万普支付 SDK for Android 接口文档

V4.1.2

万普世纪支付 SDK（以下简称支付 SDK）提供了一套 Android 应用内支付的完整开发包，以及 Demo 源代码，便于开发者在 Android 应用中方便快捷的集成支付的各种功能，目前支持支付宝、银行卡、充值卡、财付通、余额支付五大支付方式。

引入 SDK 包

1. 引入 Jar 包:

A．将 Demo 解压包中 libs 目录中的 jar 包复制到项目 libs 目录下,并将 arm64-v8a、armeabi、armeabi-v7a、mips、x86 复制到 libs 目录下:

```
WanpuPay_4.1.2.jar
alipaysdk.jar
android-support-v13.jar
MobileSecSdk.jar
utdid4all-1.0.4.jar
payecopluginjar.jar
TenpayServiceSDK_V5.0.jar
UPPayPluginExStd.jar
UPPayAssistEx.jar
```

注：如果应用需混淆编译，则需在混淆配置中排除 jar 包中的内容，配置代码如下：

```
-keep public interface com.wanpu.pay.** {*;}
-keep public class com.wanpu.pay.** {*;}
-keep class com.alipay.android.app.IAlixPay{*;}
-keep class com.alipay.android.app.IAlixPay$Stub{*;}
-keep class com.alipay.android.app.IRemoteServiceCallback{*;}
-keep class com.alipay.android.app.IRemoteServiceCallback$Stub{*;}
-keep class com.alipay.sdk.app.PayTask{ public *;}
-keep class com.alipay.sdk.auth.AlipaySDK{ public *;}
-keep class com.alipay.sdk.auth.APAuthInfo{ public *;}
-keep class com.alipay.mobilesecuritysdk.*
-keep class com.ut.*
-keep public interface com.payeco.android.plugin.** {*;}
-keep public class com.payeco.android.plugin.** {*;}
-keep public interface com.tenpay.android.service.** {*;}
-keep public class com.tenpay.android.service.** {*;}
-keep public interface com.unionpay.** {*;}
-keep public class com.unionpay.** {*;}
-keep public interface com.UCMobile.** {*;}
-keep public class com.UCMobile.** {*;}
```

2. 引入必要的资源文件:

将 Demo 解压包中的 Resources 目录下的 assets、drawable、layout、values 目录下的资源文件复制到工程中对应的文件夹中，没有请创建。

注：assets 下新增 data.bin 文件

3. 添加配置信息：

3.1 将以下 Activity 相关内容添加注册到 AndroidManifest.xml 文件中:

```
<activity
      android:name="com.wanpu.pay.PayView"
      android:configChanges="keyboardHidden|orientation" />
<activity
    android:name="com.payeco.android.plugin.PayecoPluginLoadingActivity"
    android:exported="true"
    android:screenOrientation="portrait"
    android:theme="@android:style/Theme.Translucent"
    android:windowSoftInputMode="adjustPan|stateHidden" />
<activity
    android:name="com.payeco.android.plugin.PayecoOrderDetailActivity"
    android:screenOrientation="portrait"
    android:windowSoftInputMode="adjustPan|stateHidden" />
<activity
    android:name="com.payeco.android.plugin.PayecoPayResultActivity"
    android:screenOrientation="portrait"
    android:windowSoftInputMode="adjustPan|stateHidden" />
<activity
    android:name="com.payeco.android.plugin.PayecoWebViewActivity"
    android:screenOrientation="portrait"
    android:windowSoftInputMode="adjustPan|stateHidden" />
<activity
    android:name="com.payeco.android.plugin.PayecoRiskControlActivity"
    android:configChanges="orientation|keyboardHidden"
    android:screenOrientation="portrait"
    android:windowSoftInputMode="adjustPan|stateHidden" />

<activity
      android:name="com.unionpay.uppay.PayActivity"
      android:configChanges="orientation|keyboardHidden"
      android:excludeFromRecents="true"
      android:screenOrientation="portrait"
      android:theme="@style/Theme.UPPay" />
<activity
      android:name="com.alipay.sdk.app.H5PayActivity"
      android:configChanges="orientation|keyboardHidden|navigation"
      android:exported="false"
      android:screenOrientation="behind" />
```

3.2 将以下权限添加到 AndroidManifest.xml 文件中：

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
 <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<!-- 以下为银行卡支付需多添加的权限 -->
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.GET_TASKS" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" />
```

加入接口代码

1.初始化计数器

在应用首个启动 Activity 中的 onCreate()方法中调用统计器接口:

```
PayConnect.getInstance("APP_ID", "APP_PID", context);
```

APP_ID:　应用标识 ID,从万普平台（www.waps.cn）获取。一个 APP_ID 只能用于一个固定包名的应用，一旦应用正式上线后不能更换包名，否则将导致认证失败。不同包名的应用需使用独立的 APP_ID。

APP_PID:　应用的发布渠道标识。每个渠道包需使用不同的 PID 参数，便于区分不同渠道产生的数据。

2.添加支付接口

2.1 调用支付接口，启动选择支付方式的界面:

```
PayConnect.getInstance(context).pay(context,
                orderId, userId, price, goodsName, goodsDesc, notifyUrl, new MyPayResultListener());
```

参数说明：

参数	名称	说明
context	应用上下文	
orderId	订单号	开发者自定义的支付订单号，每条支付数据必须要使用不同订单号
userId	用户标识	应用存在用户系统的情况下，该值为当前用户标识 没有用户系统的应用可用设备 ID 代替用户帐号： PayConnect.getInstance(context).getDeviceId(context);
price	请求支付金额	该次支付请求的支付金额。(float 类型)
goodsName	商品名称	该次支付请求所购买商品的名称（需小于 30 汉字）
goodsDesc	商品描述	该次支付请求所购买商品的描述（需小于 30 汉字）
notifyUrl	服务器通知接口	开发者指定的服务器通知接口（必须以“http://”开头），在支付成功时，万普服务器会以 get 方式向该接口发送结果数据，该接口具体参数请参看“服务器通知接口”。该参数如设置为空，则不进行服务器端通知。

参数	名称	说明
MyPayResultListener	开发者自定义类	该类需实现 SDK 中的 PayResultListener 接口中的 onPayFinish(Context payViewContext,String order_id,int resultCode, String resultString, int payType, float amount, String goods_name)方法，用于接收支付结果的回调参数。 回调参数说明： payViewContext：支付选择界面上下文 resultCode：支付返回状态码。0 为支付成功，其他为失败 resultString：支付平台返回的支付结果提示信息 payType：支付方式。 1.银行卡 2.支付宝 3.充值卡 4.财付通 amount：实付金额 goods_name：商品名称

2.2 实现一个 PayResultListener 接口，处理客户端返回的支付结果

说明：开发者有两种方法可以获得订单支付结果，一种是通过客户端实现一个 PayResultListener 接口（必须实现），也可以通过再指定一个服务器端的 notifyUrl 地址来接收订单状态。

如果指定了 notifyUrl 参数，则支付成功后，万普平台会主动通知指定接口地址，开发者返回内容为“success”，即作为交易成功的回执，无需在客户端再手动发送交易回执。否则，如果未指定 notifyUrl 参数，则需要通过 PayResultListener 接口来处理交易，并在交易完成后手动调用 confirm()接口发送交易回执。

支付成功超过 3 分钟仍未收到到交易回执的订单，将在再次调用 pay 方法时，自动重新触发一次 PayResultListener 接口，同步最近一次支付成功但未受到回执的历史订单。用户如遇到支付成功后订单状态未同步的情况，可在 3 分钟后重新进入一次支付界面即可自动同步。

PayResultListener 接口实现的代码示例:

```
public class MyPayResultListener implements PayResultListener{
        @Override
        public void onPayFinish(Context payViewContext,String order_id,int resultCode, String resultString, int payType, float amount, String goods_name) {
            if(resultCode == 0){ // 支付成功
                Toast.makeText(getApplicationContext(), resultString + " : " + amount + "元", Toast.LENGTH_LONG).show();
                // 支付成功时关闭当前支付界面
                PayConnect.getInstance(MainActivity.this).closePayView(payViewContext);
                // TODO 在客户端处理支付成功的操作

                // 未指定 notifyUrl 的情况下，交易成功后，必须发送回执
                PayConnect.getInstance(MainActivity.this).confirm(order_id, payType);
            }else{// 支付失败
                Toast.makeText(getApplicationContext(), resultString, Toast.LENGTH_LONG).show();
            }
        }
}
```

3.销毁使用的资源

在最后退出应用的 Activity 的 onDestroy()方法中，或开发者自定义的推出方法中调用下面代码：

```
PayConnect.getInstance(context).close();  //以前版本的 finalize()方法作废
```

服务器端通知接口

如果应用有的服务器，万普支平台将以 get 方式将支付结果同时提交到商户指定的 notifyUrl

1.通知参数说明：

参数名	名称	说明
order_id	商户订单 ID¤	长度 50 位以内的字符串
app_id	商户应用 ID	由万普平台生成的应用标识，32 位字符串
user_id	用户 ID	游戏中的用户唯一 ID
pay_type	支付方式	银行卡:1 或 6，支付宝:2，充值卡:3，财付通:4，余额支付:5
result_code	支付结果状态	成功:0，失败为其他值
result_string	支付结果描述	“支付成功”或支付失败原因描述（UTF8 编码）
trade_id	支付流水号	由第三方支付平台返回的实际交易记录编号
amount	实际支付金额	例如：1.00
pay_time	支付时间	例如：2013-05-08 14:29:52.0
sign	签名	规则 md5(orderId+userId+price+key) key 值在接入时请由万普提供

2.商户响应信息：

商户成功收到通知后请直接返回“success”字符串即可。

3.通知地址示例：

http://cpserver.com/recerve?order_id=1367994550017&app_id=09f277ca386ee99cb4c910e09f562112&user_id=test_user&pay_type=2&result_code=0&result_string=Success&trade_id=2013050835671212&amount=1.00&pay_time=2013-05-08%2014:29:52.0&sign=xxxxxxx

4.支付通知重发：

若万普支付平台未收到商户返回的“success”，则会将同一笔订单的通知进行周期性重发（时间时隔：3 分钟，10 分钟，30 分钟，1 小时，2 小时，6 小时，12 小时）。受网络因素影响，万普平台可能会对某条订单的支付结果进行重复通知，开发者需根据订单号进行排重识别。

注意：如果指定了 notifyUrl，为避免第三方伪造支付结果请求，须在接受万普平台结果通知的服务端程序中增加 IP 限制，并将万普支付平台网关 IP 加入可访问名单中。万普支付平台 IP：219.234.85.205,219.234.85.206，219.234.85.207，219.234.85.217， 219.234.85.234