

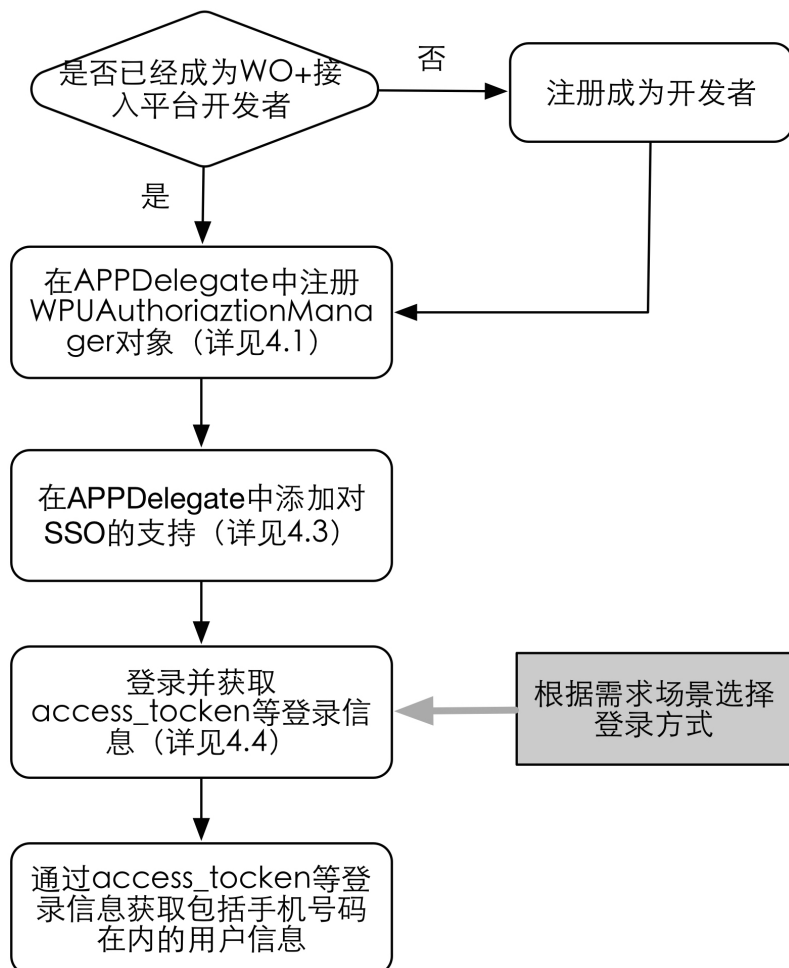
# 沃通行证iOS平台SDK文档

版本：1.0.2

## 一、名词解释

- 1) NET取号：通过运营商移动网络自动取号
- 2) 隐式登录：SDK在后台进行NET取号，取到手机号后以alert形式弹出loading，并进行授权，授权成功后loading消失，用户不参与交互。若取号失败则不进行弹窗
- 3) 一键登录：SDK在后台进行NET取号，取到手机号后以alert形式弹出授权页面，用户确认登录后授权成功。若取号失败则不进行弹窗
- 4) 验证登录：SDK在后台进行NET取号，不论取号成功与否，登录界面都将以modal的形式推出
- 5) client\_id：第三方申请到的AppKey
- 6) client\_secret：第三方申请到的AppSecret
- 7) redirect\_uri：第三方申请时填写的回调地址

## 二、SDK 接入流程



## 三、SDK 接入设置

### 1) 注册成为开发者，创建应用



### 2) 请按要求填写应用基本信息

管理中心/ 创建应用

1

填写基本信息

2

填写平台信息

3

提交成功

应用名称

请注意，应用审核通过后，应用名称不允许修改。

应用类型

请选择

应用简介

最多80字

应用图片

请上传应用水印图片  
30\*30像素，2M以内，支持PNG、JPG。

图片预览

上传

### 3) 填写平台信息

其中信任IP为开发者受信任的服务器IP地址

1

填写基本信息

2

填写平台信息

3

提交成功

应用官网

https://developer.wo.cn,http://www.unisk.cn/,http://127.0.0.1:8080/应用名称等

应用H5地址

https://developer.wo.cn,http://www.unisk.cn/,http://127.0.0.1:8080/应用名称等

信任IP

删除

添加IP

应用回调地址

https://developer.wo.cn,http://www.unisk.cn/,http://127.0.0.1:8080/应用名称等

#### 4) 设定Bundle ID 和Scheme

请在此页面中填写AppStore下载地址、Bundle ID 、Scheme。这样您的应用才能正常使用沃通行证授权和回调。

注：AppStore下载地址如果没有，可不填写。Bundle ID 和Scheme应和xcode中的保持一致，如图：（URL Scheme的填写需保证完整，如wopass，填写时需加上://，即：wopass://）

