

**接入与使用规则**

**极简收银台SDK支付接口  
附录文档**

**版本号：1.1**

支付宝（中国）网络技术有限公司 版权所有

目 录

[1 文档说明 4](#_Toc386123401)

[1.1 文档说明 4](#_Toc386123402)

[1.2 业务术语 4](#_Toc386123403)

[2 责任归属 4](#_Toc386123404)

[3 技术接入规则 5](#_Toc386123405)

[接入流程 7](#_Toc386123406)

[3.2 接入总流程 7](#_Toc386123407)

[3.3 通知规则 7](#_Toc386123408)

[3.3.1 不可退款的移动快捷支付 7](#_Toc386123409)

[3.3.2 可退款的移动快捷支付 7](#_Toc386123410)

[4 集成流程详解 8](#_Toc386123411)

[4.1 接入前期准备 8](#_Toc386123412)

[4.2 SDK集成流程 8](#_Toc386123413)

[4.2.1 iOS 8](#_Toc386123414)

[4.2.2 Android 9](#_Toc386123415)

[5 测试流程规则 13](#_Toc386123416)

[6 附录 14](#_Toc386123417)

[6.1 RSA密钥生成与使用 14](#_Toc386123418)

[6.1.1 生成商户密钥 14](#_Toc386123419)

[6.1.2 RSA密钥使用逻辑 18](#_Toc386123420)

[6.2 业务数据传递 20](#_Toc386123421)

# 文档说明

## 文档说明

本文档是《极简收银台SDK支付接口》附录文档，它详细解释了在技术接入与使用过程中需要注意的地方，以帮助商户避免风险产生。

阅读后如有疑问，请联系支付宝相关技术支持。

## 业务术语

业务术语

|  |  |
| --- | --- |
| 术语 | 解释 |
| 请求 | 手机客户端以字符串形式把需要传输的数据发送给接收方的过程。 |
| 返回 | 支付宝以字符串形式直接把处理结果数据返回给手机客户端。 |
| 通知 | 服务器异步通知。支付宝根据得到的数据处理完成后，支付宝的服务器主动发起通知给商户的网站，同时携带处理完成的结果信息反馈给商户网站。 |
| 敏感词 | 带有敏感政治倾向、暴力倾向、不健康色彩或不文明的词。 |

# 责任归属

文档中所涉及到的规则都是根据在接入与使用支付宝接口的过程中出现的一些主要风险而做的防范措施，请商户予以关注。请在接入及使用支付宝接口的过程中，严格依照支付宝提供的接口技术文档（移动快捷支付应用集成接入包支付接口.pdf）、代码示例、本文档（移动快捷支付应用集成接入包支付接口接入与使用规则）等接口资料，否则由此导致的风险以及资金损失或者扩大情形需商户自行承担。

