接入与使用规则

支付宝钱包支付接口开发包 2.0 标准版

附录文档

版本号：1.6

目 录

1 文档说明

1.1 文档说明

本文档是《支付宝钱包支付接口开发包 2.0 标准版》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(支付宝钱包支付接口开发包 2.0 标准版.pdf)、代码示例、本文档（支付宝钱包支付接口开发包 2.0 标准版接入与使用规则）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	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、sign_type。	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。	调用 RSA 密钥时，如果是通过 pem 文件解析方式，则无需 pkcs8 编码。

类型	细则	原因
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。	开发包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	开发包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

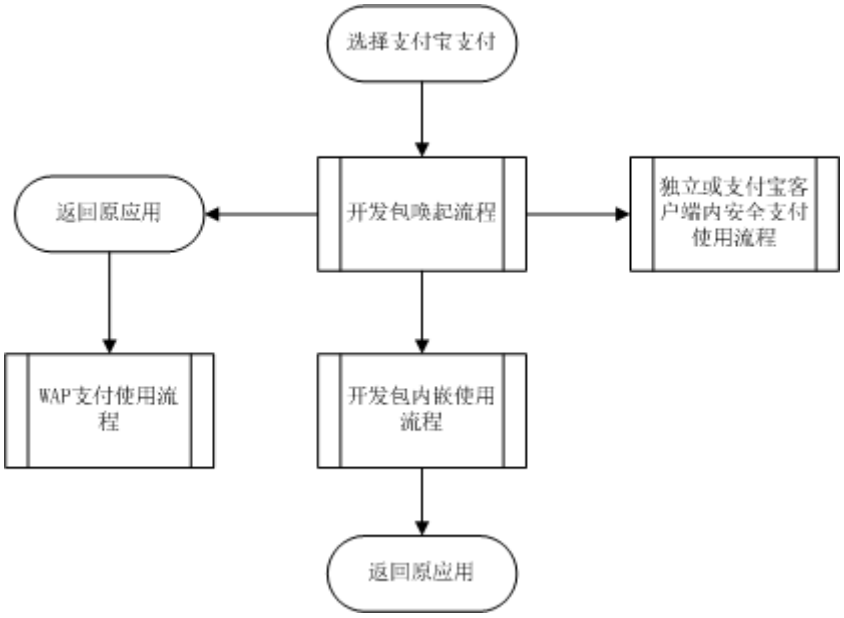

图4-1 开发包接入总流程

4.2 通知规则

4.2.1 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。

4.2.2 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。

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 开发包集成流程

5.2.1 iOS

解压接口压缩文件（文件名是 WS_MOBILE_PAY_SDK_BASE.zip），找到 iOS 的压缩文件（文件名是支付宝钱包支付开发包标准版(iOS).zip）。

1. 导入代码

步骤1： 启动 IDE（如 Xcode），把 iOS 包中的压缩文件中以下文件拷贝到项目文件夹下，并导入到项目工程中。

```
AlipaySDK.bundle
AlipaySDK.framework
```

依赖添加完毕后，如下图所示结构。

图5-1 添加依赖

步骤2： 在需要调用 AlipaySDK 的文件中，增加头文件引用。

```
#import <AlipaySDK/AlipaySDK.h>
```

步骤3： **如果你的 app 基于 9.0 编译，那么为了适配 iOS9.0 中的 App Transport Security(ATS)对 http 的限制，这里需要对支付宝的请求地址 alipay.com 做例外，在 app 对应的 info.list 中添加如下配置（文中以 XML 格式描述）。**

```
<key>NSAppTransportSecurity</key>
    <dict>
        <key>NSExceptionDomains</key>
        <dict>
            <key>alipay.com</key>
            <dict>
                <!--Include to allow subdomains-->
                <key>NSIncludesSubdomains</key>
                <true/>
                <!--Include to allow insecure HTTP requests-->
                <key>NSTemporaryExceptionAllowsInsecureHTTPLoads</key>
                <true/>
                <!--Include to specify minimum TLS version-->
                <key>NSTemporaryExceptionMinimumTLSVersion</key>
                <string>TLSv1.0</string>
                <key>NSTemporaryExceptionRequiresForwardSecrecy</key>
                <false/>
            </dict>
        </dict>
    </dict>
```