应用平台

请至少选择一个平台。若应用还未提交到应用市场，请先勾选需要的平台，将下载地址等暂时置空，待审核通过、开发完毕，提交到各平台应用市场后，再修改应用的下载地址为正确的地址。

☒ iOS 应用

AppStore下载地址(选填)

https://developer.wo.cn,http://www.unisk.cn/,http://127.0.0.1:8080/应用名称等

Apple AppStore中的下载地址，如应用还未上线，可置空，待应用上线后再行修改，地址格式：https://developer.wo.cn,http://www.unisk.cn/,http://127.0.0.1:8080/应用名称等

Bundle ID


IOS应用唯一标识

Scheme

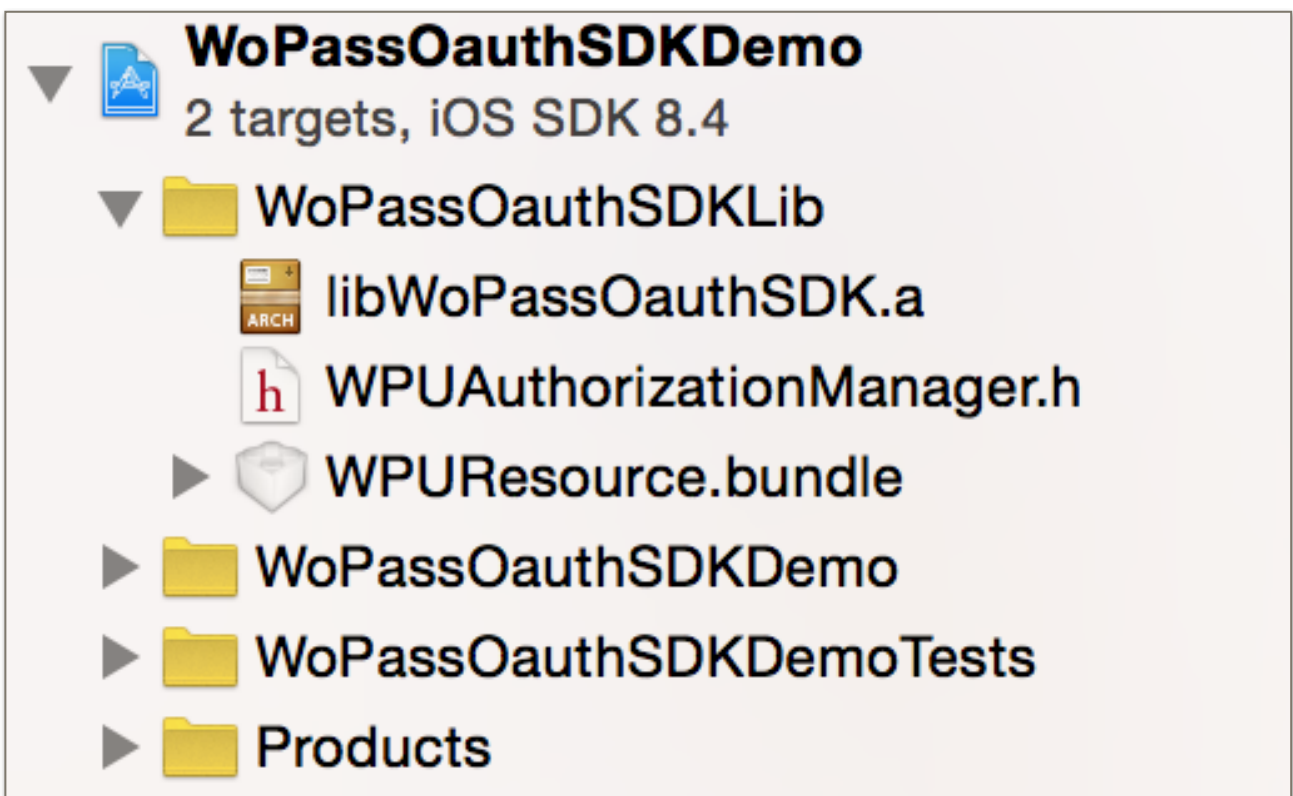
方便APP之间互相调用，通过系统的OpenURI来打开该app，并可以传递一些参数；每个URL必须能唯一标识一个APP。

