**手机支付控件接入指南**

3.0.0

中国银联

2015-12-09

## 版本信息

| 版本号 | 日期 | 说明 |
| --- | --- | --- |
| 2.0.0 | 2013-01-08 | 初稿 |
| 2.1.0 | 2015-09-16 | 1、修正部分错误  2、删除浏览器调起控件相关说明，浏览器方式请参考web支付相关 |
| 3.0.0 | 2015-12-09 | 1、使用UPPaymentControl替换老版本支付控件  2、修改了接口说明和调用插件说明  3、增加了4.4工程配置说明  4、增加了老版本银联商户修改说明 |

目录

版本信息 2

1 概述 1

2 支付流程介绍 1

3 测试帐号 2

4 iOS客户端 3

4.1 SDK说明 3

4.2 接口说明 4

4.3 添加SDK包 6

4.4 工程配置 7

4.5 调用插件 10

5 老版控件用户修改说明 13

5.1 添加SDK包 13

5.2 工程配置 14

5.3 接口说明和调用插件 16

6 常见问题总结 17

­­

## 概述

银联手机支付控件(以下简称支付控件)，主要为合作商户的手机客户端提供安全、便捷的支付服务。用户通过在支付控件中输入银行卡卡号、手机号、密码（借记卡和预付卡）或者CVN2、有效期（信用卡）、验证码等要素完成支付。

**(温馨提示: 对于使用中国银联老版支付控件商户可以跳过其他章节，直接对照本文档第5部分对工程进行改造)**

## 支付流程介绍

通过支付控件进行交易的流程如下图：



流程图说明：

（1）用户在客户端中点击购买商品，客户端发起订单生成请求到商户后台；

（2）商户后台收到订单生成请求后，按照《手机控件支付产品接口规范》组织并推送订单信息至银联后台；

（3）银联后台接收订单信息并检查通过后，生成对应交易流水号（即TN），并回复至商户后台（应答要素：交易流水号等）；

（4）商户后台接收到交易流水号（TN），将交易流水号返回至客户端；

（5）客户端通过交易流水号（TN）调用支付控件；

（6）用户在支付控件中输入相关支付信息后，由支付控件向银联后台发起支付请求；

（7）支付成功后，银联后台将支付结果通知给商户后台；

（8）银联后台同时也将支付结果通知支付控件；

（9）支付控件显示支付结果并将支付结果返回至客户端；

**注: 本文档主要关注上述流程中（5）（9）部分的实现**

iOS版本支付控件适用iOS 6.0及以上版本终端设备。

## 测试帐号

以下是测试用卡号、手机号等信息（此类信息仅供测试使用，不会发生真实交易）

|  |
| --- |
| 招商银行借记卡：6226090000000048     手机号：18100000000     密码：111101     短信验证码：123456（先点获取验证码之后再输入）     证件类型：01身份证     证件号：510265790128303     姓名：张三 |
| 华夏银行贷记卡：6226388000000095     手机号：18100000000     CVN2：248     有效期：1219     短信验证码：123456（先点获取验证码之后再输入）     证件类型：01身份证     证件号：510265790128303     姓名：张三 |

## iOS客户端

本小节涉及到SDK的接口说明、Xcode工程配置及其接口调用细节，需要读者具有一定iOS编程经验。

### SDK说明

商户开发者在获取到银联提供的开发包后请检查SDK文件所在目录upmp\_iphone/paymentcontrol，以下部分提及的文件均在该目录中。银联支付控件静态库，以下简称UPPaymentControl，包含以下两个文件：

UPPaymentControl.h

libPaymentControl.a

### 接口说明

1. 支付接口

|  |
| --- |
| - (BOOL)startPay:(NSString\*)tn  fromScheme:(NSString\*)schemeStr  mode:(NSString\*)mode  viewController:(UIViewController\*)viewController |

各个参数定义如下表：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **类型** | **含义** |
| tn | NSString\* | 必填项；  交易流水号，商户后台向银联后台提交订单信息后，由银联后台生成并下发给商户后台的交易凭证； |
| schemeStr | NSString \* | 必填项;  商户自定义协议，商户在调用支付接口完成支付后，用于引导支付控件返回而定义的协议，具体请参考4.4第二步中URL Type定义; |
| mode | NSString\* | 必填项；  接入模式，标识商户以何种方式调用支付控件，该参数提供以下两个可选值：  "00"代表接入生产环境（正式版本需要）；  "01"代表接入开发测试环境（测试版本需要）； |
| viewController | UIViewController\* | 必填项；  发起调用的视图控制器，商户应用程序调用银联手机支付控件的视图控制器； |
| 返回值 | BOOL | YES：调起支付控件成功；  NO：调起支付控件失败； |

2、检查是否安装银联App的接口

|  |
| --- |
| - (BOOL)isPaymentAppInstalled |

此函数无传入参数，主要功能检测用户手机上是否安装银联支付的APP，当用户手机上安装了银联支付APP时候返回YES。

|  |  |  |
| --- | --- | --- |
| **参数名称** | **类型** | **含义** |
| 返回值 | BOOL | YES：已安装银联支付APP；  NO：未安装银联支付APP； |

