



# 接入与使用规则

---

## 移动支付接口智能 SDK 版

### 附录文档

版本号：1.0

## 目 录

<b>1 文档说明</b>	<b>4</b>
1.1 文档说明	4
1.2 业务术语	4
<b>2 责任归属</b>	<b>4</b>
<b>3 技术接入规则</b>	<b>5</b>
<b>4 接入流程</b>	<b>7</b>
4.1 接入总流程	7
4.2 通知规则	7
4.2.1 不可退款的移动支付	7
4.2.2 可退款的移动支付	7
<b>5 集成流程详解</b>	<b>8</b>
5.1 接入前期准备	8
5.2 SDK 集成流程	8
5.2.1 iOS	8
5.2.2 Android	10
<b>6 测试流程规则</b>	<b>14</b>
<b>7 附录</b>	<b>14</b>
7.1 如何获得 PID 与密钥	14
7.2 RSA 密钥生成与使用	16
7.2.1 生成商户密钥	16

---

7.2.2 RSA 密钥使用逻辑 .....	21
7.3 业务数据传递.....	23

# 1 文档说明

## 1.1 文档说明

本文档是《移动支付接口智能 SDK 版》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。

## 1.2 业务术语

表1-1 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。

# 2 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（移动支付接口智能 SDK 版.pdf）、代码示例、本文档（移动支付接口智能 SDK 版接入与使用规则）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。

### 3 技术接入规则

表3-1 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

类型	细则	原因
pkcs8 编码	移动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

## 4 接入流程

### 4.1 接入总流程

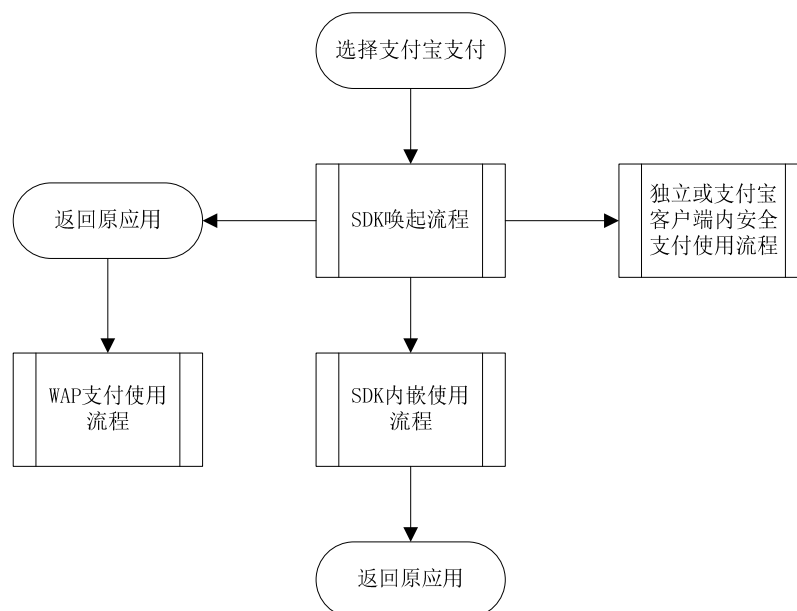


图4-1 SDK 接入总流程

### 4.2 通知规则

#### 4.2.1 不可退款的移动支付

移动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。

#### 4.2.2 可退款的移动支付

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。

## 5 集成流程详解

### 5.1 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。

### 5.2 SDK集成流程

#### 5.2.1 iOS

1. 添加必要的头文件、bundle、库文件

复制

WS\_SECURE\_PAY\_SDK\iOS\_SDK\AlipaySdkDemo\AlipaySdkDemo\alipay 目录下的

```
AlixPayLib.a  
Alixpay.bundle  
AlixLibService.h
```



注意：

AlixPayLib\_Arc.a 为支持 ARC 的 lib，如果商户程序设置为自动引用计数，则替换为该 lib。

快捷支付 SDK 核心库（必选）

```
PartnerConfig.h  
DataSigner.h  
DataVerifier.h  
AlipayRsaLib.a  
libssl.a  
libcrypto.a
```



拷贝添加 JSON 开源库（SBJSON）源文件（必选）



注意：

为了避免商户也使用 SBJSON 造成的重定义错误，SDK 将 SBJSON 的实现挪移到了商户程序客户端，因此如果提示未定义请自助添加 DEMO 中 \AlipaySdkDemo\alipay\json 目录下 SBJSON 的实现。

\AlipaySdkDemo\alipay\json 文件夹下面的源文件：

#### Framworks

```
CFNetwork.framework
SystemConfiguration.framework
Security.framework
```

本地签名验签（客户端本地签名验签时需要，可选）

订单生成、结果处理可参考 demo 工程下 Alipay 目录中的：

```
AlixpayOrder.h
AlixpayOrder.m
AlixpayResult.h
AlixpayResult.m
```