☐ Android 应用

General	Capabilities	Info	Build Settings	Build Phases	Build Rules
<p>▼ Identity</p> <p>Bundle Identifier <input type="text" value="unisk.cn.woPass"/></p> <p>Version <input type="text" value="3.0.0"/></p> <p>Build <input type="text" value="3.0.2"/></p> <p>Team <input type="text" value="None"/></p>					

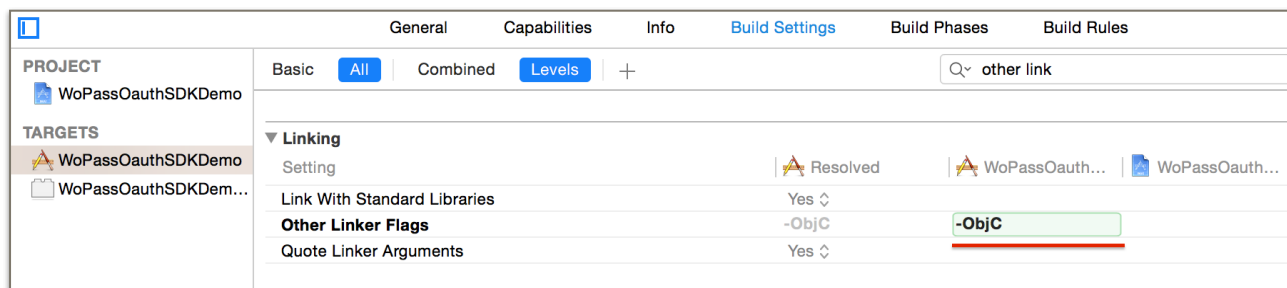
General	Capabilities	Info	Build Settings	Build Phases	Build Rules
<p>► Exported UTIs (0)</p> <p>► Imported UTIs (0)</p> <p>▼ URL Types (3)</p> <p>unisk.cn.woPass</p> <div>  <div> <p>Identifier <input type="text" value="unisk.cn.woPass"/></p> <p>Icon <input type="text" value="None"/></p> </div> <div> <p>URL Schemes <input type="text" value="wopass"/></p> <p>Role <input type="text" value="Editor"/></p> </div> </div>					

5) 将SDK文件添加到工程



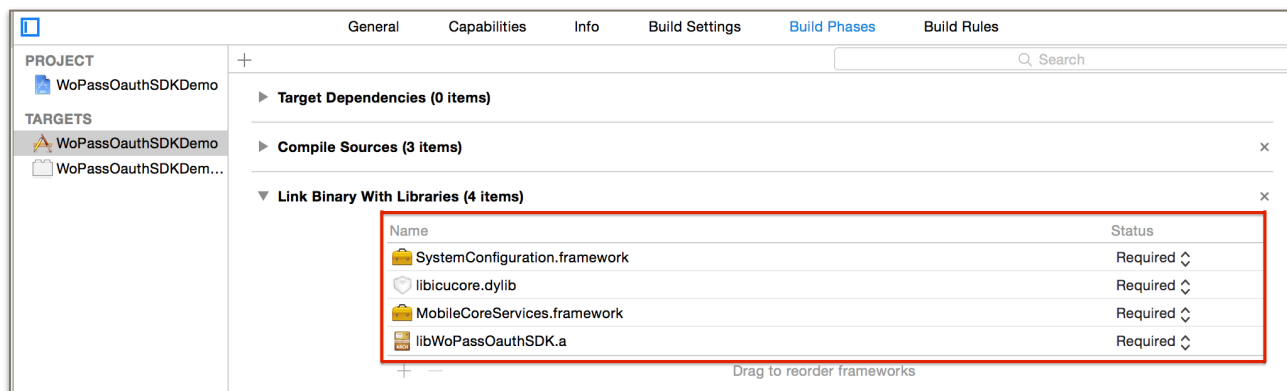
将从开放平台网站上下载的libwoPassSDK文件夹添加至工程，其中包含WPUAuthorizationManager.h、libWoPassOauthSDK.a以及WPUResource.bundle文件。

6) 在工程中引入静态库后，需要在编译时添加-ObjC编译选项，避免静态库加载不全造成运行时崩溃。方法：程序Target -> Build Settings->Linking 下 Other Linker Flags项添加-ObjC。