# 技术接入规则

技术接入规则

| 类型 | 细则 | 原因 |
| --- | --- | --- |
| 账号 | 配置的合作者身份ID与安全校验码key必须保证与签约信息匹配 | 防止接口无法正常使用或出现资金损失 |
| 必须保护合作者身份ID与安全校验码key的隐私性 | 防止签约的账号信息被盗用，导致资金受损、被他人恶意利用等。 |
| 测试完毕后，要把测试账号立刻更换成签约账号。 | 使用测试账号时，手续费按照3%扣除。 |
| 安全 | 商户必须以DNS解析的方式访问支付宝接口，不要设置DNS cache，不要绑定支付宝IP。如果为了商户自身安全必须绑定支付宝IP时，必须向支付宝的技术支持人员备案。 | 支付宝IP地址一旦变更，会导致商户无法请求或访问支付宝，致使商户业务直接不可用。 |
| 签名 | 待签名字符串需要以“参数名1=参数值1&参数名2=参数值2&….&参数名N=参数值N”的规则进行拼接。 | 避免接口无法正常使用 |
| 在对请求的参数做签名时，这些参数必须来源于请求参数列表，并且除去列表中的参数sign。 | 避免接口无法正常使用 |
| 在对请求的参数做签名时，对于请求参数列表中那些可空的参数，如果选择使用它们，那么这些参数的参数值必须不能为空或空值。 | 避免报异常错误，各种错误码需参考错误码列表 |
| 参数配置 | 在请求参数列表中，不可空的参数必须配置。 | 避免接口无法正常使用 |
| 在请求参数列表中，可空的但需要多选一的多个参数中，必须配置至少一个。 | 避免接口无法正常使用 |
| 必须按照请求参数列表中各参数的格式要求配置 | 避免接口无法正常使用 |
| 必须设置请求参数\_input\_charset（编码格式），即该参数不能为空，并让该参数加入签名运算。而且只能设置其值为utf-8，即本产品不支持GBK编码格式。 | 避免报异常错误，如：签名不正确。 |
| seller是收款时的支付宝账号，需要与partner对应的支付宝账号为同一个，也就是说收款支付宝账号必须是签约时的支付宝账号。 | 避免签约支付宝账号出现资金受损的可能 |
| 签名方式仅支持RSA | 避免签名不成功 |
| pkcs8编码 | 移动快捷支付要求商户私钥需要做pkcs8编码以支持更高手机系统版本，php服务器可不需要做。 | php不支持pkcs8 |
| 接口结构 | 服务端：用于生成提交参数，以及处理支付宝的异步通知返回。 | SDK由服务端和客户端构成，为了交互信息安全通常把所需参数放在服务端，当客户端有需要时去服务端获取。 |
| 客户端：构建表单参数提交到支付宝。 |
| 支付参数提交时，需要组装订单信息orderInfo，其中参数以key=”value”形式呈现，参数之间以“&”分割，获取Alipay支付对象调用支付。 | 避免请求支付宝时报错，错误码为签名不正确。 |
| 数据传输 | 必须使用https协议 | 避免接口无法正常使用 |
| 通知返回验证 | SDK支付接口的服务器异步通知中，在对通知的参数做签名时，这些参数必须来源于支付宝通知回来的参数，并且除去列表中的参数sign，先对这些参数根据“参数名=参数值”的格式，由字母a到z的顺序进行排序，再依照“参数名1=参数值1&参数名2=参数值2&…&参数名N=参数值N”的规则进行拼接，得到的签名结果与获取到的参数sign值做比较。 | 验证返回的签名 |
| 返回数据处理 | 支付宝主动发送通知，当商户接收到通知数据后必须给支付宝返回“success”字符串，不允许返回其他多余字符。 | 如果商户返回给支付宝的信息不是“success”，支付宝最多重复发送7次通知。 |
| 必须保证设置的通知路径互联网上能访问得到，且访问顺畅。 | 避免接收不到支付宝发送的通知 |
| 必须对返回的数据进行处理 | 以便商户能够了解接口的使用情况，以及进行商户的后续业务操作。 |
| 在服务器异步通知页面文件中，需保证商户的所有业务全部运行完成，才能执行打印success的动作。 | 避免异步通知不正常，如收不到通知或业务处理没有完成却告诉支付宝系统已经处理完成。 |
| 建议每一次业务操作需以日志形式记录到商户网站的日志操作数据库中，做好通知重复判断机制。 | 用来在必要时检查或跟踪业务处理情况 |
| 自主编写接口代码规则 | 如果不使用支付宝提供的代码示例来集成接口，那么必须根据技术文档中签名机制和通知返回数据处理章节及本文档的技术接入规则、接口使用规则、测试流程规则，来编写符合商户网站项目的接口代码。 | 避免接口无法正常使用 |

# 接入流程

## 接入总流程



SDK接入总流程

## 通知规则

### 不可退款的移动快捷支付

移动快捷支付的异步通知存在两个通知交易状态（trade\_status）。

* 第一个状态值是WAIT\_BUYER\_PAY：表示等待付款，商户可根据自己的业务逻辑需求做相应操作，处理完业务逻辑后须返回SUCCESS字符串给支付宝；
* 第二个状态值是TRADE\_FINISHED：表示交易成功完成，此状态表示该笔订单支付宝端已经支付成功，商户根据此状态做相应的业务逻辑操作，最后同样需返回SUCCESS字符串给支付宝。