## 2. 添加自定义 URL Scheme

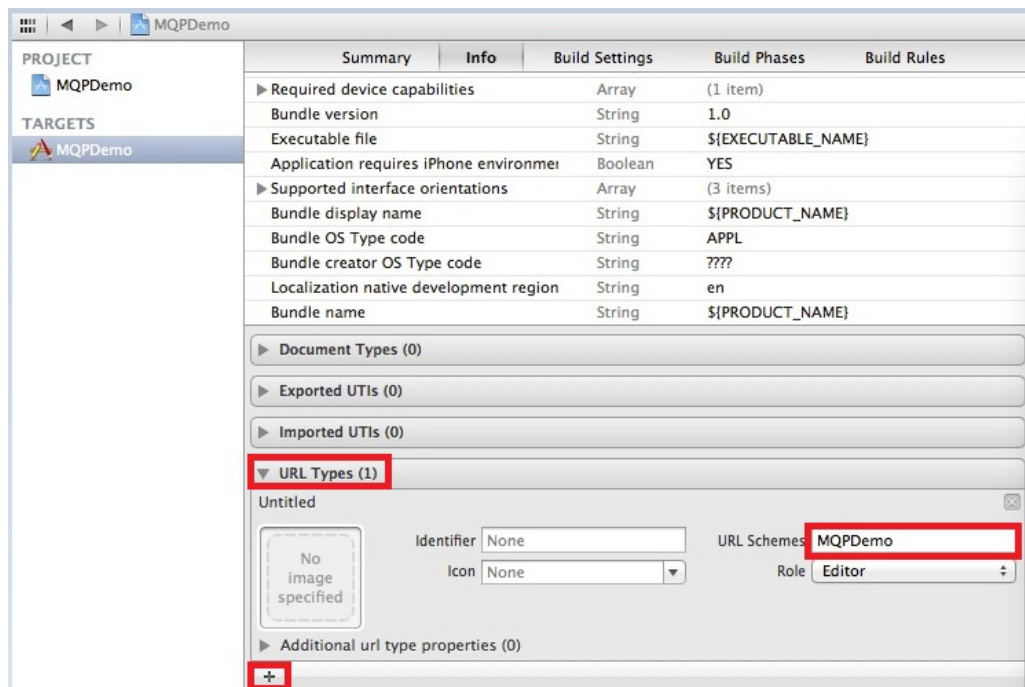


图5-1 添加自定义 URL Scheme

URL Scheme 在回调结果使用，建议起名稍复杂一些，尽量避免同其他程序冲突。

### 3. 订单数据生成

在调用快捷支付 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。

### 4. 调用支付接口

代码如下：

```
(void)payOrder:(NSString*)order AndScheme:(NSString*)scheme selector:(SEL)selector target:(id)target;
```

### 5. 支付结果获取和处理

调用快捷支付后，将通过 2 种途径获得支付结果：

#### (1) payOrder 方法的客户端返回

- 应用内 wap 支付返回结果：通过对象 target 的 selector 方法返回结果。
- 应用外快捷支付返回结果：通过 AppDelegate 的方法 handleOpenURL 获取（详情可参考 demo 中 AppDelegate.m）的 handleOpenURL 方法，支付结果的详细信息详见接口文档中的同步返回参数说明。

需要同时实现上面两个接口，否则有可能接收不到回调结果。

#### (2) 支付宝服务器通知

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，会用 POST 方法调用 notify\_url，传输支付结果，详见接口文档中的异步通知参数说明。

## 5.2.2 Android

### 1. 开发包

快捷支付 SDK 以插件的形式集成在商户应用的客户端工程里，主要包括三部分：

- 核心插件 jar 包 alipay.jar
- 资源 Library 工程 alipay\_lib
- 移动支付 apk 包 alipay\_msp.apk（可选）



注意：

如果商户应用自带 alipay\_msp.apk，不可修改此文件名，并请将其放在商户工程的 assets 目录下。

2. 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:

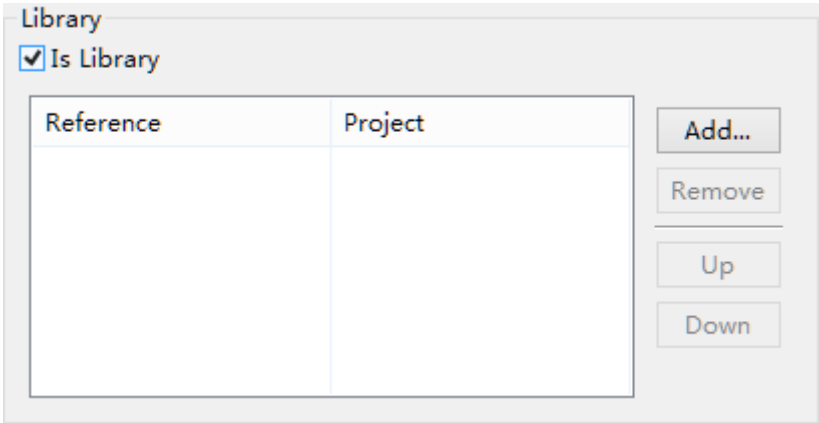


图5-2 导入资源工程

在商户应用工程的 Properties->Android 中添加, 如图:

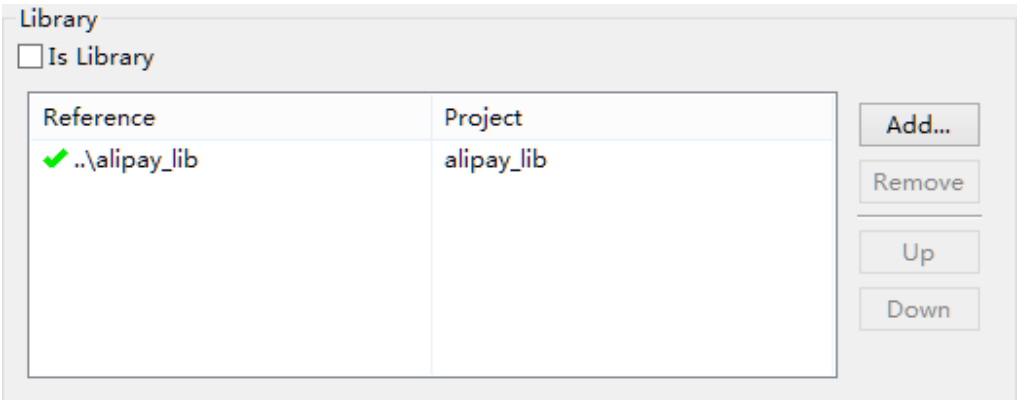


图5-3 添加 library

将 alipay.jar 复制至商户应用工程的 libs 目录下, 通过 Java Build Path 导入进工程, 如图:

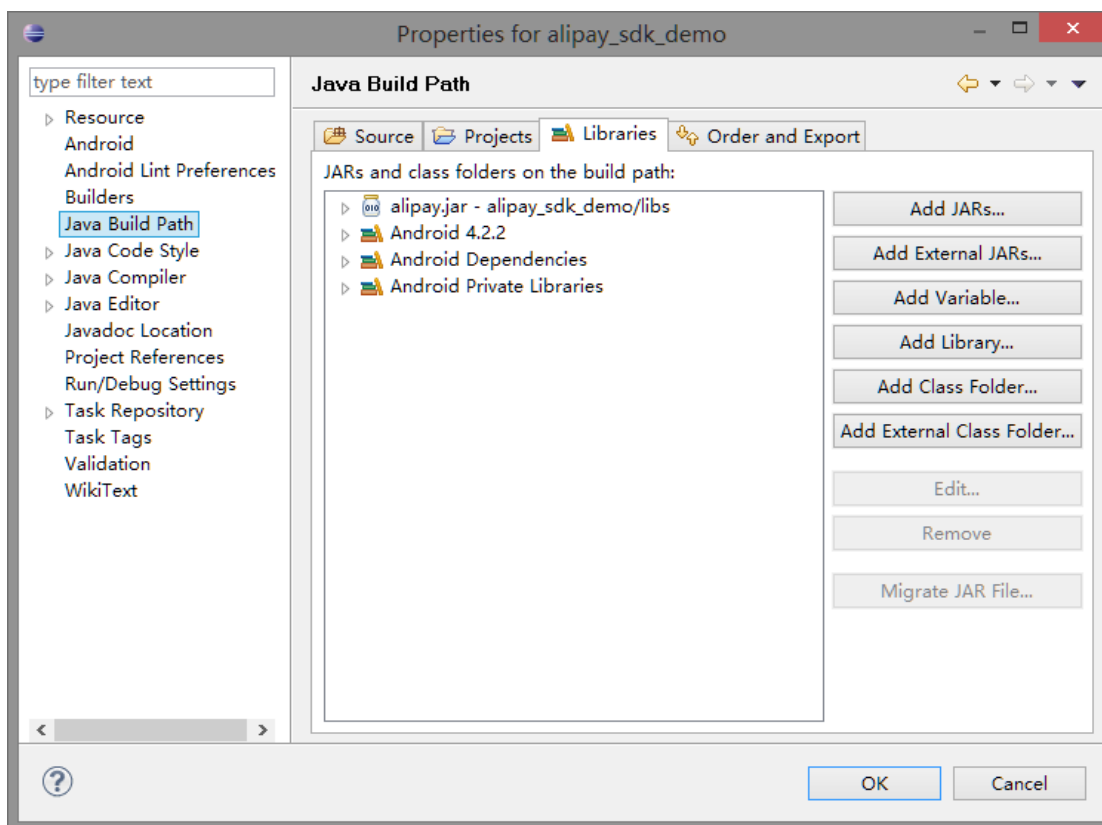


图5-4 添加 alipay.jar

### 3. 修改Manifest

在商户应用工程的 `AndroidManifest.xml` 文件里面添加 `Activity` 声明：

```
<activity
    android:name="com.alipay.android.app.sdk.WapPayActivity"
    android:screenOrientation="portrait">
</activity>
```

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" />
```

至此，SDK 开发资源导入完成。

### 4. proguard配置

如果需要对代码进行混淆，请保证 `project.properties` 文件中添加了此行内容：

```
proguard.config=proguard.cfg
```

然后在 `proguard.cfg` 文件中加入下面的内容：

```
-keep class com.alipay.android.app.IAliPay{*;}
-keep class com.alipay.android.app.IAlipay{*;}
```

```
-keep class com.alipay.android.app.IRemoteServiceCallback{*;}
```

## 5. 订单数据生成

在调用快捷支付 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。

## 6. 接口调用

获取 Alipay 支付对象调用支付，此接口方法实现为同步调用，将阻塞商户应用 UI 线程，所以调用此接口需启动新线程，并使用 **looper** 为 **main looper** 的 **Handler** 对象与 UI 线程传递消息。

代码示例：

```
//获取订单组装字符串
final String orderInfo = getOrderInfo( );

new Thread() {
    public void run() {

        //获取 Alipay 对象，构造参数为当前 Activity 和 Handler 实例对象
        Alipay alipay = new Alipay(DemoActivity.this, mHandler);

        //调用 pay 方法，将订单信息传入
        String result = alipay.pay(orderInfo);