7) 添加依赖库至工程

将依赖库添加至工程，方法：程序Target -> Build Phase -> Link Binary With Libraries 下添加以下FrameWork至工程中： MobileCoreServices.framework、libcucore.dylib、SystemConfiguration.framework



## 四、SDK 集成代码示例

1) 在AppDelegate.m中引入WPUAuthorizationManager.h头文件，并添加属性authManger

```
#import "AppDelegate.h"
#import "WPUAuthorizationManager.h"
```

```
@interface AppDelegate ()

@property (nonatomic, strong)WPUAuthorizationManager *authManager;

@end
```

2) 在AppDelegate的application:didFinishLaunchingWithOptions:方法中加入以下代码

对于授权码模式:

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

    // 授权码模式
    self.authManager = [WPUAuthorizationManager sharedManagerWithParams:@{

        @"client_id" : @"",

        @"redirect_uri" :
        @"http://www.anywhere",

        @"client_secret" : @"",
    }

    options:WPUAuthCode
    delegate:self];

    return YES;
}
```

对于简化模式:

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

    // 简化模式
    self.authManager = [WPUAuthorizationManager sharedManagerWithParams:@{

        @"client_id" : @"",

        @"redirect_uri" :
        @"http://www.anywhere",
    }

    options:WPUAuthImplicit
    delegate:self];

    return YES;
}
```

注意: 不建议使用授权码模式, 因为client\_secret 暴露会引起安全问题。容易反编译导致client\_secret 泄露。

参数说明:

- \* @param params 授权参数列表。其中:  
client\_id : 申请时获得的AppKey

client\_secret : 申请时获得的AppSecret  
redirect\_uri : 创建应用时填写的应用回调地址

\* @param options 授权模式选项。  
简化模式不支持过期自动刷新，可通过WPUAuthCode | WPUAuthAutoRefresh，来设置token过期自动刷新。

\* @param delegate 代理

3) 若需使用沃通行证的SSO功能，在AppDelegate的中加入以下代码

```
- (BOOL)application:(UIApplication *)application handleOpenURL:(NSURL *)url {  
    [self.authManager handleURL:url];  
    return YES;  
}  
  
- (BOOL)application:(UIApplication *)application openURL:(NSURL *)url  
    sourceApplication:(NSString *)sourceApplication annotation:(id)  
    annotation {  
    [self.authManager handleURL:url];  
    return YES;  
}  
  
- (BOOL)application:(UIApplication *)app openURL:(NSURL *)url options:  
    (NSDictionary<NSString *,id> *)options {  
    [self.authManager handleURL:url];  
    return YES;  
}
```

4) 获取用户授权，可通过调用如下两个方法来获取用户授权

1、验证授权方式：调用此方法后，授权界面将在加载完毕后以modal的方式推出

```
- (void>manualAuthorizationWithCompleted:(WPUAuthorizeCompletedAction)  
    authorizeCompletedAction andLoginPageWillAppearAction:  
    (WPULoginPageWillAppearAction) loginPageWillAppearAction  
    accessPhoneNumberNeeded:(BOOL) isNeeded;
```

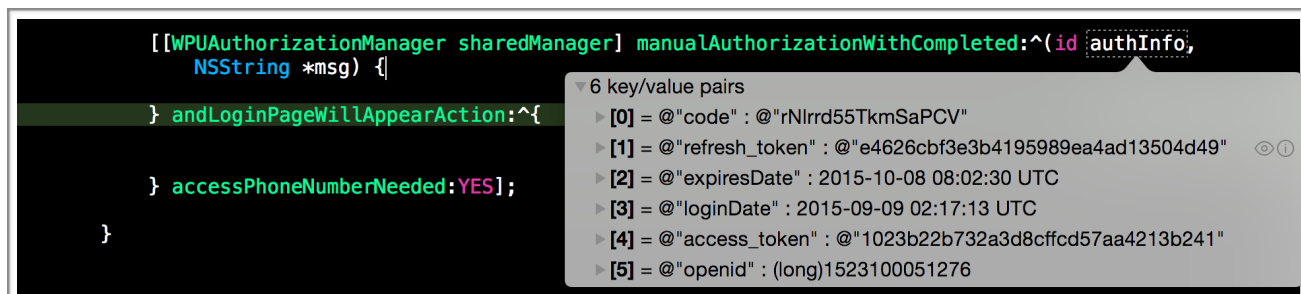
参数说明：

\* @param authorizeCompletedAction  
\* @param loginPageWillAppearAction

授权完成的回调  
授权界面即将推出的回调

\* @param isNeeded isNeeded = YES: 如果使用的是联通蜂窝网络, SDK将从网络层获取用户手机号码, 如果使用的是非联通蜂窝网络, SDK将不进行此项操作; isNeeded = NO: SDK将不进行取号操作。

