renren:(Renren *)renren requestFailWithError:(ROError*)error;
```

使用 iOS SDK 完成一键上传照片

1. 请参照《使用 iOS SDK 提供的页面授权方式》的 1, 2。

2. 完成授权后，创建 Image 的对象和描述，

```
UIImage* image = [UIImage imageNamed:@"test.png"];
NSString *caption = @"这是一张测试图片";
```

3. `[renren publishPhotoSimplyWithImage:image caption:caption];`

调用上面的方法即可完成上传照片的功能。

4. 需要实现 RenrenDelegate 中代理方法

```
/**
 * 接口请求成功，第三方开发者实现这个方法
 * @param renren 传回代理服务器接口请求的 Renren 类型对象。
 * @param response 传回接口请求的响应。
 */
- (void)renren:(Renren *)renren requestDidReturnResponse:(ROResponse*)response;

/**
 * 接口请求失败，第三方开发者实现这个方法
 * @param renren 传回代理服务器接口请求的 Renren 类型对象。
 */
```



```
* @param response 传回接口请求的错误对象。  
*/  
- (void)renren:(Renren *)renren requestFailWithError:(ROError*)error;
```

使用 iOS SDK 提供的 widget dialog （以发布状态为例）

1. 请参照《使用 iOS SDK 提供的页面授权方式》的 1，2。

2. 完成授权后，创建请求 widget dialog 的字典

```
NSMutableDictionary *params=[NSMutableDictionary dictionaryWithObjectsAndKeys:  
    @"http://dev.renren.com/",@"url",  
    @"人人网开放平台",@"name",  
    @"访问我们",@"action_name",  
    @"http://dev.renren.com/",@"action_link",  
    @"来自人人网 iOS SDK",@"description",  
    @"欢迎使用人人网 SDK !",@"caption",  
  
    @"http://hdn.xnimg.cn/photos/hdn421/20090923/1935/head_1Wmz_19242g019116.jpg",@"i  
mage",  
  
    nil];
```

3. `[renren dialog:@"feed" andParam:params andDelegate:self];`

调用上面的方法即可完成发布状态的功能。

4. 需要实现 RenrenDelegate 中代理方法

```
/**  
 * 接口请求成功，第三方开发者实现这个方法  
 * @param renren 传回代理服务器接口请求的 Renren 类型对象。  
 * @param response 传回接口请求的响应。  
 */  
- (void)renren:(Renren *)renren requestDidReturnResponse:(ROResponse*)response;  
  
/**  
 * 接口请求失败，第三方开发者实现这个方法  
 * @param renren 传回代理服务器接口请求的 Renren 类型对象。  
 * @param response 传回接口请求的错误对象。  
 */  
- (void)renren:(Renren *)renren requestFailWithError:(ROError*)error;
```

使用 iOS SDK 退出授权状态

1 在授权以后，通过调用下面的方法退出授权状态。

```
[renren logout:self];
```

2. 需要实现 RenrenDelegate 中代理方法

```
/**
 * 用户登出成功后被调用 第三方开发者实现这个方法
 * @param renren 传回代理登出接口请求的 Renren 类型对象。
 */
- (void)renrenDidLogout:(Renren *)renren;
```

四、支付功能使用说明

PS: 如果不需要使用支付功能，完全可以略过此节内容。支付功能是人人网本次 sdk 更新的重点。面向广大移动应用开发者，旨在将为众多社交游戏带来丰厚利润的人人豆支付业务带到移动平台上。

注意：人人网提供的支付功能支持开发者有自己的 Web Server 的情况，也支持没有自己的 Web Server 的情况。有 Web Server 情况的服务端配置请参考：

<http://wiki.dev.renren.com/wiki/%E6%A0%A1%E5%86%85%E8%B1%86>

关于支付流程，开发者可参照本文件同目录下的《人人网开放平台安全支付 1.0. pdf》。iOS 开发者可暂时不必关心其中的内容，先看本文档。

前置条件，开发者首先需要申请人人网开放平台应用，开发者申请应用之后，需要申请开通支付功能。

取得人人支付的对象

1. 需要创建 Renren 对象，这部分内容请参考上一章节。

2. 通过调用下面的方法，取得 RenrenPay 的对象，其中传入的参数需要注意一下，

```
RenrenPay renrenPay = [self.renren getRenrenPayWithSecret:
@"4723a695c09e4ddebbe8d87393d95fb4" andLocalMem:YES];
```

- 第一个参数传入的是 APP 的 Secret，这个值在申请人人应用的时候会分配给你，为了保证应用信息的安全，请您妥善保管这个值。
- 第二个参数的意义是——是否使用本地存储功能，如果传入 YES，SDK 会开通本地存储功能，在本地保存订单信息的数据库，并提供订单查询，订单修复及订单删除功能。如果你关闭了本地存储功能，SDK 不会在本地存储任何信息，并关闭订单查询，订单修复及订单删除功能。

支付流程

1. 首先你要生成一个订单号，SDK 中提供了生成订单号的方法，提供默认的订单号。当然，你也可以通过自己的算法生成订单号。使用 SDK 中生成订单号的方法如下所示，