        //处理返回结果
    }
}.start();
```

## 7. 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

商户应用客户端获取 **pay( )** 返回的字符串信息，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。

## 6 测试流程规则

表6-1 测试流程规则

步骤	调试内容	备注
<b>第一步：</b> 在本机单独对这个接口进行调试	<ul style="list-style-type: none"><li>正常获取授权令牌</li><li>正常唤起客户端支付</li></ul>	仅仅把接口配置好，不要放在商户的正式 app 项目中。
<b>第二步：</b> 在服务器上单独对这个接口进行调试	<ul style="list-style-type: none"><li>正常获取授权令牌</li><li>正常唤起客户端支付</li><li>alipay 同步返回</li><li>服务器异步通知返回</li></ul>	本机调试没有问题后，再放入服务器中调试。
<b>第三步：</b> 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
<b>第四步：</b> 在本机对融合后的 app 项目进行调试	<ul style="list-style-type: none"><li>整个业务操作流程</li><li>正常唤起客户端支付</li><li>alipay 同步返回</li><li>服务器异步通知返回</li><li>业务逻辑后续的执行</li></ul>	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。

## 7 附录

### 7.1 如何获得PID与密钥

**步骤1：** 使用签约支付宝账号登录“商家服务”平台中的“我的商家服务”（<https://b.alipay.com/order/serviceIndex.htm>）。



图7-1 我的商家服务

**步骤2：** 点击“查询 PID、Key”，即可查看到签约支付宝账号、合作者身份 ID（PID）的信息。



图7-2 查询 PID

**步骤3：** 输入支付密码，查询 key。



图7-3 查询 Key