3、返回结果接口

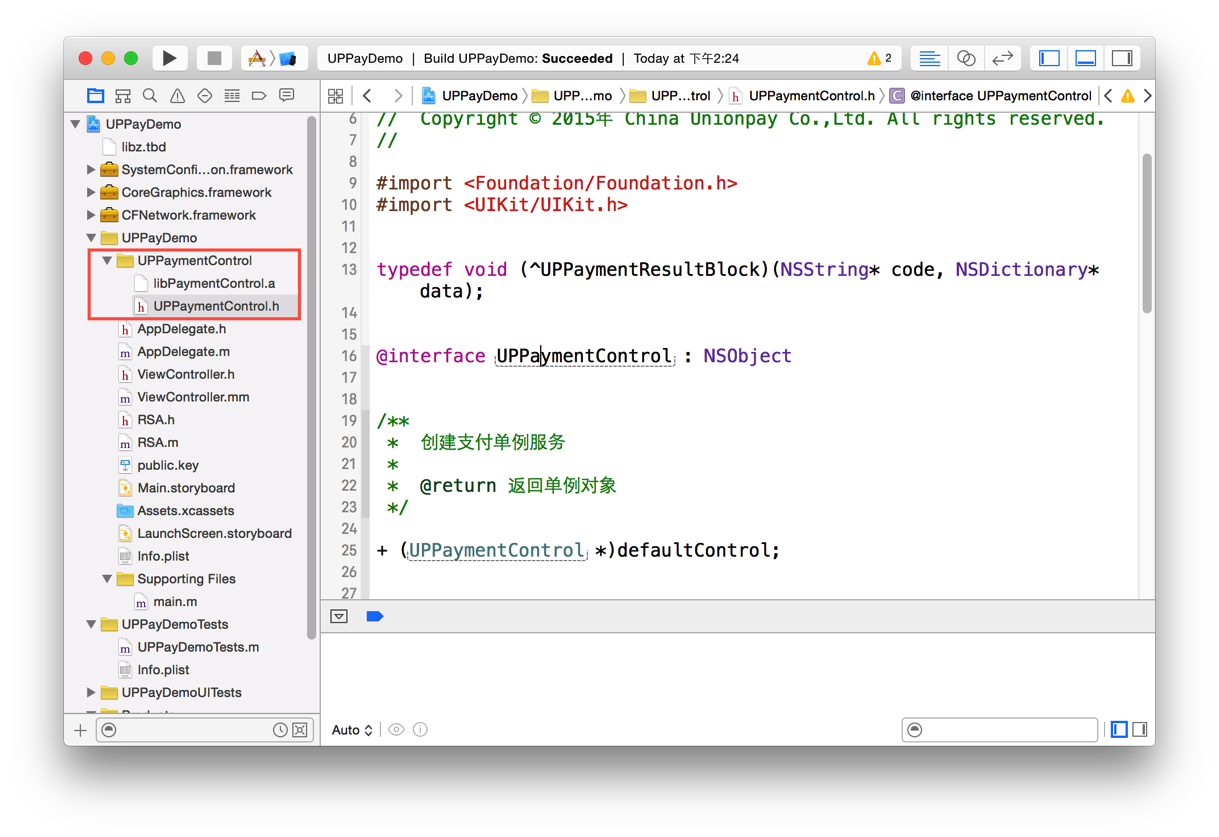
|  |
| --- |
| - (void)handlePaymentResult:(NSURL\*)url  completeBlock:(UPPaymentResultBlock)completionBlock; |

各参数定义如下表：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **类型** | **含义** |
| url | NSURL\* | 必填项；  支付结果url，传入后由SDK解析并通过completionBlock回调商户客户端； |
| completionBlock | Block | 必填项；  商户客户端定义的结果处理方法，此方法包行code、data两个传入参数，本文档4.5第三部分将详细说明； |

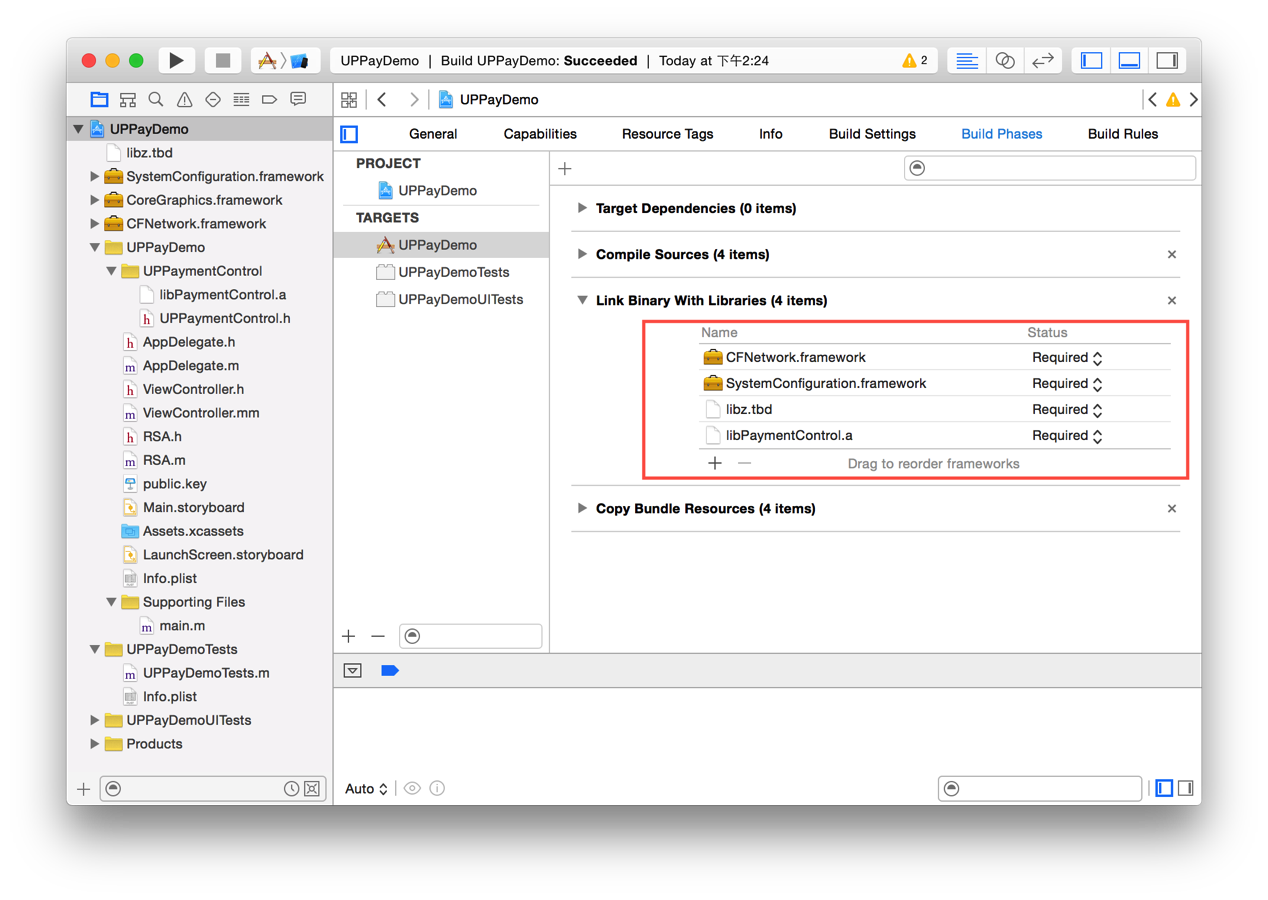
### 添加SDK包

使用UPPaymentControl需要将paymentcontrol/inc目录下UPPaymentControl.h文件和paymentcontrol/libs目录下的libPaymentControl.a文件添加到商户应用的工程中，添加后如图：