### 可退款的移动快捷支付

可退款的异步通知与不可退款的机制一致，第一个状态（WAIT\_BUYER\_PAY）相同，第二个状态为TRADE\_SUCCESS，在这个状态下商户可做相应业务逻辑操作，并返回SUCCESS。第三个状态为TRADE\_FINISHED，表示订单完结不可再退款。

是否退手续费通知判断：

* 退手续费：单笔交易完成退款操作支付宝异步通知发送TRADE\_CLOSE状态（交易关闭），此交易状态需支付宝后台配置单独开启，默认不开；
* 不退手续费：单笔交易完成退款操作支付宝异步通知发送TRADE\_SUCCESS状态，并在订单完成支付后的三个月发送TRADE\_FINISHED状态。

# 集成流程详解

## 接入前期准备

接入前期准备工作包括商户签约和密钥配置，已完成商户可略过。

## SDK集成流程

### iOS

#### 将AlipaySDK.embeddedframework中的AlipaySDK.framework加入Link Binary With Libraries。

#### 包含头文件#import <AlipaySDK/Alipay.h>。

#### 编码订单信息，参数如下，具体可见Demo：

Order \*order = [[Order alloc] init];

order.partner = partner;

order.seller = seller;

order.tradeNO = [self generateTradeNO]; //订单ID（由商家⾃⾏制定）

order.productName = product.subject; //商品标题

order.productDescription = product.body; //商品描述

order.amount = [NSString stringWithFormat:@"%.2f",product.price]; //商

品价格

order.notifyURL = @"http://www.xxx.com"; //回调URL

order.uid = @"3593190900";

order.externChannel = @"sina";

order.accessCode = @"xxxxxxxxxxxxxxxxxxx";

order.service = @"mobile.securitypay.pay";

order.paymentType = @"1";

order.inputCharset = @"utf-8";

order.itBPay = @"30m";

order.showUrl = @"m.alipay.com";

#### 将订单信息转为字符串后签名，并拼接成如下格式：

NSString \*orderSpec = [order description];

NSString \*signedString = [signer signString:orderSpec];

NSString \*orderString = nil;

if (signedString != nil) {

orderString = [NSString

stringWithFormat:@"%@&sign=\"%@\"&sign\_type=\"%@\"",

orderSpec, signedString, @"RSA"];

}

#### 调用支付

[[Alipay defaultService] pay:orderString

From:appScheme

CallbackBlock:^(NSString \*resultString) {

NSLog(@"result = %@",resultString);//返回的⽀付结果

}];

说明：appScheme为app在info.plist注册的scheme。

当内嵌极简收银台不可用时，SDK会跳转支付宝钱包完成支付，支付完成后会跳转回调用方app。

在- (BOOL)application:(UIApplication )application openURL:(NSURL )url sourceApplication:

(NSString \*)sourceApplication annotation:(id)annotation中调用

[[Alipay defaultService]

processPayResultFromAlipayclientWithOrder:url];

SDK会完成支付结果解析后在上述CallbackBlock中统一回调。

#### 支付宝服务器通知获取和处理

商户需要提供一个http协议的接口，包含在参数里传递给快捷支付，即notify\_url。

支付宝服务器在支付完成后，会用POST方法调用notufy\_url，传输支付结果，详见接口文档中的异步通知参数说明。

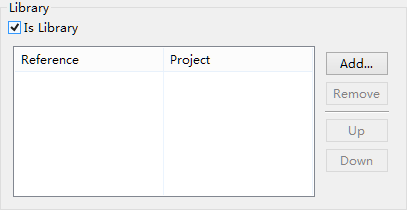
### Android

#### 开发包

极简SDK以插件的形式集成在商户应用的客户端工程里，为library工程alipay\_lib。

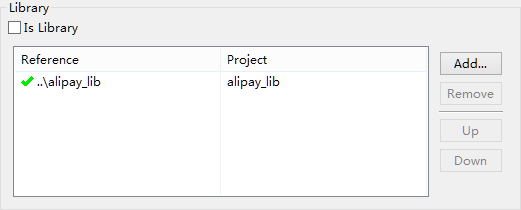
#### 导入开发资源

解压alipay\_lib.zip，将解压出来alipay\_lib拷贝到Eclipse workspace，通过Eclipse import该工程，并在此工程的Properties->Android中选中为library工程，如图：