**注意：**

输入支付密码需要安装数字证书或支付盾。

## 7.2 RSA密钥生成与使用

### 7.2.1 生成商户密钥

#### 1. 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：



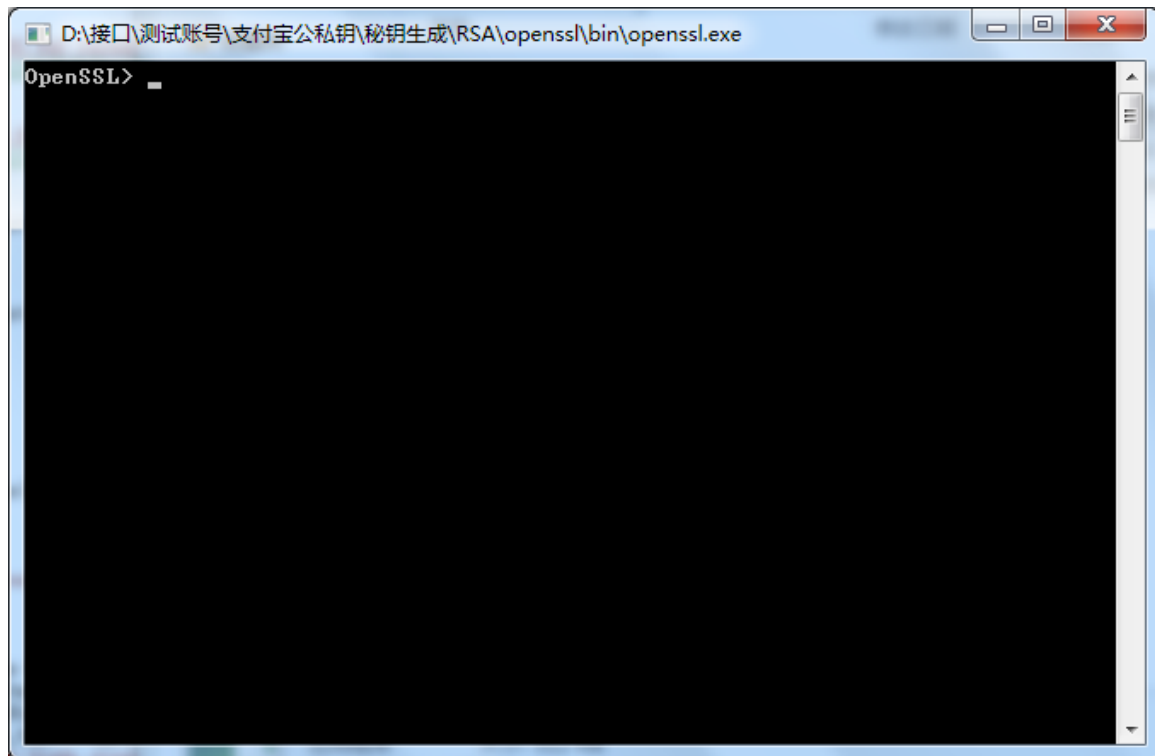


图7-4 执行 openssl.exe 文件

## 2. 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（**请妥善保管该文件**，PHP 开发语言中需要使用该文件），以下为命令正确执行截图：

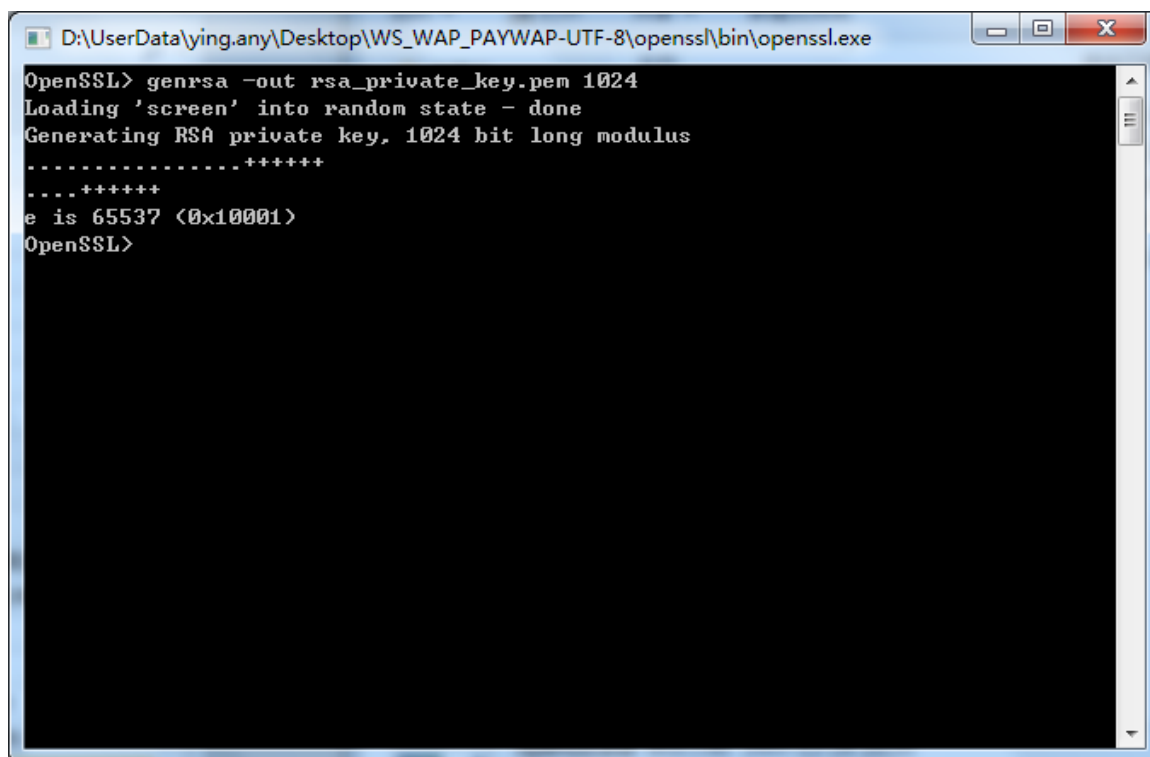
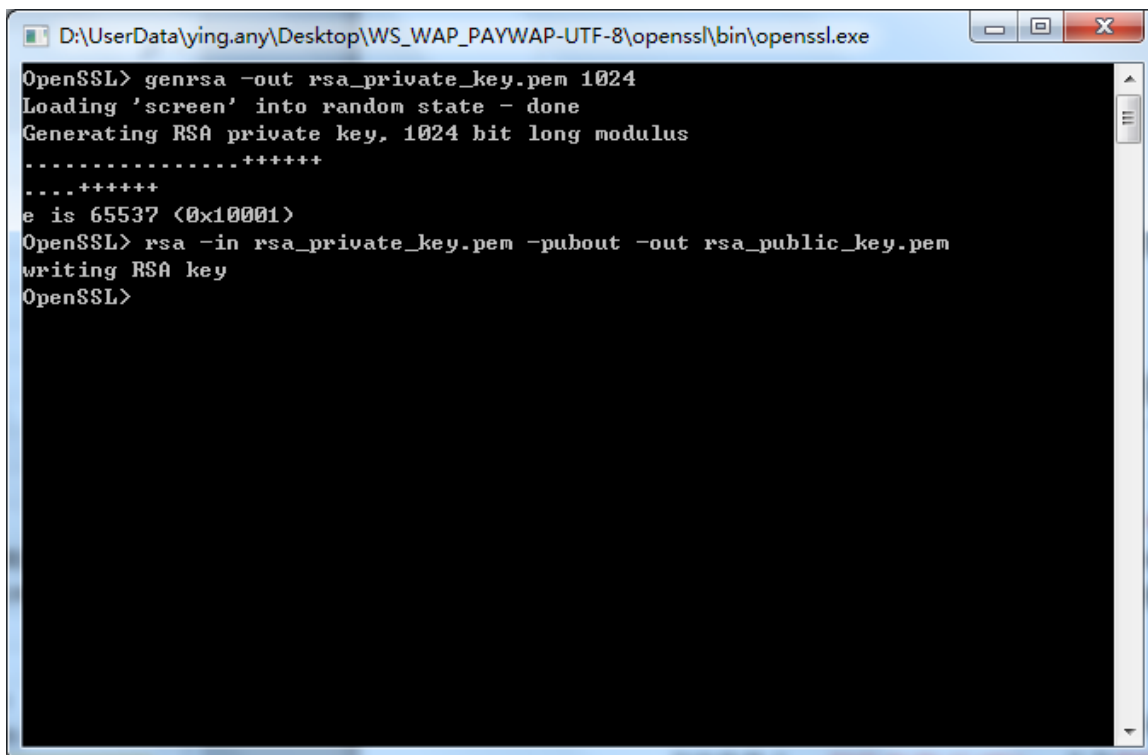


图7-5 生成 RSA 私钥

### 3. 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：

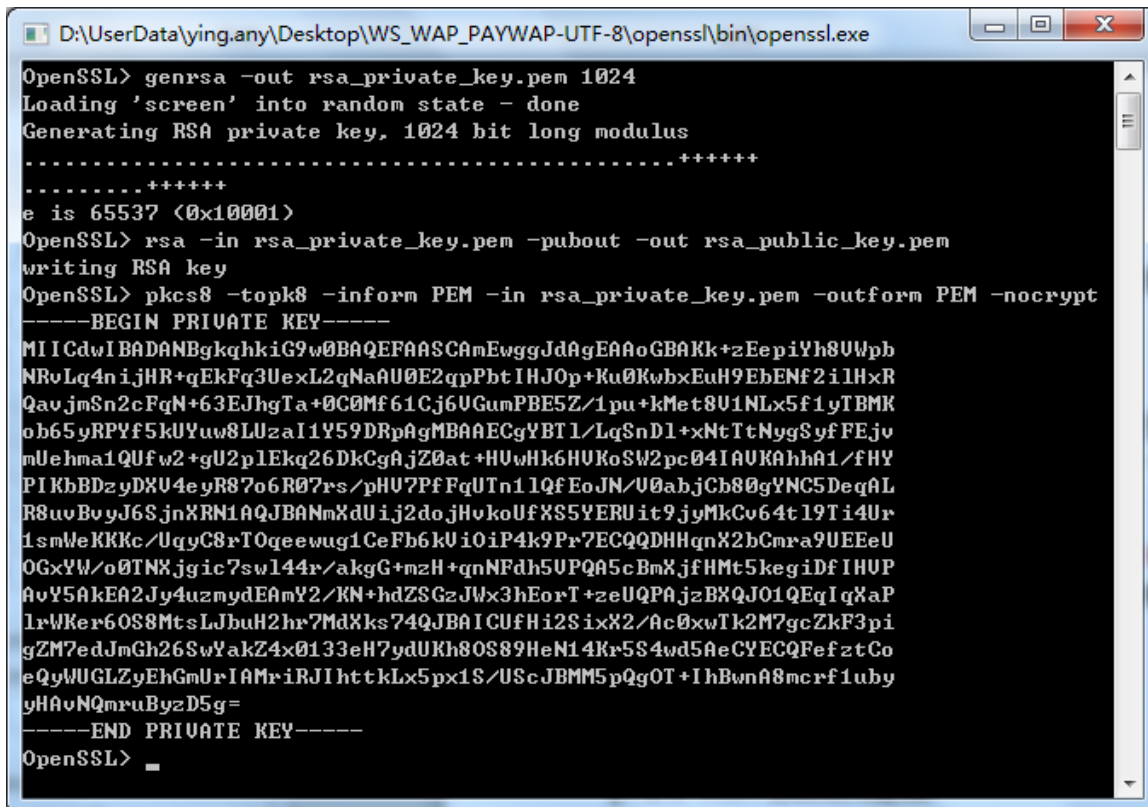


