

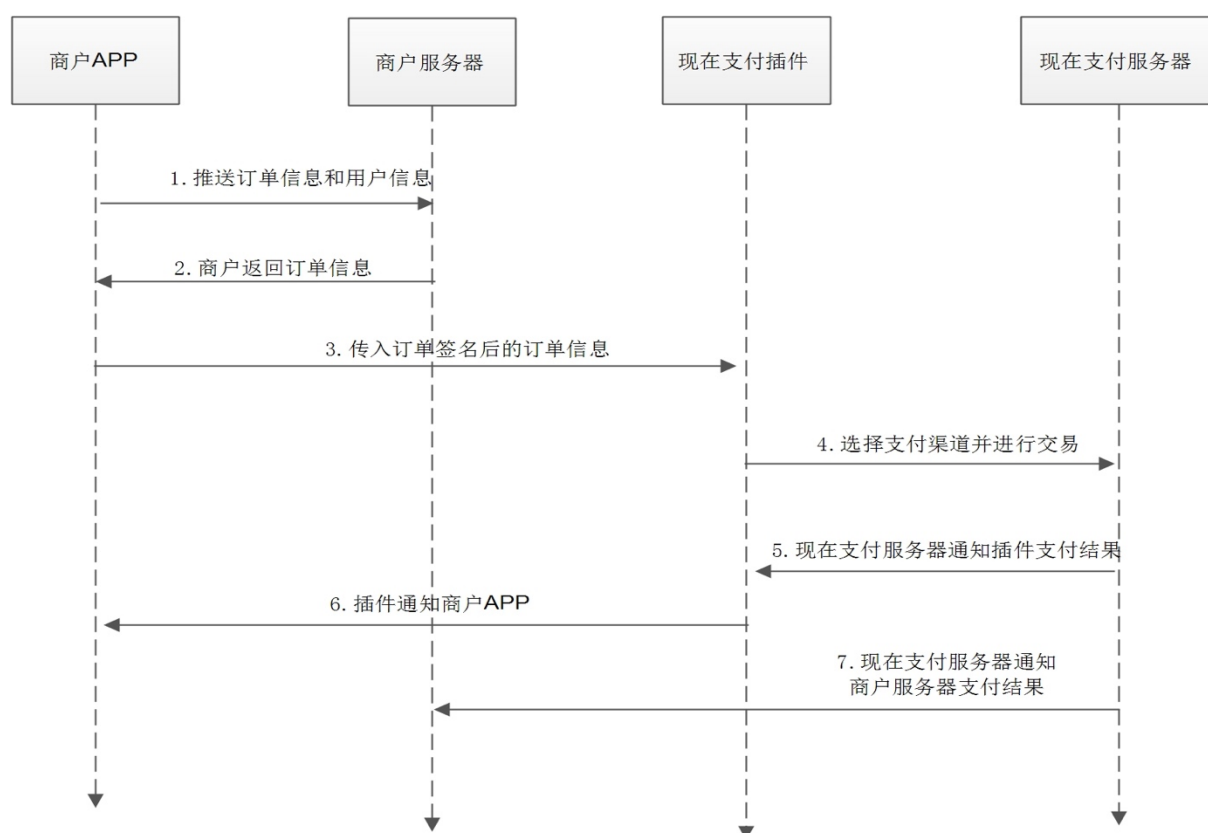
现在支付
中小开发者商户客户端
接入指南
V 2.0.1

一、概述	3
1.1 支付流程介绍:.....	3
二、iOS客户端接入流程	4
1. 文件引用	4
2. 工程设置	6
若使用的IDE是Xcode8及以上请在Capabilities中将Keychain sharing设置为开启。	6
3. 调用支付接口	6
2.4 Apple Pay接入	8
附录A.....	10
附录B.....	11

一、概述

现在支付控件包括银联支付和支付宝支付，主要为开发者的手机客户端提供安全、便捷的支付服务，目前支付控件支持Android和iOS两个平台，用户通过输入银行卡号或支付宝账号等有效信息完成支付。

1.1 支付流程介绍:



步骤说明:

1. 商户APP向商户服务器发送订单信息以及账户信息。
2. 商户APP将符合插件调起接口规范的信息传入调起方法，并调起现在支付支付插件。
3. 用户在插件中支付渠道的选择并完成支付操作。
4. 支付完成后，现在支付支付插件接收服务器发送的支付结果通知。
5. 现在支付支付插件通知商户APP支付情况。**(交易状态以商户后台收到的支付结果通知为准)**
6. 支付成功后，现在支付服务器通知商户服务器交易信息。

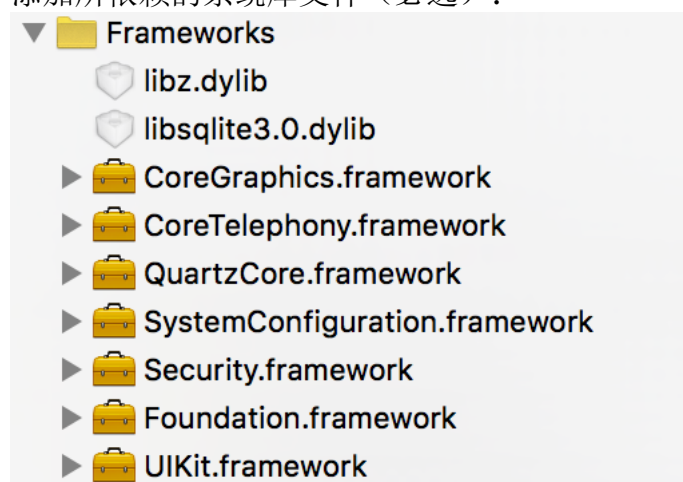
注意:

为了安全考虑，推荐商户服务器收到信息后，根据插件调起接口规范说明（见附录A）组合信息，并对指定字段进行MD5签名。

二、iOS客户端接入流程

1. 文件引用

添加所依赖的系统库文件（必选）：

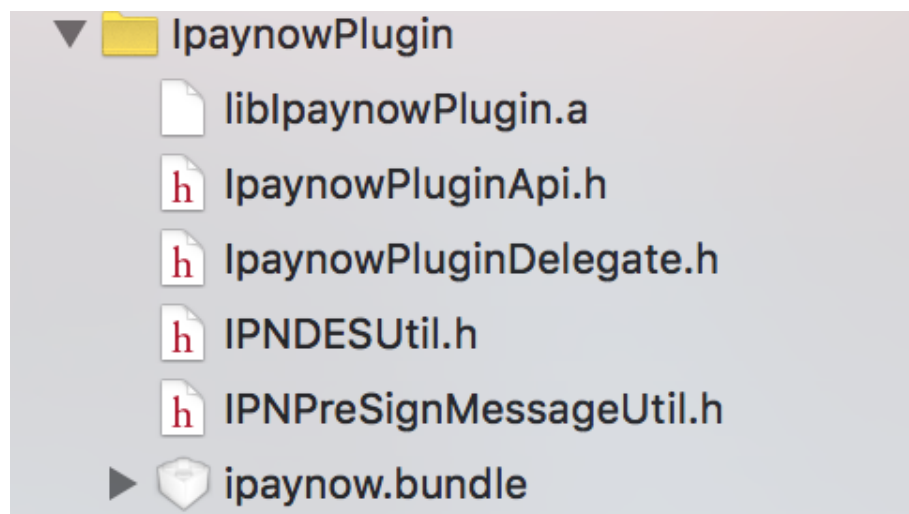


添加基础插件包（必选）：

其中包括ipaynow.bundle、

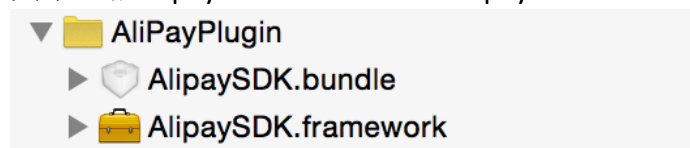
libIPayNowPlugin.a、

IPaynowPluginApi.h、IPaynowPluginDelegate.h及IPNPreSignMessageUtil.h、IPNDESUtil。



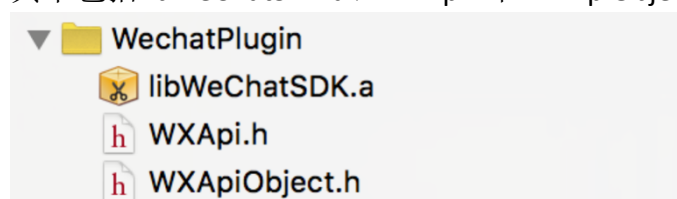
添加支付宝插件包（使用支付宝时引用，可选）：

其中包括AlipaySDK.framework、AlipaySDK.bundle、CoreMotion.framework



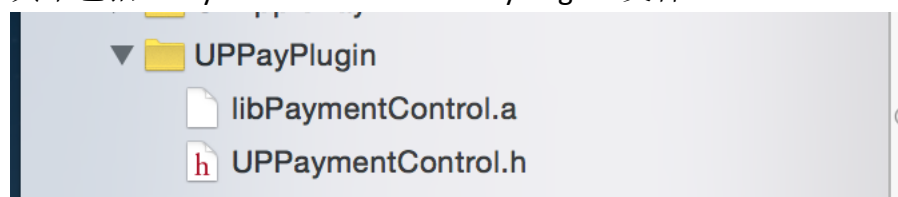
添加微信插件包（使用微信支付时引用，可选）：

其中包括libWeChatSDK.a、WXApi.h和WXApiObject.h。



添加银联插件包（使用银联支付时引用，可选）：

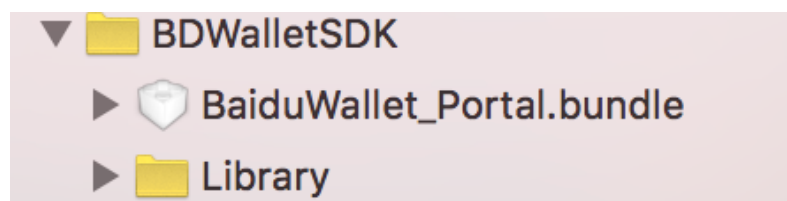
其中包括libPaymentControl.a、UPPayPlugin.h文件。



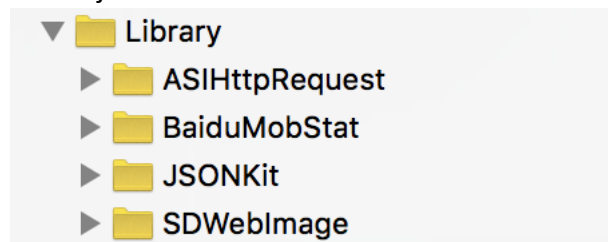
添加百度插件包及依赖第三方库（使用百度钱包时引用，可选）：

其中插件包包括BaiduWalletSDKBundle.bundle、

libBaiduWalletSDK.a、BDWalletSDKMainManager.h文件；

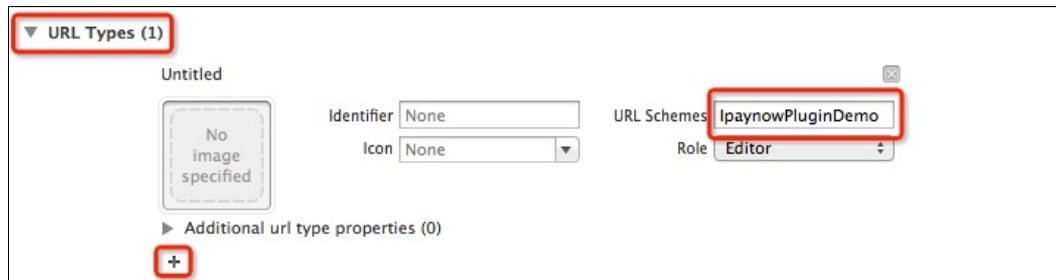


百度钱包插件所依赖的第三方库文件在Library文件夹下，第三方库需要添加的有：ASIHttpRequest、JSONKit、TouchJSON、Reachability。添加第三方库文件时，需要根据当前ARC项目对非ARC的第三方文件设置特定的标志，在Build Phases对这些文件添加“-fno-objc-arc”。