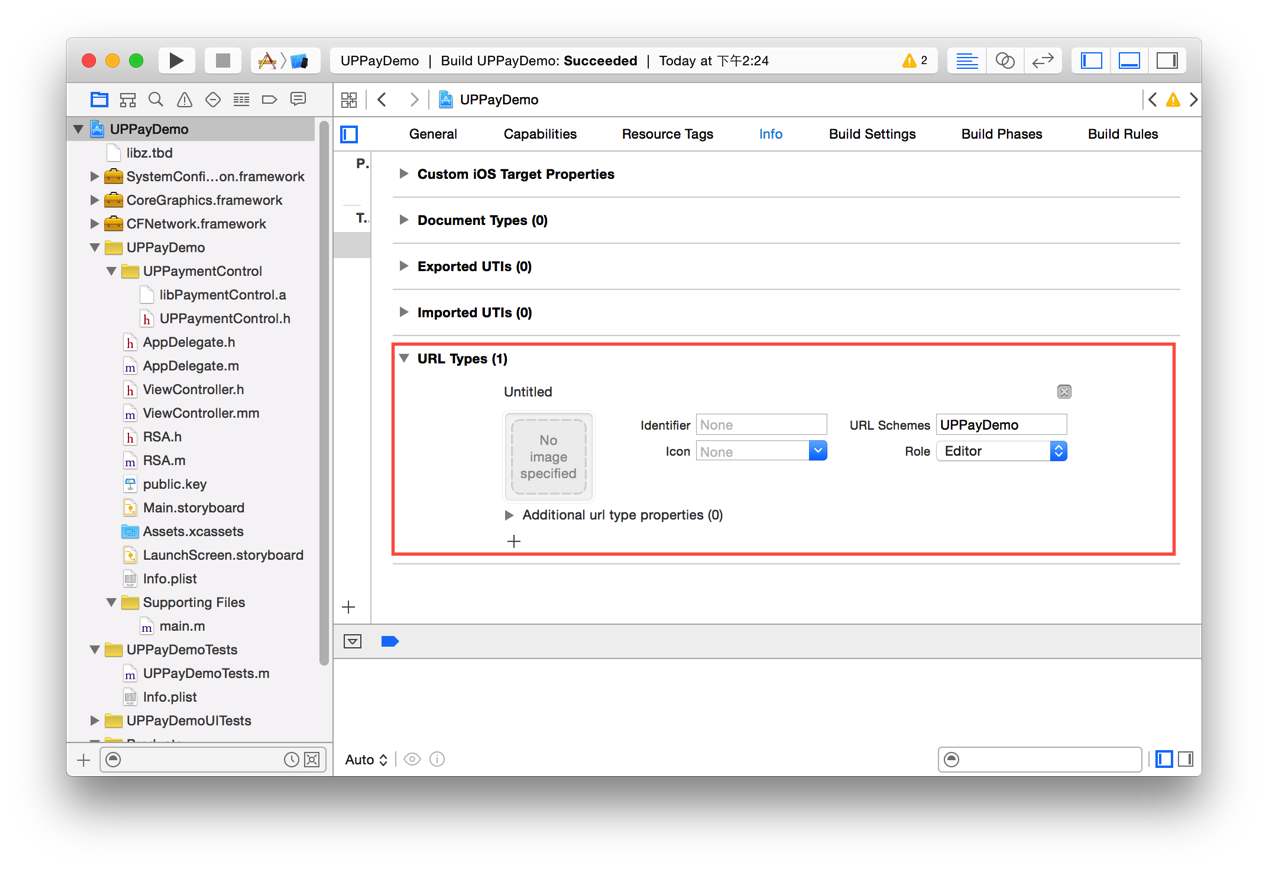


### 工程配置

1. 使用UPPaymentControl需要添加CFNetwork.framework、ＳystemConfiguration.framework 、libz、libPaymentControl.a到工程中，添加后如下图：

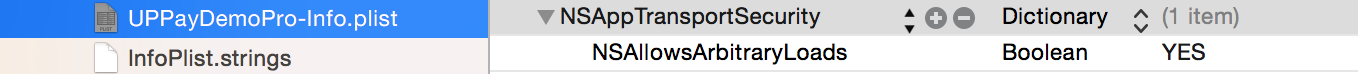


2、在工程info.plist设置中添加一个URL Types回调协议(在UPPayDemo工程中使用“UPPayDemo”作为协议)，用于在支付完成后返回商户客户端。