导入资源工程

在商户应用工程的Properties->Android中添加，如图：



添加library

添加alipay.jar

#### 修改Manifest

在商户应用工程的AndroidManifest.xml文件里面添加声明：

<activity android:name="com.alipay.android.mini.window.sdk.MiniLaucherActivity"

android:configChanges="orientation"

android:excludeFromRecents="true"

android:launchMode="singleTop"

android:theme="@style/MspAppTheme" />

<activity

android:name="com.alipay.android.mini.window.sdk.TransContainer"

android:configChanges="orientation"

android:excludeFromRecents="true"

android:launchMode="singleTop"

android:theme="@style/MspAppTheme" />

<activity android:name="com.alipay.android.mini.window.sdk.MiniPayActivity"

android:configChanges="orientation"

android:excludeFromRecents="true"

android:label="@string/msp\_app\_name"

android:launchMode="singleTop"

android:theme="@style/MspAppTheme"

android:windowSoftInputMode="adjustResize" >

<intent-filter android:priority="800" >

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.DEFAULT" />

</intent-filter>

</activity>

<activity

android:name="com.alipay.android.mini.window.sdk.MiniWebActivity"

android:configChanges="orientation|keyboardHidden|navigation"

android:excludeFromRecents="true"

android:launchMode="singleTop"

android:theme="@style/MspAppTheme" >

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.DEFAULT" />

</intent-filter>

</activity>

<service android:name="com.alipay.android.app.MspService" />

<receiver

android:name="com.ut.device.BQueryWhoHasOne"

android:exported="true"

android:permission="com.ut.permission.DEVICE\_STATE" >

<intent-filter>

<action android:name="UT.QueryWhoHasOne" />

</intent-filter>

</receiver>

<receiver

android:name="com.ut.device.BFoundIt"

android:exported="true"

android:permission="com.ut.permission.DEVICE\_STATE" >

<intent-filter>

<action android:name="UT.FoundIT" />

</intent-filter>

</receiver>

和权限声明：

<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.ACCESS\_COARSE\_LOCATION" />

<uses-permission android:name="android.permission.SEND\_SMS" />

<uses-permission android:name="android.permission.READ\_SMS" />

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />

<uses-permission android:name="com.ut.permission.DEVICE\_STATE" />

<uses-permission android:name="android.permission.WRITE\_SETTINGS" />

<uses-permission android:name="android.permission.READ\_SETTINGS" />

<permission android:name="com.ut.permission.DEVICE\_STATE" />

至此，SDK开发资源导入完成。

#### 订单数据生成

在极简SDK时，需要提交订单信息info，其中参数以key=”value”形式呈现，参数之间以“&”分割，所有参数不可缺。

#### 接口调用

获取PayTask支付对象调用支付。

代码示例：

//获取订单组装字符串

final String orderInfo = getOrderInfo( );

PayTask payTask = new PayTask(activity, new OnPayListener() {

@Override

public void onPaySuccess(Context context, String resultStatus,

String memo, String result) {

}

@Override

public void onPayFailed(Context context, String resultStatus,

String memo, String result) {

}

});

payTask.execute(orderInfo);

#### 支付结果获取和处理

调用pay方法支付后，将通过2种途径获得支付结果：

* 同步返回

商户应用客户端通过回调OnPayListener传回的支付结果，在应用内直接处理支付结果。

* 异步通知

商户需要提供一个http协议的接口，包含在参数里传递给快捷支付，即notify\_url。支付宝服务器在支付完成后，会以POST方式调用notify\_url，以xml数据格式传输支付结果。