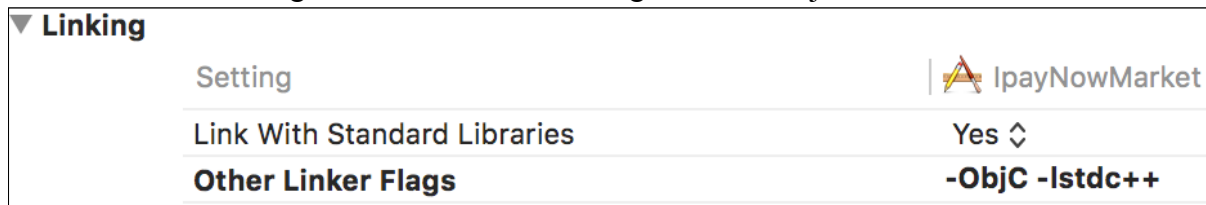
2. 工程设置

在工程的Build Settings中找info，设置URL Types，添加自定义URL Scheme。



URL Scheme在回调结果使用，建议起名稍复杂一些，尽量避免同其他程序冲突。

在工程的Build Settings中找到Other Linker Flags 中添加-ObjC、-lstdc++宏。



若使用的IDE是Xcode8及以上请在Capabilities中将Keychain sharing设置为开启。



3. 调用支付接口

第一步：使用插件中的 IPNPreSignMessageUtil 工具类生成待签名方法：对类中的字段进行赋值,调用 generatePreSignMessage()方法进行待签名串的生成。(若生成结果为 null, 则说明有必传参数没有赋值)

```
NSDateFormatter *dateFormatter = [[NSDateFormatter alloc] init];
[dateFormatter setDateFormat:@"%yyyMMddHHmmss"];
```

```
IPNPreSignMessageUtil *preSign=[[IPNPreSignMessageUtil alloc] init];
preSign.appId=@"1408709961320306";
preSign.consumerId=@"IPN_001";
preSign.consumerName=@"1号消费者";
preSign.mhtOrderNo=[dateFormatter stringFromDate:[NSDate date]];
preSign.mhtOrderName=@"IOS插件测试用例";
preSign.mhtOrderType=@"01";
preSign.mhtCurrencyType=@"156";
preSign.mhtOrderAmt=@"10";
preSign.mhtOrderDetail=@"关于订单验证接口的测试";
preSign.mhtOrderStartTime=[dateFormatter stringFromDate:[NSDate date]];
```

```
preSign.notifyUrl=@"http://localhost:10802/";
preSign.mhtCharset=@"UTF-8";
preSign.mhtOrderTimeOut=@"3600";
```

`NSString *originStr=[preSign generatePresignMessage];`
 (mhtOrderTimeOut、mhtReserved、consumerId、consumerName和payChannelType为选发字段,若有必选字段没有赋值则返回nil)

注:若要传入mhtReserved字段且字段中含有"*"、“&”、“=”，等特殊符号，在获取到签名之后请单独对该字段进行一次url编码，再调用支付，否则可能导致验签失败。

补充说明：通过payChannelType字段可指定跳转到某支付渠道。

第二步：请求后台服务器对待签名串进行签名。

```
NSURL* url = [NSURL URLWithString:kSignURL];
NSMutableURLRequest * urlRequest=[NSMutableURLRequest
requestWithURL:url];
[urlRequest setHTTPMethod:@"POST"];
urlRequest.HTTPBody=[presignStr dataUsingEncoding:NSUTF8StringEncoding];
NSURLConnection* urlConn = [[NSURLConnection alloc]
initWithRequest:urlRequest delegate:self];
[urlConn start];
```

(除mhtSignature字段外还需要加入mhtSignType字段，示例代码中在后台已处理)

第三步：第一步生成的待签名串与第二步服务器生成的签名串拼接起来,传入插件调起方法中。

```
NSString* data = [[NSMutableString alloc] initWithData:mData
encoding:NSUTF8StringEncoding];
NSString* payData=[_presignStr stringByAppendingString:@"&"];
payData=[payData stringByAppendingString:data];
[IpaynowPluginApi pay:payData AndScheme:@"TestPlugin" viewController:self
delegate:self];
```