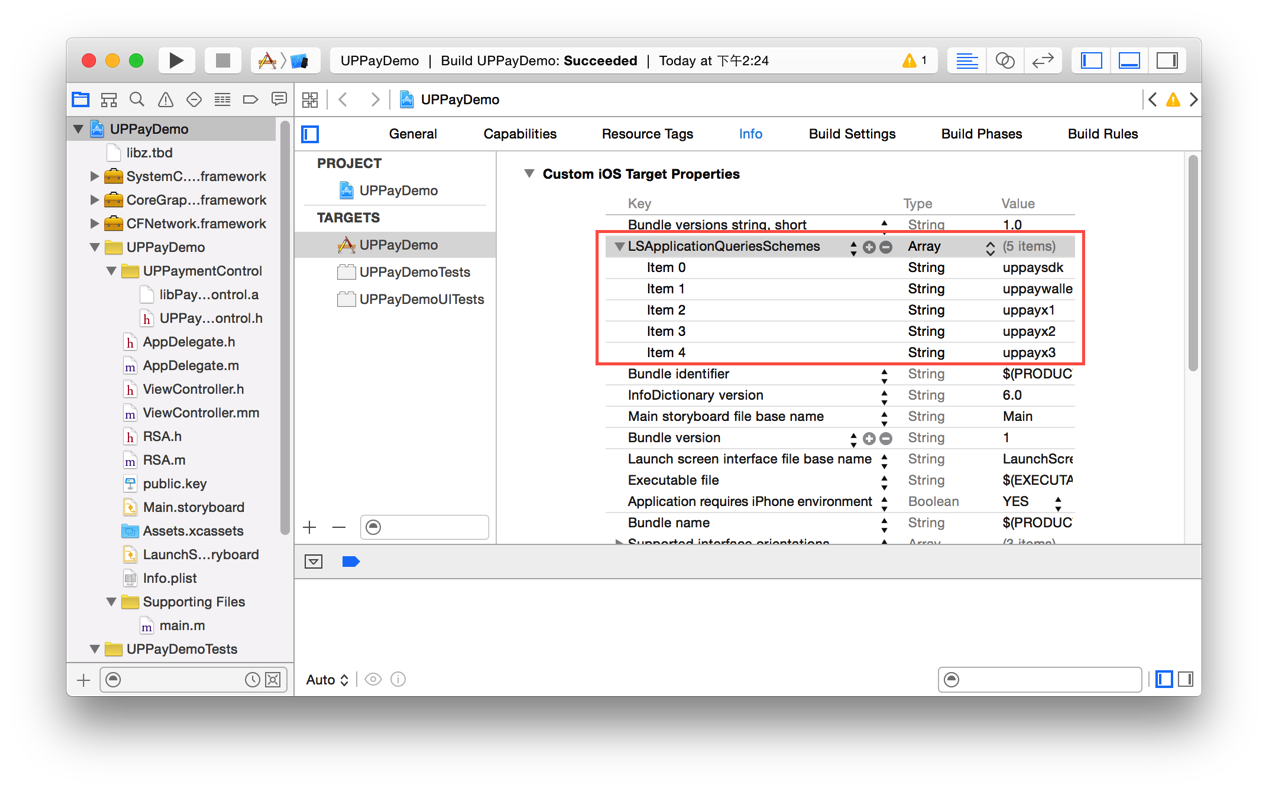
3、http请求设置

在Xcode7.0之后的版本中进行http请求时，需要在工程对应的plist文件中添加NSAppTransportSecurity Dictionary 并同时设置里面NSAllowsArbitraryLoads 属性值为 YES，具体设置可参照以下截图：



4、添加协议白名单

在Xcode7.0之后的版本中进行开发，需要在工程对应的plist文件中，添加LSApplicationQueriesSchemes Array并加入uppaysdk、uppaywallet、uppayx1、uppayx2、uppayx3五个item，具体设置可参考以下截图：



或者直接添加如下代码到plist文件中：

|  |
| --- |
| <key>LSApplicationQueriesSchemes</key>  <array>  <string>uppaysdk</string>  <string>uppaywallet</string>  <string>uppayx1</string>  <string>uppayx2</string>  <string>uppayx3</string>  </array> |

### 调用插件

在需要调用支付控件接口的代码文件内引用头文件UPPaymentControl.h。

（注意：如果工程的compile source as 选项的值不是Objective–C++，则引用此头文件的文件类型都要改为.mm）

1、支付接口调用

商户App从商户服务器获取tn，当tn不为空时，调用支付接口。

|  |
| --- |
| //当获得的tn不为空时，调用支付接口  if (tn != nil && tn.length > 0)  {  [[UPPaymentControl defaultControl]  startPay:tn  fromScheme:@"UPPayDemo"  mode:self.tnMode  viewController:self];  } |

2、检测是否已安装银联App接口调用

|  |
| --- |
| if([[UPPaymentControl defaultControl] isPaymentAppInstalled])  {  //当判断用户手机上已安装银联App，商户客户端可以做相应个性化处理  } |

3、返回结果接口调用

支付控件结果处理函数handlePaymentResult: completeBlock:需要在工程AppDelegate文件的application: openURL: options: 方法中进行调用。

支付控件结果处理函数handlePaymentResult: completeBlock:包含两个参数，参数1url为支付结果串，由handlePaymentResult: completeBlock:方法解析url内容；参数2completionBlock为商户APP定义的结果处理方法，包含两个传入参数code和data，其中code表示支付结果，取值为suceess,fail,cancel分别表示支付成功、支付失败和支付取消，data表示结果签名数据，商户使用银联公钥验证结果真实性。

对于新增的签名信息需注意以下几点：

1. 前台返回的支付结果中包含银联签名，要在商户后台对签名进行校验后才能展示结果。

2. 前台签名使用的密钥和算法与后台结果中的签名一致。

3. 如果商户APP在客户端内进行签名验证，要自行实现签名密钥更新的机制，否则更换密钥后会导致验签失败。（不推荐）

4. 商户订单是否成功支付应该以商户后台收到全渠道返回的支付结果为准，此处支付控件返回的结果仅作为参考。

调用支付接口后，结果处理方法示例代码：

|  |
| --- |
| - (BOOL)application:(UIApplication \*)app openURL:(NSURL \*)url options:(NSDictionary<NSString\*, id> \*)options {  [[UPPaymentControl defaultControl] handlePaymentResult:url completeBlock:^(NSString \*code, NSDictionary \*data) {    //结果code为成功时，先校验签名，校验成功后做后续处理  if([code isEqualToString:@"success"]) {    //数据从NSDictionary转换为NSString  NSDictionary \*data;  NSData \*signData = [NSJSONSerialization dataWithJSONObject:data  options:0  error:nil];  NSString \*sign = [[NSString alloc] initWithData:signData encoding:NSUTF8StringEncoding];    //判断签名数据是否存在  if(data == nil){  //如果没有签名数据，建议商户app后台查询交易结果  return;  }    //验签证书同后台验签证书  //此处的verify，商户需送去商户后台做验签  if([self verify:sign]) {  //支付成功且验签成功，展示支付成功提示  }  else {  //验签失败，交易结果数据被篡改，商户app后台查询交易结果  }  }  else if([code isEqualToString:@"fail"]) {  //交易失败  }  else if([code isEqualToString:@"cancel"]) {  //交易取消  }  }];    return YES;  } |

completeBlock中的NSDictionary \*data结构如下**：**

|  |
| --- |
| sign —— 签名后做Base64的数据  data —— 用于签名的原始数据，结构如下：  pay\_result —— 支付结果success，fail，cancel  tn —— 订单号 |

