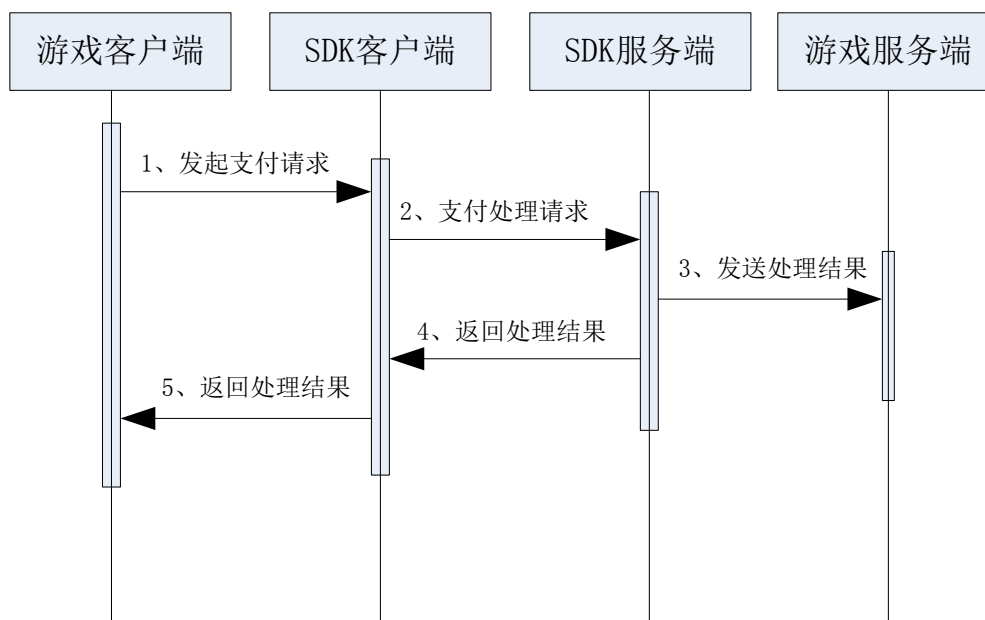


目录

一、消耗可币处理流程示意图:	1
二、回调参数说明:	1
三、验证签名方法&示例.....	2
3.1 签名步骤	2
3.2 相关 java 代码.....	3
3.3 签名公钥	3
四、游戏服务端回调处理说明:	4

一、消耗可币处理流程示意图：



消耗可币成功后，SDK 服务器会根据开发上传的回调地址（客户端代码 payinfo->setcallbackurl 中设置）通知开发者订单信息，回调方法为 HTTP POST。

二、回调参数说明：

参数名	说明	类型及长度限制
notifyId	回调通知 ID（该值使用系统为这次支付生成的订单号）	String(50)
partnerOrder	开发者订单号（客户端上传）	String(100)
productName	商品名称（客户端上传）	String(50)
productDesc	商品描述（客户端上传）	String(100)
price	商品价格(以分为单位)	int
count	商品数量（一般为 1）	int
attach	请求支付时上传的附加参数（客户端上传）	String(200)
sign	签名	String

三、验证签名方法&示例

3.1 签名步骤

1) 利用回调的参数生成baseString，方法如下：

```
private static String getKebiContentString(String url) {  
    final String[] strings = url.split("&");  
    final Map<String, String> data = new HashMap<String, String>();  
    for (String string : strings) {  
        final String[] keyAndValue = string.split("=");  
        data.put(keyAndValue[0], keyAndValue[1]);  
    }  
    final StringBuilder baseString = new StringBuilder();  
    baseString.append("notifyId=");  
    baseString.append(data.get("notifyId"));  
    baseString.append("&");  
    baseString.append("partnerOrder=");  
    baseString.append(data.get("partnerOrder"));  
    baseString.append("&");  
    baseString.append("productName=");  
    baseString.append(data.get("productName"));  
    baseString.append("&");  
    baseString.append("productDesc=");  
    baseString.append(data.get("productDesc"));  
    baseString.append("&");  
    baseString.append("price=");  
    baseString.append(data.get("price"));  
    baseString.append("&");  
    baseString.append("count=");  
    baseString.append(data.get("count"));  
    baseString.append("&");  
    baseString.append("attach=");  
    baseString.append(data.get("attach"));  
    return baseString.toString();  
}
```