NSString *data

主要包含商户的订单信息，key=“value”形式，以&连接。

NSString *scheme

商户程序注册的URL protocol，供支付完成后回调商户程序使用。

UIViewController*viewController

商户应用程序调试手机支付的当前UIViewController。

id<IpaynowPluginDelegate>delegate

实现IpaynowPluginDelegate方法的UIViewController。

第四步：实现插件通知接口

接收通知接口应用内结果通知接口为IpaynowPluginDelegate，包含如下方法：

- (void)iPaynowPluginResult:(IPNPayResult)result errorCode:(NSString *)errorCode

```

        errorInfo:(NSString *)errorInfo;
    IPNPayResult result: 支付结果枚举
    NSString *errorCode : 支付失败或未知时返回错误码
    NSString *errorInfo : 支付不成功返回错误信息
    
```

应用间结果通知接口如下:

iOS9之前使用:

```

+ (BOOL)application:(UIApplication*)application openURL:(NSURL *)url
    sourceApplication:(NSString *)sourceApplication annotation:(id)annotation;
    
```

iOS9及以后使用:

```

+ (BOOL)application:(UIApplication*)application openURL:(NSURL *)url options:
(NSDictionary<NSString *,id> *)options
    
```

在AppDelegate.h中添加相应方法实现, 通过在AppDelegate.m中实现来完成独立返回url异步通知。

注①: 选择微信支付后会跳转到微信客户端进行支付, 支付完成需要手动切回App,故请开发者慎重选择开启微信支付。

微信支付需要在AppDelegate.m中添加[IpaynowPluginApi willEnterForeground]方法。

注②: 若使用苹果应用内支付需要在第三步前先调用

```

+ (void) setProductIdentifier:(NSString *)productID andQuantity:(NSInteger)quantity
    orderNo:(NSString *)orderNo;
    NSString *productID : 商品ID
    NSInteger quantity : 商品数量
    NSString *orderNo : 商户订单号
    
```

2.4 Apple Pay接入

接入Apple Pay, 插件接口几乎不用更改, 但是, 和苹果打交道, 总要做点什么, 以下是需要配置的几个地方。

1. 升级Xcode版本到7.2或者以上, 因为我们需要的iOS SDK版本为9.2及以上;
2. iPhone需要是iPhone 6/6 +及以上, 系统版本是iOS 9.2及以上, 需要设置Touch ID、开机密码和登录上iCloud账户, 这样wallet里才会有添加银行卡入口;
3. 申请苹果商户ID(Merchant IDs)
 - 3.1 Identifiers —> Merchant IDs —> Add a Merchant ID
4. App ID 增加Apple Pay Service

App Services

Select the services you would like to enable in your app. You can edit your choices after this App ID has been registered.

- Enable Services:
- ☐ App Groups
 - ☐ Associated Domains
 - ☐ Data Protection
 - ☐ Complete Protection
 - ☐ Protected Unless Open
 - ☐ Protected Until First User Authentication
 - ☒ Game Center
 - ☐ HealthKit
 - ☐ HomeKit
 - ☐ Wireless Accessory Configuration
 - ☒ Apple Pay
 - ☐ iCloud
 - ☐ Compatible with Xcode 5
 - ☐ Include CloudKit support (requires Xcode 6)
 - ☒ In-App Purchase
 - ☐ Inter-App Audio
 - ☐ Passbook
 - ☐ Push Notifications
 - ☐ VPN Configuration & Control

5、工程设置

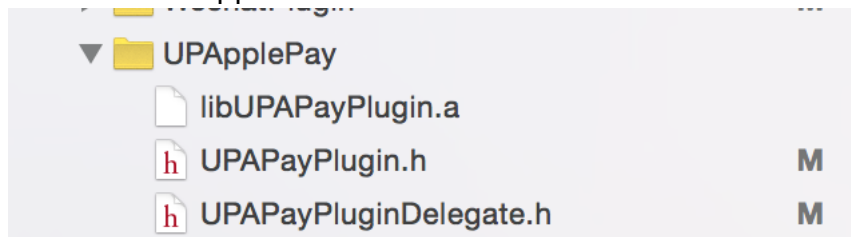
Xcode -> Capabilities -> Apple Pay打开，这时候会看到在developer后台设置的Merchant ID, 打上勾。可以看到项目里的entitlements里多了一项，如图：