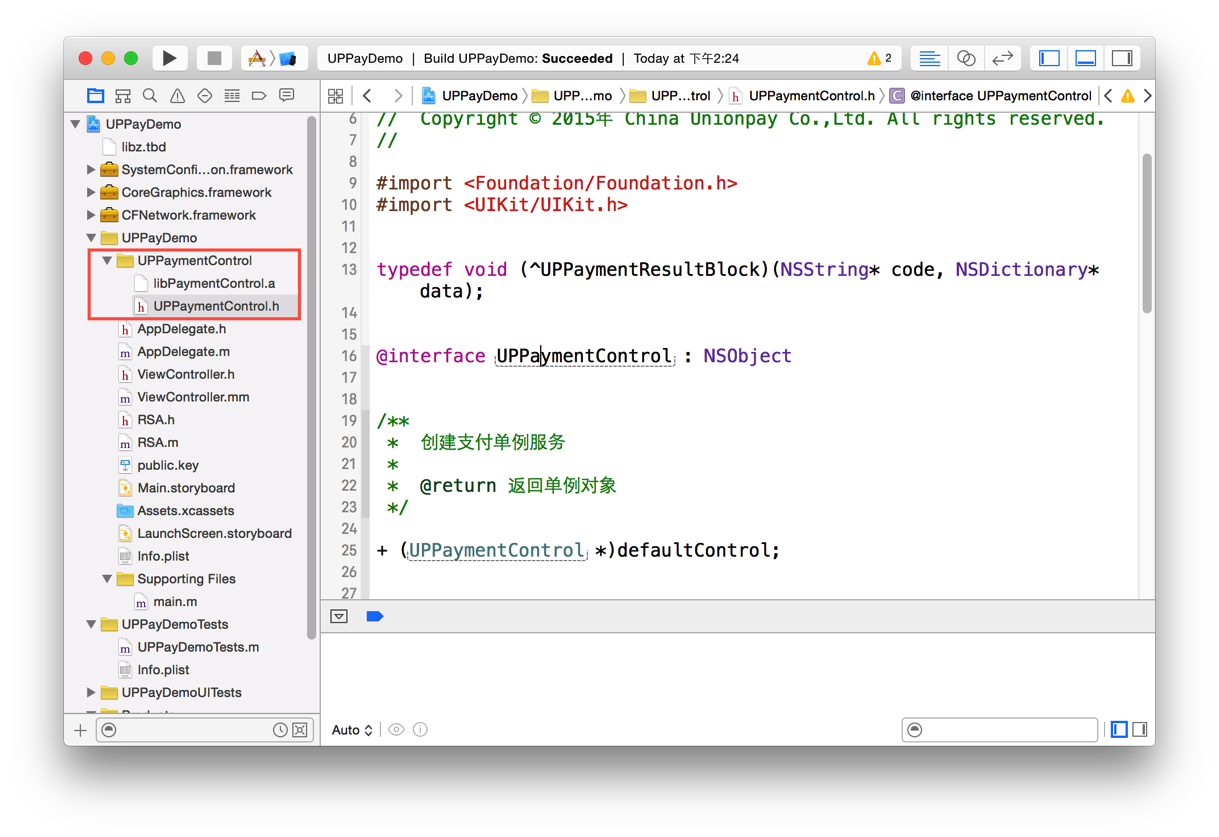
Data转换为String后的示例如下：

"{"sign":"ZnZY4nqFGu/ugcXNIhniJh6UDVriWANlHtIDRzV9w120E6tUgpL9Z7jIFzWrSV73hmrkk8BZMXMc/9b8u3Ex1ugnZn0OZtWfMZk2I979dxp2MmOB+1N+Zxf8iHr7KNhf9xb+VZdEydn3Wc/xX/B4jncg0AwDJO/0pezhSZqdhSivTEoxq7KQTq2KaHJmNotPzBatWI5Ta7Ka2l/fKUv8zr6DGu3/5UaPqHhnUq1IwgxEWOYxGWQgtyTMo/tDIRx0OlXOm4iOEcnA9DWGT5hXTT3nONkRFuOSyqS5Rzc26gQE6boD+wkdUZTy55ns8cDCdaPajMrnuEByZCs70yvSgA==","data":"pay\_result=success&tn=201512151321481233778"}"