调用及授权成功后的授权信息如下图:



授权信息authInfo以NSDictionary的形式给出, 共计6个键值对。若使用的是简化模式, 将不返回code字段; msg为状态信息, 如果授权成功将返回“成功”

2、按级别授权方式: 调用此方法后, 授权形式于应用的安全级别有关。应用的安全等级共分为0、1、2三个级别。

应用安全级别与授权方式的对应关系如下所示:

安全级别为0: 应用的授权形式为隐式登录授权

安全级别为1: 应用的授权形式为一键登录授权

安全级别为2: 应用的授权形式为验证登录授权

```
- (void)autoAuthorizationWithCompleted:(WPUAuthorizeCompletedAction)
authorizeCompletedAction andLoginPageWillAppearAction:
(WPULoginPageWillAppearAction) loginPageWillAppearAction
delayTimeInterval:(NSInteger)timeInterval;
```

参数说明:

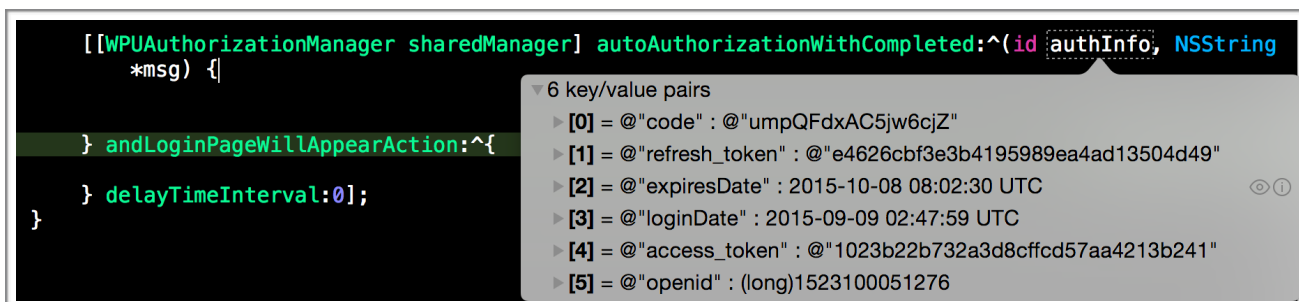
- \* @param authorizeCompletedAction
- \* @param loginPageWillAppearAction
- \* @param timeInterval

授权完成的回调

授权界面即将弹出的回调

在timeInterval间隔后调用此方法

调用及授权成功后的授权信息如下图:





授权信息authInfo以NSDictionary的形式给出，共计6个键值对。若使用的是简化模式，将不返回code字段；  
msg为状态信息，如果授权成功将返回“成功”

若设置了option为WPUAuthCode | WPUAuthAutoRefresh，则在token过期时调用这两个方法将会刷新token，不再弹出授权界面。

#### 5) 获取用户信息

可通过调用如下方法来获取用户信息

```
- (void)getUserDataWithCompletedAction:(WPUGetUserDataCompletedAction)
    getUserDataCompletedAction;
```

参数说明：

\* @param getUserDataCompletedAction 用户信息获取完毕的回调

调用及成功后的用户信息如下图：

```
[[WPUAuthorizationManager sharedManager]
    getUserDataWithCompletedAction:^(id userData, id authInfo,
    NSString *msg) {

}];
```

userData为用户信息，将以NSDictionary的形式给出，共计6个键值对，包含：

avatar：头像连接

created\_date：沃通行证账户创建时间

email：电子邮箱

mobile：手机号码（若用户没有授权获取手机号，将不包含此字段）

nickname：昵称

user\_id：用户唯一标识

authInfo为授权信息，数据形式与授权方法相同。

msg为状态信息，如果授权成功将返回“成功”

若设置了option为WPUAuthCode | WPUAuthAutoRefresh，则在token过期时调用此方法会自动刷新token并获取用户信息。

#### 6) 清除授权信息

调用 - (void)clearAuthInfo 可将授权信息清除