Key	Type	Value
▼ Entitlements File	Dictionary	(2 items)
▼ com.apple.developer.in-app-payments	Array	(1 item)
Item 0	String	your Merchant ID

在Link Binary With Libraries中添加PassKit.framework:

▼ Link Binary With Libraries (27 items)		Status
	Name	
	PassKit.framework	Required

6、添加依赖包到App工程中，添加后如图：



已经加有旧版本的现在支付SDK，则需要升级最新的SDK包。

7、在代码里设置Merchant ID，设置方法如下：

```
[IpaynowPluginApi setMerchantID:@"your merchant ID"];
```

8、调用Apple Pay支付，payChannelType传入值为61即可。

附录A

调起插件接口信息规范:

字段名称	字段Key	格式	必填	备注
商户应用唯一标识	appId	String(1,40)	Y	现在支付业务提供
商户订单号	mhtOrderNo	String(1,40)	Y	字母、数字
商户商品名称	mhtOrderName	String(1,40)	Y	
商户交易类型	mhtOrderType	String(2)	Y	01普通消费
商户订单币种类型	mhtCurrencyType	String(3)	Y	156 人民币
商户订单交易金额	mhtOrderAmt	String(1,22)	Y	单位(人民币): 分 整数, 无小数点
商户订单详情	mhtOrderDetail	String(1,1000)	Y	
商户订单超时时间	mhtOrderTimeOut	Number(4,0)	N	60~3600 秒, 默认3600
商户订单开始时间	mhtOrderStartTime	String(14)	Y	yyyyMMddHHmmss
商户后台通知URL	notifyUrl	String(1,200)	Y	HTTP协议
商户字符编码	mhtCharset	定值	Y	UTF-8
渠道类型	payChannelType	定值		银联支付:11 支付宝支付:12; 微信支付:13; 点卡支付:16; 充值卡支付:19 百度钱包:50;
商户保留域	mhtReserved	String(100)	N	商户可以对交易进行标记, 现在支付将原样返回给商户
商户签名方法	mhtSignType	定值	Y	MD5

商户数据签名	mhtSignature	String(1,64)	Y	签名逻辑见接口附录说明5.1 BXXX交易的MD5签名逻辑说明。除如下字段外，其它字段都参与MD5签名。排除的有： mhtSignType, mhtSignature
--------	--------------	--------------	---	---

附录B

第一步：对参与MD5签名的字段按字典升序排序后，分别取值后并排除值为空的字段键值对，最后组成key1=value1&key2=value2....keyn=valuen "表单字符串"。

第二步：对MD5密钥进行加密得到"密钥MD5值"。

第三步：最后对 第一步中得到的表单字符串&第二步得到的密钥MD5值 做MD5签名

PS：MD5密钥是用户在注册应用的时候生成的，每个应用一个MD5密钥。

样例：

```
appId=8888888888888888&mhtCharset=UTF-8&mhtOrderNo=20140821161747&mhtOrderName=%E9%99%B6%E6%A0%91%E5%BC%BA&mhtOrderType=01&mhtCurrencyType=156&mhtOrderAmt=1&mhtOrderDetail=%E5%85%B3%E4%BA%8E%E8%AE%A2%E5%8D%95%E9%AA%8C%E8%AF%81%E6%8E%A5%E5%8F%A3%E7%9A%84%E6%B5%8B%E8%AF%95&mhtOrderStartTime=20140821161747&notifyUrl=http%3A%2F%2Flocalhost%3A10802%2F&mhtSignature=72e3b9fea03b81b88224fe0eab1459d9&mhtSignType=MD5
```

使用简便流程注意：

如用安卓文档中推荐的简便接入流程的话，服务器只要根据商户APP上送的待签名字符串根据以下公式生成签名值即可：

签名公式:MD5（待签名串+"&" +MD5（现在支付提供的密钥））；

生成签名值后，根据以下公式拼接好发送给商户APP：

拼接公式: mhtSignature=（签名公式得到的值）+"&" + mhtSignType=MD5;

样例: mhtSignature=1519adb35c04e0b962d8ca68476d9d56&mhtSignType=MD5

商户使用简便流程时，商户后台需要先做UTF-8的url解码后再做签名处理