## 老版控件用户修改说明

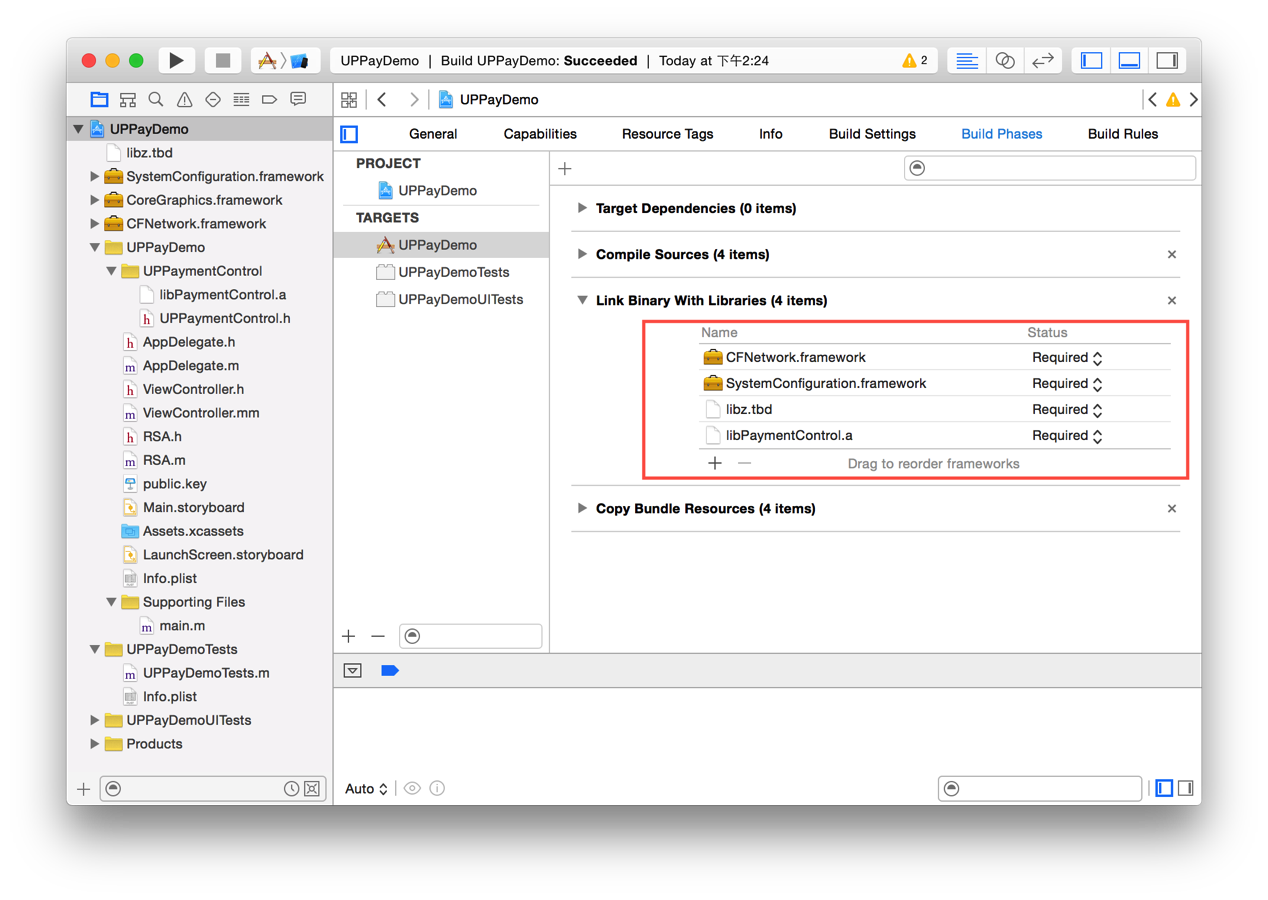
### 添加SDK包

使用老版本银联支付控件的商户请先将老版本银联支付控件从工程中删除，再将新paymentcontrol/inc目录下UPPaymentControl.h文件和paymentcontrol/libs目录下libPaymentControl.a文件添加到商户应用的工程中；添加后如图：

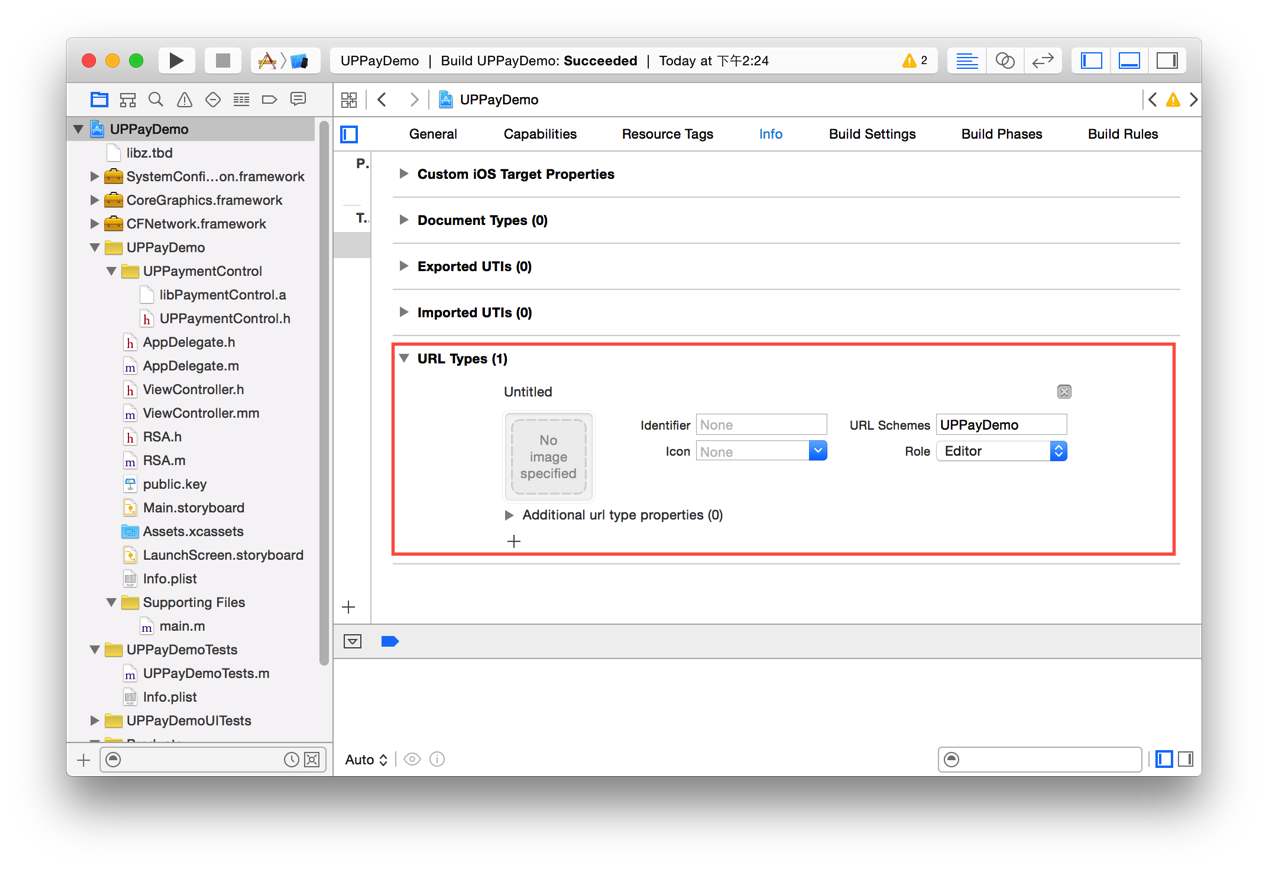


### 工程配置

1、使用老版本银联支付控件的商户请先将工程配置中老版支付控件对应的三方库删除，再添加CFNetwork.framework、SystemConfiguration.framework 、libPaymentControl.a、libz到工程中；添加后如下图：

