

手机支付控件使用指南

2.0.0

中国银联

2013-01-08

版本号	日期	说明
2.0.0	2013-01-08	初稿

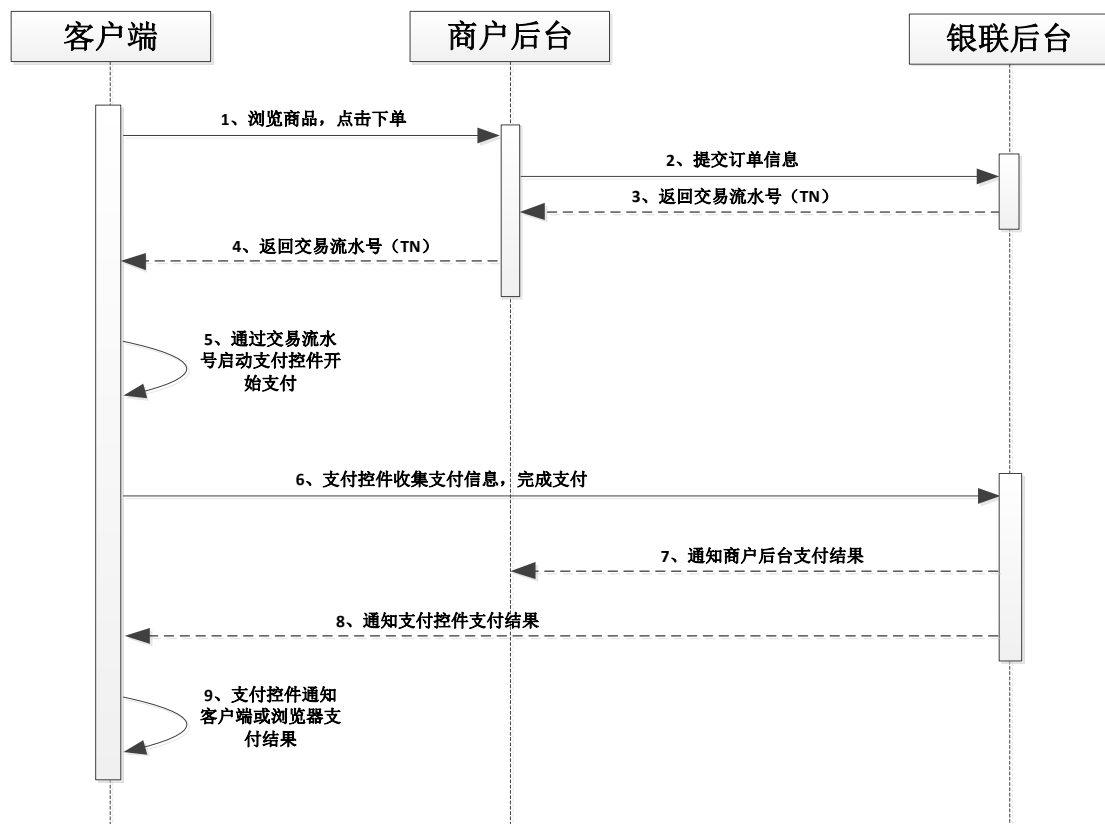
一、概述	1
二、支付流程介绍	1
二、测试帐号	2
三、iOS 客户端	2
1、 接口说明	2
2、 添加 SDK 包	3
3、 调用支付控件	3
四、Android 客户端	4
1、 接口说明	4
2、 添加 SDK 包	5
3、 调用支付控件	5
五、手机 Web 站点	6
1、 支持浏览器	6
2、 方式一	7
3、 方式二	7
4、 Web 订单生成	8
六、开发者注意事项	9
七、常见问题总结	9
1、 iOS 平台常见问题	9
2、 Android 平台常见问题	10
附录一、JS 代码	10

一、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或手机 Web 网站提供安全、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。

二、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《UPMP 商户接入接口规范》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), 并回复交易流水号至商户后台(应答要素: 交易流水号等);
- (4) 商户后台接收到交易流水号, 将交易流水号返回给客户端;

- (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给客户端;

注: 本文档主要关注上述流程中 (5)、(9) 部分的实现

目前各个平台支持的设备情况如下:

Android 平台 SDK 主要适用于 Android 1.6 及以上版本的终端设备;

iOS 版本支付控件适用 iOS 4.3 及以上版本终端设备。

二、测试帐号

- 提供测试使用卡号、手机号信息 (此类信息仅供测试, 不会发生正式交易)

三、iOS 客户端

本小节提供给那些具有一定 iOS 编程经验和了解面向对象概念的读者使用。

1、接口说明

```
+ (BOOL)startPay:(NSString*)payData
    sysProvide:(NSString*)sysProvide
    spId:(NSString*)spId
    mode:(NSString*)mode
viewController:(UIViewController *)viewController
    delegate:(id<UPPayPluginDelegate>)delegate;
```

各个参数的介绍如表3-1:

表3-1 接口参数说明

参数名称	类型	含义
payData	NSString*	必填项; 交易流水号信息, 银联后台生成, 通过商户后台返回到客户端并传 入支付控件;

sysProvide	NSString*	保留使用，这里输入nil
spId	NSString*	保留使用，这里输入nil
mode	NSString*	必填项； 接入模式设定，两个值： @"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； 商户应用程序调用银联手机支付的当前UIViewController；
delegate	id<UPPayPluginDelegate>	必填项； 实现 UPPayPluginDelegate 方法的 UIViewController；

2、添加 SDK 包

- 将 sdk/inc 文件夹中的 UPPayPlugin.h、UPPayPluginDelegate.h 和 sdk/libs 文件夹下 libUPPayPlugin.a 三个文件添加到 UPPayDemo 工程根目录中；
- 右键单击工程，通过菜单中 Add Files to 将 UPPayPlugin.h、UPPayPluginDelegate.h 和 libUPPayPlugin.a 添加到工程中；

3、调用支付控件

- 添加 QuartzCore.framework 到 UPPayDemo 工程中；
- 在需要调用支付控件的源文件内引用头文件 UPPayPlugin.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）
- 在工程 target 的 other link flags 中添加-ObjC 宏；
- 通过调用

```

+ (BOOL)startPay:(NSString*)payData
    sysProvide:(NSString*) sysProvide
      spId:(NSString*) spId
    mode:(NSString*)mode
viewController:(UIViewController *)viewController
    delegate:(id<UPPayPluginDelegate>)delegate;

```

实现控件的调用

e) 处理支付结果

银联手机支付控件有三个支付状态返回值：success、fail、cancel，分别代表：支付成功、支付失败、用户取消支付。这三个返回状态值以字符串的形式作为回调函数参数(NSString*)result 返回。通过在工程中添加头文件

“UPPayPluginDelegate.h”，在处理交易结果的界面，实现

UPPayPluginDelegate 接口，根据该头文件中的回调函数：

-(void)UPPayPluginResult:(NSString*)result 来实现回调方法，从而可以根据支付结果的不同进行相关的处理。

四、Android 客户端

本小节提供给那些具有一定 Android 编程经验和了解面向对象概念的读者使用。

1、接口说明

upmp_android/UPPayAssistEx.jar包中定义了启动支付控件的接口，接口定义如下：

```
public static int startPay(Activity activity, String spId,
                          String sysProvider, String orderInfo, String mode)
```