```
D:\UserData\ying.any\Desktop\WS_WAP_PAYWAP-UTF-8\openssl\bin\openssl.exe
OpenSSL> genrsa -out rsa_private_key.pem 1024
Loading 'screen' into random state - done
Generating RSA private key, 1024 bit long modulus
.....+++++
....+++++
e is 65537 (0x10001)
OpenSSL> rsa -in rsa_private_key.pem -pubout -out rsa_public_key.pem
writing RSA key
OpenSSL>
```

图7-6 生成 RSA 公钥

#### 4. 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：



```
D:\UserData\ying.any\Desktop\WS_WAP_PAYWAP-UTF-8\openssl\bin\openssl.exe
OpenSSL> genrsa -out rsa_private_key.pem 1024
Loading 'screen' into random state - done
Generating RSA private key, 1024 bit long modulus
.....+++++
.....+++++
e is 65537 (0x10001)
OpenSSL> rsa -in rsa_private_key.pem -pubout -out rsa_public_key.pem
writing RSA key
OpenSSL> pkcs8 -topk8 -inform PEM -in rsa_private_key.pem -outform PEM -nocrypt
-----BEGIN PRIVATE KEY-----
MIICGwIBADANBgkqhkiG9w0BAQEFAASCAmEwggJdAgEAAoGBAKk+zEepiYh8UWpb
NRvLq4nijiHR+qEkFq3UexL2qNaAU0E2qpPbtIHJO+Ku0KwbxEuH9EbENf2ilHxR
QavjmSn2cFqN+63EJhgTa+0C0MF61Cj6UGumPBE5Z/1pu+kMet8U1NLx5f1yTBMK
ob65yRPYf5kUYuw8LUza11Y59DRpAgMBAAGCgYBT1/LqSnD1+xNtTtNyqSyfFEju
mUehma1QUfw2+gU2p1Ekq26DkCgA jZ0at+HUwHk6HUkoSW2pc04IAUkAhha1/fHY
PIKbBDzyDXU4eyR87o6R07rs/pHU7PffqUTn1lQfEoJN/U0abjCb80gYNC5DeqAL
R8uvBoyJ6SjnXRN1AQJBANmXduij2dojHvkoUfXS5YERUit9jyMkCv64t19Ti4Ur
ismWeKKKc/UqyC8rT0qeewug1CeFb6kVi0iP4k9Pr7ECQQDHHqnX2bCmra9UEEeU
OGxYW/o0TNXjgic7swl44r/akgG+mzH+qnNFDh5UPQA5cBmXjfhMt5kegidfIHUP
AvY5AkEA2Jy4uzmydEAmY2/KN+hdZSGzJWx3hEorT+zeUQPAjzBXQJO1QEgIqXaP
lrWker60S8MtsLJbuH2hr7MdXks74QJBAlCUfHi2S ixX2/Ac0xwTk2M7gcZkF3pi
gZM7edJmGh26SwYakZ4x0133eH7ydUKh80S89HeN14Kr5S4wd5AeCYECQFefztCo
eQyWUGLZyEhGmUrIAMrIRJIhttkLx5px1S/UScJBMM5pQgOT+IhBwnA8mcrf1uby
yHA0NQmruByzD5g=
-----END PRIVATE KEY-----
OpenSSL>
```

图7-7 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：

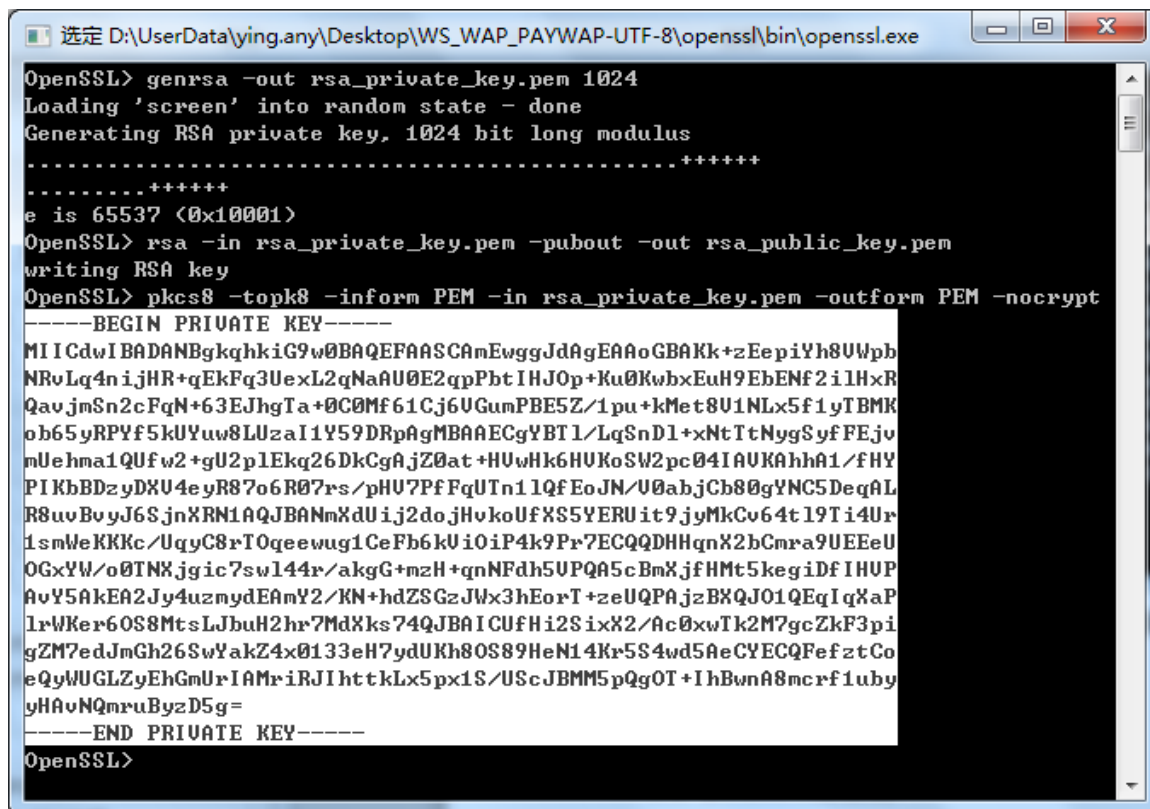


图7-8 选中要复制的文字

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](#)）。

## 7.2.2 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。

### 1. 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：

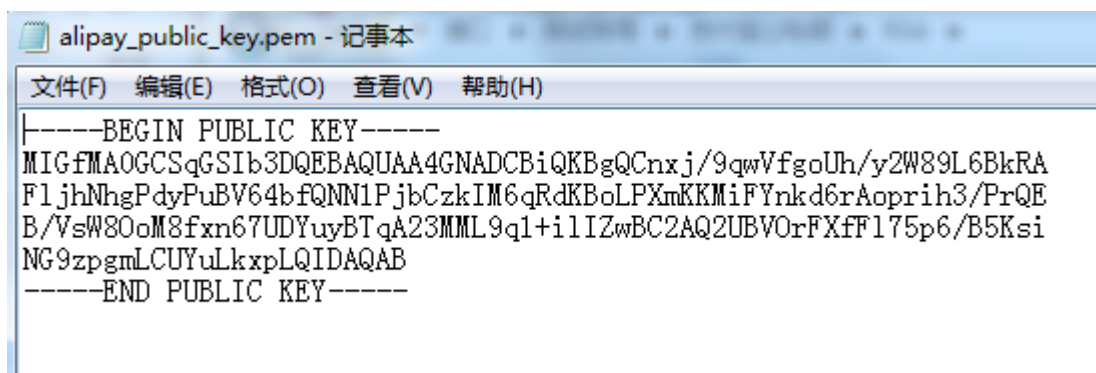


图7-9 支付宝公钥示意图

- (5) 保存该记事本，并改变后缀名为.pem。

## 2. 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。

## 7.3 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。