2) 对baseString进行验证签名

```
/**
 * 验证签名的方法
 * @param content
 * @param sign
 * @return
 */
public static boolean doCheck(String content, String sign) {
    String charset = "utf-8";
    try {
        KeyFactory keyFactory = KeyFactory.getInstance("RSA");
        byte[] encodedKey = Base64.decodeBase64(PUB_KEY.getBytes());
        PublicKey pubKey = keyFactory
            .generatePublic(new X509EncodedKeySpec(encodedKey));
        java.security.Signature signature = java.security.Signature
            .getInstance("SHA1WithRSA");
        signature.initVerify(pubKey);
        signature.update(content.getBytes(charset));
        boolean bverify = signature.verify(Base64.decodeBase64(sign
            .getBytes()));
        return bverify;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return false;
}
```

3.2 相关 java 代码

具体的Java代码在RsaUtil.java中

名称	大小	压缩后大小	类型	修改时间	CRC32
..			文件夹		
oppo_kebi_charge_demo.php	906	607	PHP 文件	2013/12/4 15...	C12CE4E4
pay_rsa_public_key.pem	272	237	PEM 文件	2013/12/12 1...	0901C78B
可币支付回调公钥.txt	216	199	Notepad++ Doc...	2013/11/8 15...	EABA84...
可币支付回调协议.doc	178,227	143,782	Microsoft Word ...	2013/11/11 1...	06A43A...
可币支付回调验证签名原理说明.txt	560	368	Notepad++ Doc...	2013/12/6 14...	FF97C16F
可币支付回调验证签名Java_Demo.java	4,029	1,723	JAVA 文件	2013/11/11 1...	04FBC1A0
可币支付回调验证签名WebServer_J2EE_Demo.java	4,650	1,597	JAVA 文件	2014/3/13 17...	838FECC6

3.3 签名公钥

验证签名使用的公钥:

名称	大小	压缩后大小	类型	修改时间	CRC32
..			文件夹		
oppo_kebi_charge_demo.php	906	607	PHP 文件	2013/12/4 15...	C12CE4E4
pay_rsa_public_key.pem	272	237	PEM 文件	2013/12/12 1...	0901C78B
可币支付回调公钥.txt	216	199	Notepad++ Doc...	2013/11/8 15...	EABA84...
可币支付回调协议.doc	178,227	143,782	Microsoft Word ...	2013/11/11 1...	06A43A...
可币支付回调验证签名原理说明.txt	560	368	Notepad++ Doc...	2013/12/6 14...	FF97C16F
可币支付回调验证签名Java_Demo.java	4,029	1,723	JAVA 文件	2013/11/11 1...	04FBC1A0
可币支付回调验证签名WebServer_J2EE_Demo.java	4,650	1,597	JAVA 文件	2014/3/13 17...	838FECC6

四、游戏服务端回调处理说明：

CP 收到回调之后需要回写处理结果。如果同一笔订单连续通知 3 次没有收到开发者回写，该订单会从回调队列删除，并将订单状态记入数据库。

返回结果数据格式：

result=arg0&resultMsg=arg1

arg0：值为“OK”或“FAIL”。两者选其一。该值为必填字段

arg1：arg0 为“OK”时该值可以为空字符串，arg0 为“FAIL”时建议提供些有意义的信息，便于查找问题，该值非强制字段。

例如： result=OK&resultMsg=成功

result=FAIL&resultMsg=网络原因发货失败

注：收到开发者回写结果，表示 CP 已正常接收到回调，无论回写的结果是成功还是失败，将改订单从回调队列中删除，并将回调结果信息更新到数据库。