参数说明：

activity ——用于启动支付控件的活动对象

spId ——保留使用，这里输入null

sysProvider ——保留使用，这里输入null

orderInfo ——订单信息为交易流水号，即TN。

mode —— 银联后台环境标识，“00”将在银联正式环境发起交易，“01”将在银联测试环境发起交易

返回值：

UPPayAssistEx.PLUGIN_VALID —— 该终端已经安装控件，并启动控件

UPPayAssistEx.PLUGIN_NOT_FOUND —— 手机终端尚未安装支付控件，需要先安装支付控件

其它辅助接口：

```
public static boolean installUPPayPlugin(Context context)
```

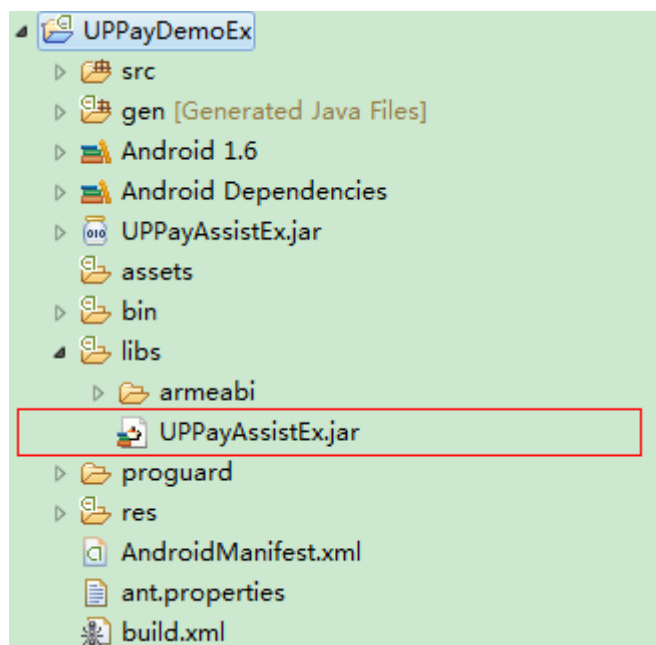
参数说明：

context ——安装控件的上下文

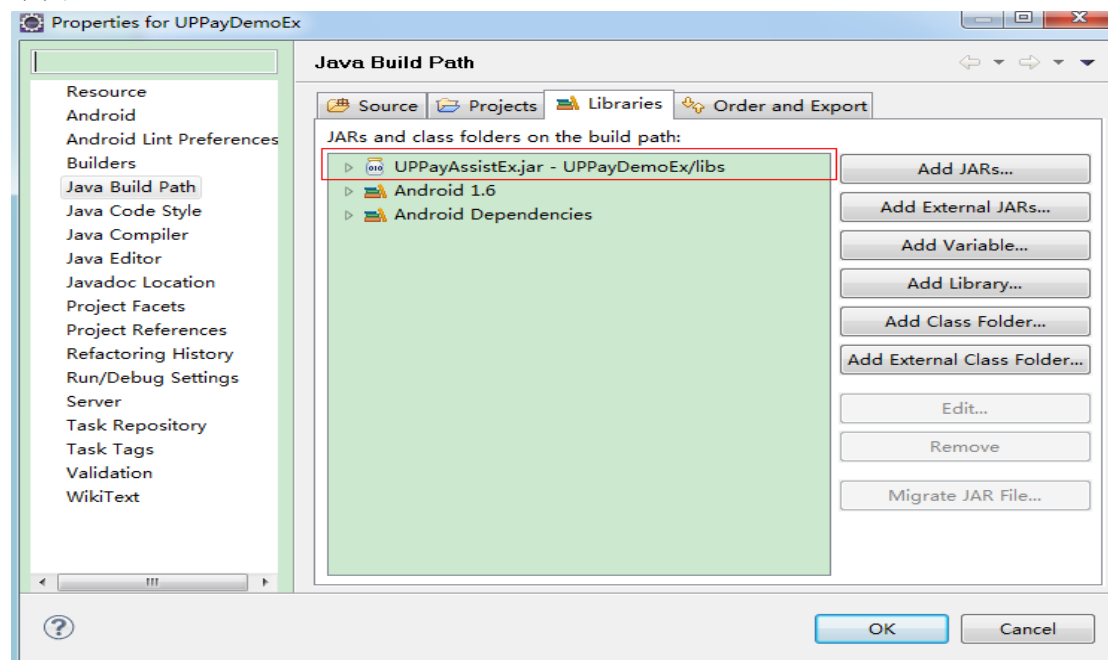
返回值： true ——安装正常，false ——安装失败

2、添加 SDK 包

将银联Android版本支付控件sdk中upmp_android目录下UPPayAssistEx.jar复制到工程的libs目录下；同时也可将upmp_android/APK目录下的UPPayPluginEx.apk复制到客户端工程的assets目录下，效果如下图：



接着请右键单击工程，选择Build Path中的 Configure Build Path ...，选中 Libraries这个tab，并通过Add Jars...导入工程libs目录下的UPPayAssistEx.jar包。如下图：