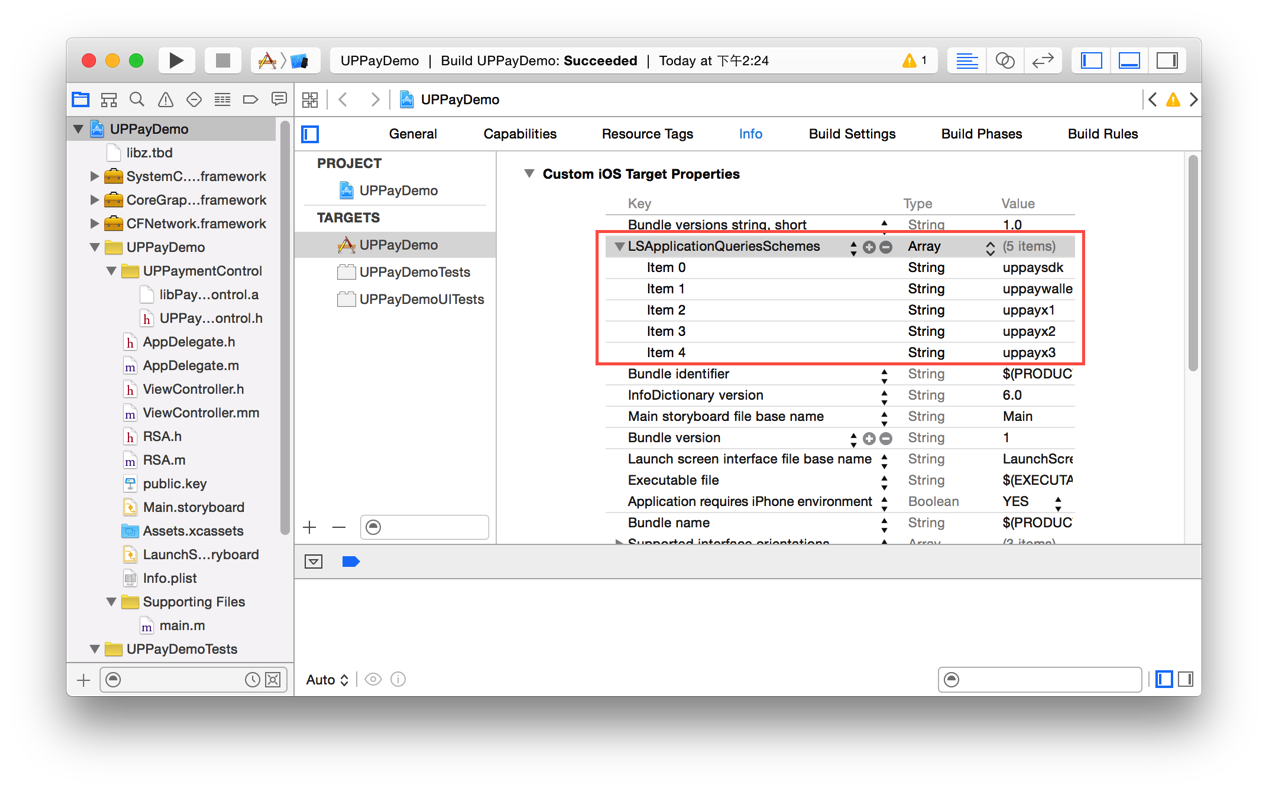


2、在工程info.plist设置中添加一个URL Types回调协议(在UPPayDemo工程中使用“UPPayDemo”作为协议)，用于在支付控件完成支付后返回到商户客户端。



3、添加协议白名单

在工程对应的plist文件中，添加LSApplicationQueriesSchemes Array并加入uppaysdk、uppaywallet、uppayx1、uppayx2、uppayx3五个item，具体设置可参考以下截图:



或者直接添加以下代码到plist文件中：

|  |
| --- |
| <key>LSApplicationQueriesSchemes</key>  <array>  <string>uppaysdk</string>  <string>uppaywallet</string>  <string>uppayx1</string>  <string>uppayx2</string>  <string>uppayx3</string>  </array> |