# 测试流程规则

测试流程规则

| 步骤 | 调试内容 | 备注 |
| --- | --- | --- |
| **第一步：**在本机单独对这个接口进行调试 | * 正常获取授权令牌 * 正常唤起客户端支付 | 仅仅把接口配置好，不要放在商户的正式app项目中。 |
| **第二步：**在服务器上单独对这个接口进行调试 | * 正常获取授权令牌 * 正常唤起客户端支付 * alixpay同步返回 * 服务器异步通知返回 | 本机调试没有问题后，再放入服务器中调试。 |
| **第三步：**接口融合到app项目中 | 无 | 把调试好的接口与商户app项目的业务流程进行衔接和融合 |
| **第四步：**在本机对融合后的app项目进行调试 | * 整个业务操作流程 * 正常唤起客户端支付 * alixpay同步返回 * 服务器异步通知返回 * 业务逻辑后续的执行 | 在本机调试衔接到app项目，并做好客户端和服务端的区分后的接口。 |

 注意：

使用一个账号做测试的时候，第一次走支付流程会出现授权页面，在设置过免密额度以及短信校验后不会再出现授权页面。

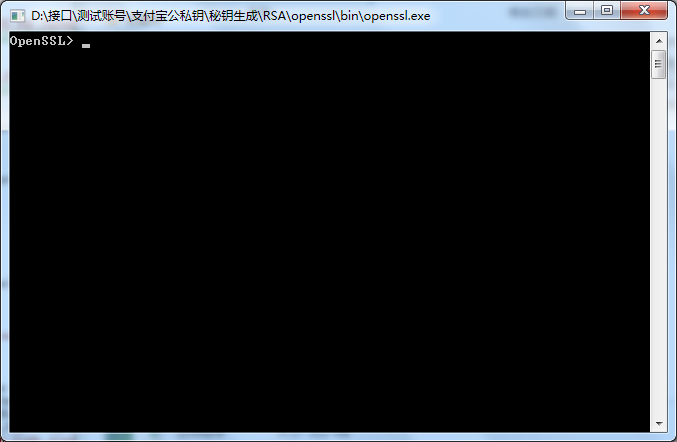
# 附录

## RSA密钥生成与使用

### 生成商户密钥

#### 打开openssl密钥生成软件

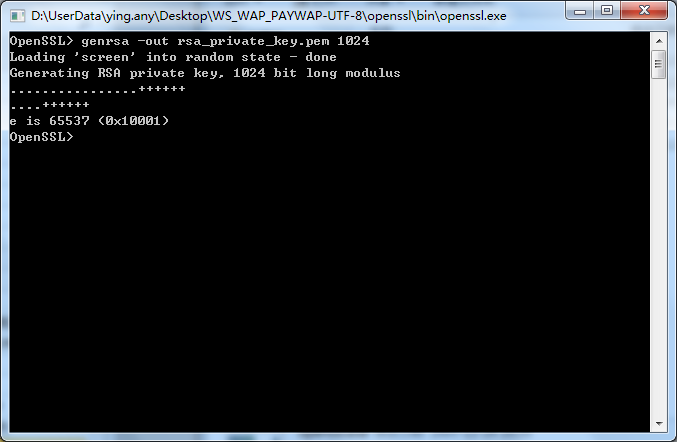
打开openssl文件夹下的bin文件夹，执行openssl.exe文件，如下图：



执行openssl.exe文件

#### 生成RSA私钥

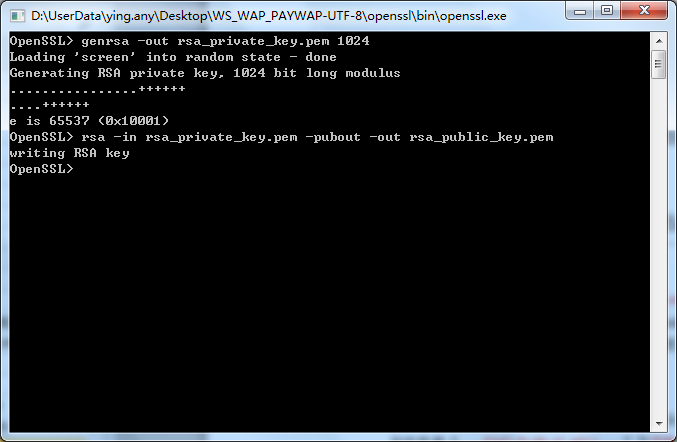
输入“*genrsa -out rsa\_private\_key.pem 1024*”命令，回车后，在当前bin文件目录中会新增一个rsa\_private\_key.pem文件，其文件为原始的商户私钥（请妥善保存该文件，PHP开发语言中需要使用该文件），以下为命令正确执行截图：