3、调用支付控件

- a) 在调用支付控件的代码文件中引入UPPayAssistEx类如：


```
import com.unionpay.UPPayAssistEx;
```

b) 接着可以通过以下方式调用支付控件：

```
// “00” - 银联正式环境
// “01” - 银联测试环境，该环境中不发生真实交易
String serverMode = “01” ;
int ret = UPPayAssistEx.startPay ( activity, null, null, tn, serverMode);
if( ret == UPPayAssist.PLUGIN_NOT_FOUND ){
    //安装Asset中提供的UPPayPlugin.apk
    // 此处可根据实际情况，添加相应的处理逻辑
    UPPayAssistEx.installUPPayPlugin(activity);
}
```

支付完成后，获取支付控件支付结果，并添加相应处理逻辑，只需实现调用Activity中的onActivityResult()方法即可，实例代码如下：

```
protected void onActivityResult( int requestCode,
                                int resultCode,
                                Intent data)
{
    if( data == null ){
        return;
    }

    String str = data.getExtras().getString("pay_result");
    if( str.equalsIgnoreCase(R_SUCCESS) ){
        showResultDialog(" 支付成功! ");
    }else if( str.equalsIgnoreCase(R_FAIL) ){
        showResultDialog(" 支付失败! ");
    }else if( str.equalsIgnoreCase(R_CANCEL) ){
        showResultDialog(" 你已取消了本次订单的支付! ");
    }
}
```

五、手机 Web 站点

1、支持浏览器


- UC 浏览器 Android 版本（通过方式一接入）
- UC 浏览器 iOS 版本（通过方式二接入）
- QQ 浏览器 Android 版本 3.6 及以上（通过方式二接入）
- 360 浏览器 Android 版本 2.7 及以上（通过方式二接入）
- Opera 浏览器 Android HD 版本 1.3 及以上（通过方式二接入）

2、方式一

调用支付控件的 Web 页面需要嵌入代码

```
<embed type="application/x-unionpayplugin"
        uc_plugin_id="unionpay"
        height="53"
        width="178"
        paydata=" 订单信息" >
</embed>
```

其中<embed>为银联手机支付 UC 插件标签项，在 UC 浏览器中显示为银联手机支付

按钮，。其中前几个参数，不要进行修改：

```
type="application/x-unionpayplugin" uc_plugin_id="unionpay" height="53"
width="178"
```

其中，**订单信息**部分构成参考 [Web 订单生成](#)。

支付控件页面嵌入代码范例（消费交易）

```
<embed type="application/x-unionpayplugin"
        uc_plugin_id="unionpay"
        height="53"
        width="178"
        paydata="dG49MjAxMjEyMjYxNjAwNTI3NTA0MjgyLFJlc3VsdFVS
        TD1odHRwJTNBJTJGJTJGMjE4LjgwLjE5%OD%0AMi4yMTM1M0ExNzI
        1JTJGY2xzdw41MkZyZXN1bHQ1M0ZpZCUzRCxVc2VUZXR0TW9kZT1m
        YWxzZQ%3D%3D%OD%0A">
</embed>
```

3、方式二

调用银联手机支付控件页面需要嵌入代码

```
<a href="uppay://uppayservice/?style=token&paydata=token"></a>
```

其中，<a>为银联手机支付非 UC 浏览器接入控件采用标签项，在非 UC 浏览器中显示为银联手机支付按钮，。银联手机支付图标部分指明该图标的路径。

参数名	参数值	说明
paydata	参考 Web 订单生成	本次交易的订单信息构成

支付控件页面嵌入代码范例

```
<a
href="uppay://uppayservice/?style=token&paydata=dG49MjAxMjEyMjYxNjAw
NTI3NTA0MjgyLFJlc3VsdFVSTD1odHRwJTNBjTJGJTJGMjE4LjgwLjE5%0D%0AMi4yMT
MlM0ExNzI1JTJGY2xzdW41MkZyZXN1bHQ1M0ZpZCUzRCxVc2VUZlN0TW9kZT1mYWxzZQ
%3D%3D%0D%0A">
</a>
```