### 接口说明和调用插件

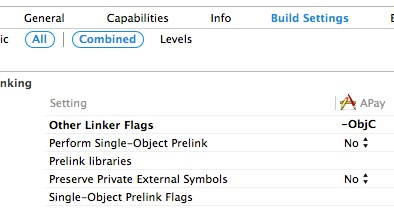
请参考4.2接口说明和4.4调用插件说明。

## 常见问题总结

更多请参见<https://open.unionpay.com>帮助中心-FAQ

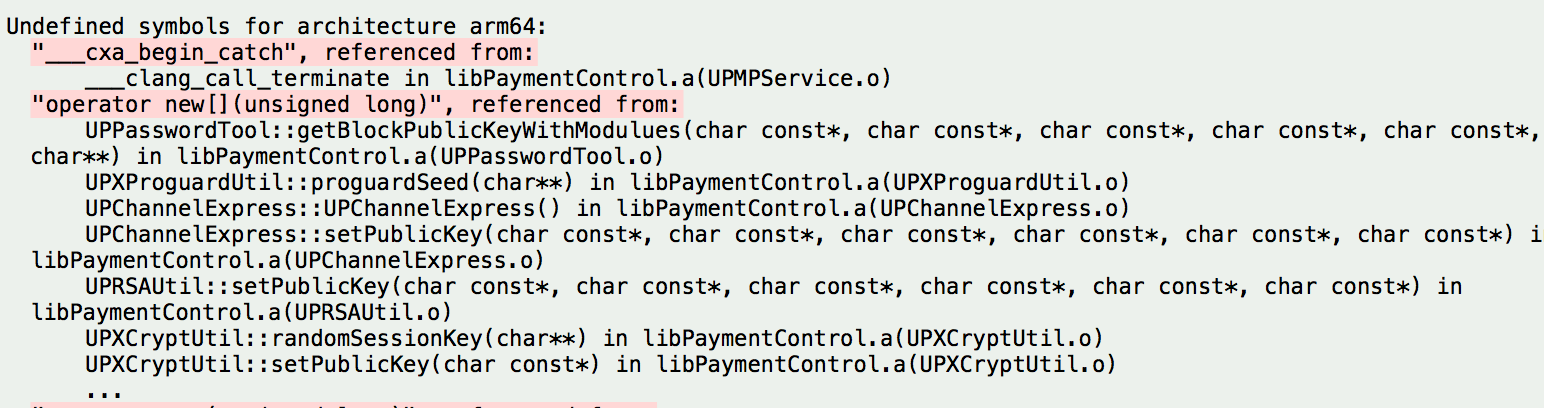
### 怎样加-ObjC宏

选择工程targets——》build settings ->Linking->other linker flags



### 编译时提示Undefined for architecture XXX 错误

如：



1）由于支付控件使用到了C、C++和OC混编的情况，所以商户工程引入UPPaymentControl.h头文件以后可能会出现链接错误，这个时候可以通过以下三种方式解决：

① 将涉及到引用UPPaymentControl.h的源文件的后缀名都改为.mm；

② 如果商户不想修改源文件的后缀名，可以在工程中添加一个空的继承自NSObject的类，并将文件.m后缀名该改为.mm即可方法为new file->Objective-C class->类名自取->保存->修改后缀名为.mm；

③ 将工程的compile source as 选项的值设置为Objective–C++；

2）由于在UPPayDemo工程中添加了自定义的库文件libPaymentControl.a

，当编译Demo工程时，应该检查工程设置Search Paths里的Framework Search Paths、Header Search Paths、Library Search Paths的路径设置，看设置路径是否正确，另外还要注意里边是否多余一些不确定的路径。

3) 将xcode工程中Build Setting -> Apple LLVM compiler Language 标签下的C++ Language Dialect 和 C++ Standard Library 属性值修改为 Compiler Default如实在无法解决，尝试添加-ObjC宏的地方修改为  -force\_load+空格+控件路径，如：-force\_load $(PROJECT\_DIR)/ libPaymentControl.a，如果还报错，上下文应该会有libPaymentControl.a文件找不到异常，比如ld: file not found: /Users/apple/Desktop/Communication 2/ libPaymentControl.a

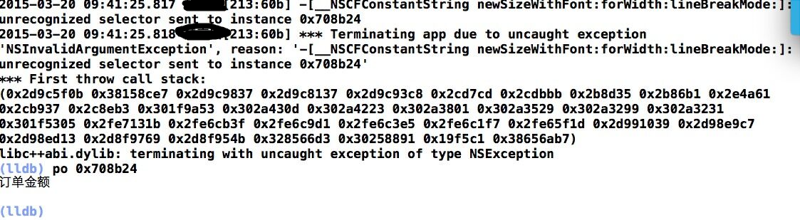
clang: error: linker command failed with exit code 1 (use -v to see invocation)

请确定libPaymentControl.a文件确实存在于此路径，可能为上下文路径配错。

### 控件闪退异常'NSInvalidArgumentException', reason: '-[\_\_NSCFConstantString newSizeWithFont:详略]'

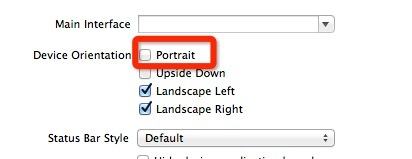
 'NSInvalidArgumentException', reason: '-[\_\_NSCFConstantString newSizeWithFont:以下略]: unrecognized selector sent to instance 0x[随机数]'

如图：



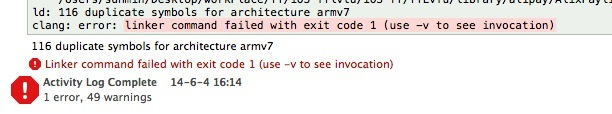
出现以上问题是由于添加-ObjC宏的地方没有配置正确，如果配-ObjC实在解决不了的话，可尝试去掉-ObjC，改为-force\_load+空格+控件路径，如：-force\_load $(PROJECT\_DIR)/ libPaymentControl.a。

### orientation异常

详细现象为控件崩溃，报错信息：Terminating app due to uncaught exception ‘UIApplicationInvalidInterfaceOrientation’, reason: ‘Supported orientations has no common orientation with the application, and shouldAutorotate is returning YES’   
请勿修改横竖屏配置，iphone控件只能竖屏，ipad控件才支持横竖屏。  
如果app本身为横屏，请代码控制横竖屏，勿修改配置。  


### 其他第三方静态库（如支付宝、微信等）的文件产生冲突duplicate symbols for architecture XXX

如：



请尝试将添加-ObjC宏的地方修改为  -force\_load+空格+控件路径，如：-force\_load $(PROJECT\_DIR)/ libPaymentControl.a，这样做不会影响支付宝等其他支付渠道功能。

### 控件界面无限加载

1. 调用控件的逻辑应当在主线程中执行，子线程中尽量不要进行UI操作，无限加载等待实际是子线程死锁所致。  
2. 请参考控件使用指南iOS “添加SDK包“里“添加-ObjC宏”，特别注意大小写不要弄混。（此步骤操作有问题时也可修改为  -force\_load+空格+控件路径，如：-force\_load $(PROJECT\_DIR)/ libPaymentControl.a）。  
3. Build Setting -> Apple LLVM compiler Language-C++：  
 

### Attempt to present XX on XX while a presentation is in progress!

Warning: Attempt to present <UPNavController: 0x8bc24d0> on <tttViewController: 0x8b9a390> while a presentation is in progress!  
animation完成的时候再调用我们的插件，两个present同时起会出警告。