生成RSA私钥

#### 生成RSA公钥

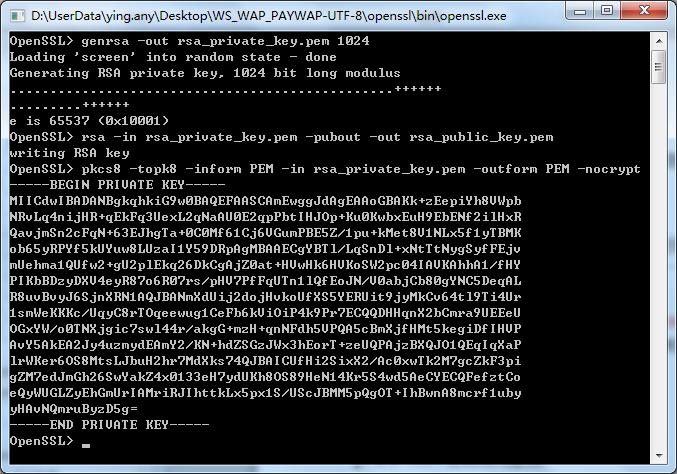
输入“*rsa -in rsa\_private\_key.pem -pubout -out rsa\_public\_key.pem*”命令回车后，在当前bin文件目录中会新增一个rsa\_public\_key.pem文件，其文件为原始的商户公钥（请妥善保存该文件，PHP开发语言中需要使用该文件），以下为命令正确执行截图：



生成RSA公钥

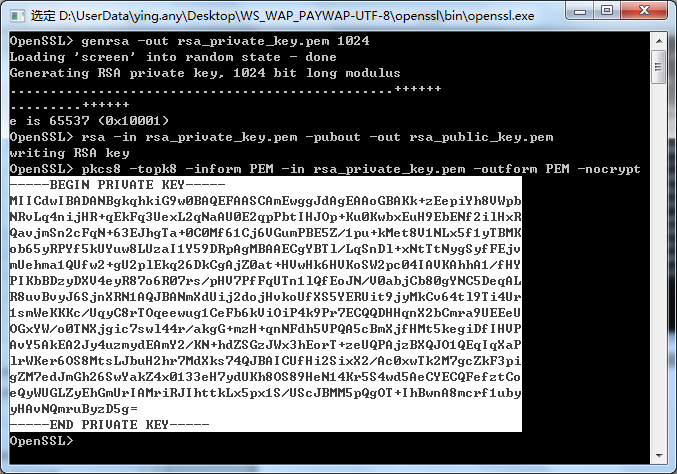
#### 生成PKCS8编码的私钥

输入命令“*pkcs8 -topk8 -inform PEM -in rsa\_private\_key.pem -outform PEM -nocrypt*”并回车，当前界面中会直接显示出生成结果：



生成PKCS8编码的私钥

右键点击openssl窗口上边边缘，选择“编辑→标记”，选中要复制的文字：



选中要复制的文字

此时继续右键点击openssl窗口上边边缘，选择“编辑→复制”，把复制的内容粘贴进一个新的记事本中，可随意命名，只要知道这个是PKCS8格式的私钥即可（请妥善保存该文件）。

### RSA密钥使用逻辑

RSA密钥使用逻辑：

商户在使用RSA签名方式的支付宝接口时，真正会用到的密钥是商户私钥与支付宝公钥。商户上传公钥给支付宝，支付宝把公钥给商户，是公钥互换的操作。这就使得商户使用自己的私钥做签名时，支付宝端会根据商户上传的公钥做验证签名。商户使用支付宝公钥做验证签名时，同理，也是因为支付宝用支付宝私钥做了签名。