```
NSString *orderNum = [self.renrenPay getOrderNumber];
```

2. 通过调用 SDK 中的方法生成一份订单，并得到订单信息对象。

```
ROPayOrderInfo *order = [self.renrenPay makePayOrderWithOrderNum:orderNum  
andAmount:amountNum  
andDescription:description andPayment:nil];
```

- 其中 orderNum 这个参数是必须传入的参数，如果传入的订单号是 nil 或者@ “” 则无法成功生成订单对象。
 - 另外 Payment 参数是开发者自己定义的，在支付结束后会返回给应用，应用可以利用这个参数进行特定的处理，可以为 nil。
3. 调用下面的接口发起支付流程。如果用户的账户余额大于需要支付的额度，则进行普通的支付流程；如果用户余额小于需要支付的额度，则进行直冲的支付流程，但不管走那个流程，第三方应用都不用处理，人人支付 SDK 会帮你完成。

```
[self.renrenPay submitPayOrderWithOrder:order  
andPermissions:nil  
andDelegate:self];
```

- 第一个参数 order 是在第 2 步中生成的订单对象，如果为 nil 则不能发起支付流程。
- 第二个参数 Permissions 的用途是这样的，如果用户使用支付功能以前没有授权应用，可以通过这个参数将授予的权限传给 SDK，这样可以将授权和支付一步完成。
- 最后一个参数是代理对象。

4. 需要实现 RenrenPayDelegate 中的代理方法

```
/**  
 * 支付成功，第三方开发者实现这个方法  
 * @param order 传回订单详细信息的对象。  
 */  
- (void)payDidSuccessWithOrder:(ROPayOrderInfo*)order;
```

```
/**  
 * 支付失败，第三方开发者实现这个方法  
 * @param error 传回订单支付失败信息的对象。  
 */  
- (void)payDidFailWithError:(ROPayError*)error;
```

查询本地保存的订单

1.调用下面的接口可以查询本地保存的订单状态，这个状态可能和服务器端状态存在出入，特别提供了修复订单的功能给用户使用。查询订单的 UI 和处理过程也不需要第三方应用处理什么，只需要调用支付 SDK 的接口就可以了。

```
[self.renrenPay queryOrderListWithDelegate:self];
```

ps: 用户在未登录状态也可以查询订单，但是查询结果为空。用户登录后只能查询到本账户的订单信息，不能查询其他账户的订单信息。

2.需要实现 RenrenPayDelegate 中的代理方法

```
/**
 * 修复成功，第三方开发者实现这个方法
 * @param order 传回订单详细信息的对象。
 */
- (void)repairOrderDidSuccessWithOrder:(ROPayOrderInfo*)order;

/**
 * 修复失败，第三方开发者实现这个方法
 * @param error 传回订单修复失败信息的对象。
 */
- (void)repairOrderDidFailWithError:(ROPayError*)error;
```

删除本地保存的订单

1.调用下面的接口可以删除本地保存的订单状态，用户只能将当前授权账户的订单信息删除，不能删除其他账户订单

```
[self.renrenPay deleteOrderList];
```

使用测试接口

1.支付功能提供了测试接口调用方式，可以方便广大应用开发者进行测试使用。

```
self.renrenPay.isTest = YES;
```

上面代码设置了 RenrenPay 为测试状态，之后再调用提交订单进行支付，或者进行订单修复操作，就都是走测试流程，不会真的进行消费。如果将 isTest 属性设置为 NO，可以关闭 RenrenPay 的测试状态。

校验订单

1.当订单支付成功或者订单校验成功后，SDK 会通过代理方法将订单对象返回给开发者。开发者可以调用下面的接口计算订单校验码，如果与订单中的校验码不相等，可以视作伪造的订单不予处理。示例代码如下，

```
if (![order.payEncode isEqualToString:[self.renrenPay getPayResultEncodeWithOrder:order  
andAppPayPassword:@"123456"]])  
{  
    //无效订单的处理  
}
```

五、功能示范

一键上传图片

实现代码见Renren SDK Demo中的“ 一键上传图片”功能。



使用常用功能接口查询当前登录用户信息

实现代码见 Renren SDK Demo 中的“当前用户信息”功能。



使用通用 API 接口发布状态

实现代码见Renren SDK Demo中的“发布状态”功能。



使用 Widget Dialog 发布自定义新鲜事

实现代码见Renren SDK Demo中的“Feed Dialog”功能



使用支付接口

提交订单





查询订单



人人网 iOS SDK 负责人: 夏文海
E-mail: wenhai.xia@renren-inc.com
QQ: 86852962