4、Web 订单生成

展示在Web页面上的订单信息生成方式如下：

- 将表 5-1 中订单信息采用 base64 进行编码；
- 将步骤 a) 中产生的结果采用 URLEncode 方式编码得到最终展示的数据；

Web 站点订单中的每个参数对由参数值组成，参数间“,” 隔开。参数间不要有空格，格式如下：

表5-1 web订单构成

tn=交易流水号,resultURL=支付通知 URL,usetestmode=是否使用银联测试环境

表5-2 订单信息域说明

参数名	参数描述	参数属性	备注
tn	交易流水号	银联后台生成，通过商户后台，返回到浏览器，并传入支付控件，为必填项	必备域
resultURL	支付结果通知url，该url主要用于支付控件在支付完成或失败后通知浏览器跳转的地址	该参数由商户提供，采用 URLEncode 方式编码 。该url需要接受一个参数，该参数由控件返回，返回参数为： 0 - 支付成功时返回； 1 - 支付失败时返回； -1 - 支付取消时返回 示例如下（其中蓝色部分为商户提供、红色的数字为控件追加）： http://example.com/example?argName=0 http://example.com/example?argName=1	必备域

		ame=1 http://example.com/example?argN ame=-1	
usetestmode	是否使用银联测试环境标识	true - 使用银联测试环境测试 false - 使用银联生产环境	必备域

六、开发者注意事项

- 1、通过浏览器方式接入支付控件时，传入的 resultURL 应该可以接受参数，并且需要采用 URLEncode 方式进行编码，编码之前的 url 形式如：

<https://example.com/example?argName=>

- 2、浏览器以及标签类别判断 js 实例代码

附件自带的 browserSnoof.js

用方法如下：

```
// accept 头部由商户后台返回给前端页面
var ret = browserSupport(accept);

if(ret.support != "true"){
    alert("该浏览器不支持银联手机支付控件");
}else{
    alert("该浏览器支持银联手机支付控件");
    alert("使用： " + ret.tag + " 标签");
}
```

七、常见问题总结

1、iOS 平台常见问题

➤ 编译错误解决

UPPayDemo 工程在编译的过程中可能会出现 Undefined symbols for architecture armv6/armv7/i386 的编译错误。如果出现这样的错误，有以下几种解决办法：

- 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 UPPayPlugin.h 头文件以后可能会出现链接错误。这个时候可以通过两种方式解决：

- ① 将涉及到引用 UPPayPlugin.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 选项的值不是 Objective-C++;
- 2) 由于在 UPPayDemo 工程中添加了自定义的库文件 libUPPayPlugin.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

2、Android 平台常见问题

附录一、JS 代码

表 1 浏览器控件支持类型判断

```
function browserSupport(accept) {  
    var ucSupport = ucSnoof(accept);  
    if(ucSupport.support == "true"){  
        return ucSupport;  
    }  
    // 通用判断  
    var userAgent = navigator.userAgent.toLowerCase();  
    // 到这里，uc 已经不判断了  
    if (platformSnoof(userAgent) == "android") {  
        if ((/360 aphone browser/.test(userAgent))  
            || (/opera\//.test(userAgent) && /oupenghd/.test(userAgent))  
            || (/qqbrowser/.test(userAgent))) {  
            return {  
                'support' : 'true',  
                'tag' : 'a'  
            }  
        }  
    }  
}
```

```
        };

        }else{

            return {

                'support' : 'false',

            };

        }

    } else {

        return {

            'support' : 'false',

        };

    }

}

function creditSupport(accept){

    var ucSupport = ucSnoof(accept);

    if(ucSupport.support == "true"){

        return ucSupport;

    } else{

        return {

            'support' : 'false',

        };

    }

}

function ucSnoof(accept){

    if (!accept) {

        return;

    }

    // uc 浏览器判断
```

```
var ucIos = new RegExp("ios_plugin/1");
var ucAndroid = new RegExp("plugin/1");
if (ucIos.test(accept)) {
    return {
        'support' : 'true',
        'tag' : 'a'
    };
} else if (ucAndroid.test(accept)) {
    return {
        'support' : 'true',
        'tag' : 'embed'
    };
} else{
    return {
        'support' : 'false',
    };
}
}

function platfromSnoof(userAgent) {
    var ios = new RegExp("iphone os");
    var android = new RegExp("android");
    if (ios.test(userAgent)) {
        return "ios";
    }
    if (android.test(userAgent)) {
        return "android";
    }
}
```