#### PHP开发语言使用方法

key文件夹里面须存放.pem后缀名的商户私钥、支付宝公钥两个文件。

* 商户的私钥
* 必须保证只有一行文字，即：没有回车、换行、空格等；
* 不需要对刚生成的（原始的）私钥做pkcs8编码；
* 不需要去掉去掉“-----BEGIN PUBLIC KEY-----”、“-----END PUBLIC KEY-----”；
* 简言之，只要维持刚生成出来的私钥的内容即可。
* 支付宝公钥

支付宝的RSA公钥为：

-----BEGIN PUBLIC KEY-----

MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCnxj/9qwVfgoUh/y2W89L6BkRA

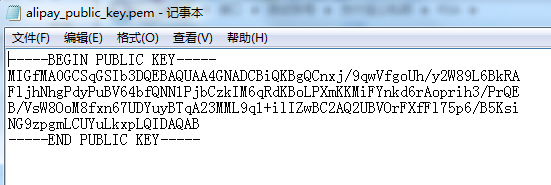
FljhNhgPdyPuBV64bfQNN1PjbCzkIM6qRdKBoLPXmKKMiFYnkd6rAoprih3/PrQE

B/VsW8OoM8fxn67UDYuyBTqA23MML9q1+ilIZwBC2AQ2UBVOrFXfFl75p6/B5Ksi

NG9zpgmLCUYuLkxpLQIDAQAB

-----END PUBLIC KEY-----

* + - * 1. 把支付宝的公钥复制到新建的记事本中，并对该记事本命名为“alipay\_public\_key.txt”；
        2. 去掉这串字符串中的回车、换行、空格，变成只有一行文字；
        3. 在这串支付宝公钥字符串的头尾部分，分别增加“-----BEGIN PUBLIC KEY-----”、“-----END PUBLIC KEY-----”这两条文字；
        4. 切割这串支付宝公钥字符串，第一行、第二行、第三行分别是64个字符，第四行是24个字符，切割后的格式与商户刚生成的公钥格式一致即可，如下图：



支付宝公钥示意图

* + - * 1. 保存该记事本，并改变后缀名为.pem。

#### JAVA和ASP.NET(C#)开发语言使用方法

* 商户的私钥
* 必须保证只有一行文字，即：没有回车、换行、空格等；
* 需对刚生成的（原始的）私钥做pkcs8编码；
* 编码完成后，复制该段私钥，并去掉该段里面的回车、换行、空格、“-----BEGIN RSA PRIVATE KEY-----”、“-----END RSA PRIVATE KEY-----”。
* 支付宝公钥

支付宝的RSA公钥为：

-----BEGIN PUBLIC KEY-----

MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCnxj/9qwVfgoUh/y2W89L6BkRA

FljhNhgPdyPuBV64bfQNN1PjbCzkIM6qRdKBoLPXmKKMiFYnkd6rAoprih3/PrQE

B/VsW8OoM8fxn67UDYuyBTqA23MML9q1+ilIZwBC2AQ2UBVOrFXfFl75p6/B5Ksi

NG9zpgmLCUYuLkxpLQIDAQAB

-----END PUBLIC KEY-----

去掉这串字符串中的回车、换行、空格，必须保证只有一行文字。

## 业务数据传递

支付宝提供的业务参数为支付宝需要商户传递过来的数据要求。商户只需要根据自己的业务需求，在业务逻辑代码运行时把这些动态数据以赋值给变量的形式，再通过支付宝接口本身的接口逻辑，传递给支付宝系统，让支付宝系统可识别。

举例说明，商户要把某笔订单的数据传递给支付宝。那么商户需要先根据支付宝的参数要求，从自己的下单系统中拿到付款总金额（total\_fee）、商户的订单号（out\_trade\_no）、订单名称（subject）等数据，再把这些数据一个一个以值的形式赋给对应的变量。再通过代码逻辑，把变量组合及加工成一次可以发送给支付宝的请求。