步骤4： 配置请求信息。

```
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.service = @"mobile.securitypay.pay";
order.paymentType = @"1";
order.inputCharset = @"utf-8";
order.itBPay = @"30m";

//应用注册 scheme,在 AlixPayDemo-Info.plist 定义 URL types
NSString *appScheme = @"alisdkdemo";

//将商品信息拼接成字符串
NSString *orderSpec = [order description];
NSLog(@"orderSpec = %@",orderSpec);

//获取私钥并将商户信息签名,外部商户可以根据情况存放私钥和签名,只需要遵循 RSA 签名规范,
并将签名字符串 base64 编码和 UrlEncode
id<DataSigner> signer = CreateRSADataSigner(privateKey);
NSString *signedString = [signer signString:orderSpec];

//将签名成功字符串格式化为订单字符串,请严格按照该格式
NSString *orderString = nil;
if (signedString != nil) {
   orderString                          =                          [NSString
stringWithFormat:@"%@&sign=\"%@\"&sign_type=\"%@\"",
                    orderSpec, signedString, @"RSA"];

   [[AlipaySDK  defaultService]  payOrder:orderString  fromScheme:appScheme
callback:^(NSDictionary *resultDic) {
//【callback 处理支付结果】
          NSLog(@"reslut = %@",resultDic);
   }];

   [tableView deselectRowAtIndexPath:indexPath animated:YES];
}
```

详细可参见 Demo 中示例文件

- AliSDKDemo\APViewController.h
- AliSDKDemo\APViewController.m
- AliSDKDemo\Order.h
- AliSDKDemo\Order.m

步骤5： 配置支付宝客户端返回 url 处理方法。

（外部存在支付包钱包，支付宝钱包将处理结果通过 url 返回。）

如示例 AliSDKDemo\APAppDelegate.m 文件中，增加引用代码：

```
#import <AlipaySDK/AlipaySDK.h>
```

在@implementation AppDelegate 中增加如下代码：

```
- (BOOL)application:(UIApplication *)application openURL:(NSURL *)url
sourceApplication:(NSString *)sourceApplication annotation:(id)annotation
{

    //如果极简开发包不可用，会跳转支付宝钱包进行支付，需要将支付宝钱包的支付结果回传给开
发包
    if ([url.host isEqualToString:@"safepay"]) {
        [[AlipaySDK defaultService] processOrderWithPaymentResult:url
standbyCallback:^(NSDictionary *resultDic) {
    //【由于在跳转支付宝客户端支付的过程中，商户 app 在后台很可能被系统 kill 了，所以 pay
接口的 callback 就会失效，请商户对 standbyCallback 返回的回调结果进行处理，就是在这个方
法里面处理跟 callback 一样的逻辑】
            NSLog(@"result = %@",resultDic);
        }];
    }
    if ([url.host isEqualToString:@"platformapi"]){//支付宝钱包快登授权返回
authCode

        [[AlipaySDK defaultService] processAuthResult:url
standbyCallback:^(NSDictionary *resultDic) {
            //【由于在跳转支付宝客户端支付的过程中，商户 app 在后台很可能被系统 kill 了，
所以 pay 接口的 callback 就会失效，请商户对 standbyCallback 返回的回调结果进行处理，就
是在这个方法里面处理跟 callback 一样的逻辑】
            NSLog(@"result = %@",resultDic);
        }];
    }
    return YES;
}
```

2. 针对Demo的运行注意

(1) 关于签名代码问题

- AliSDKDemo\Util 及下面所有文件
- AliSDKDemo\openssl 及下面所有文件
- libcrypto.a
- libssl.a

这些文件是为示例签名所在客户端本地使用。出于安全考虑，请商户尽量把私钥保存在服务端，在服务端进行签名验签。

(2) 如果遇到运行后报错，类似于以下提示信息：

```
Cannot find interface declaration for 'NSObject', superclass of 'Base64'
```

那么需要打开报错了的文件，增加头文件。

```
#import <Foundation/Foundation.h>
```

(3) 如果商户要在某个文件中使用支付宝的开发包类库，需增加引用头文件。

```
#import <AlipaySDK/AlipaySDK.h>
```

(4) 点击项目名称，点击“Build Settings”选项卡，在搜索框中，以关键字“search”搜索，对“Header Search Paths”增加头文件路径：$(SRCROOT)/项目名称。如果头文件信息已增加，可不必再增加。

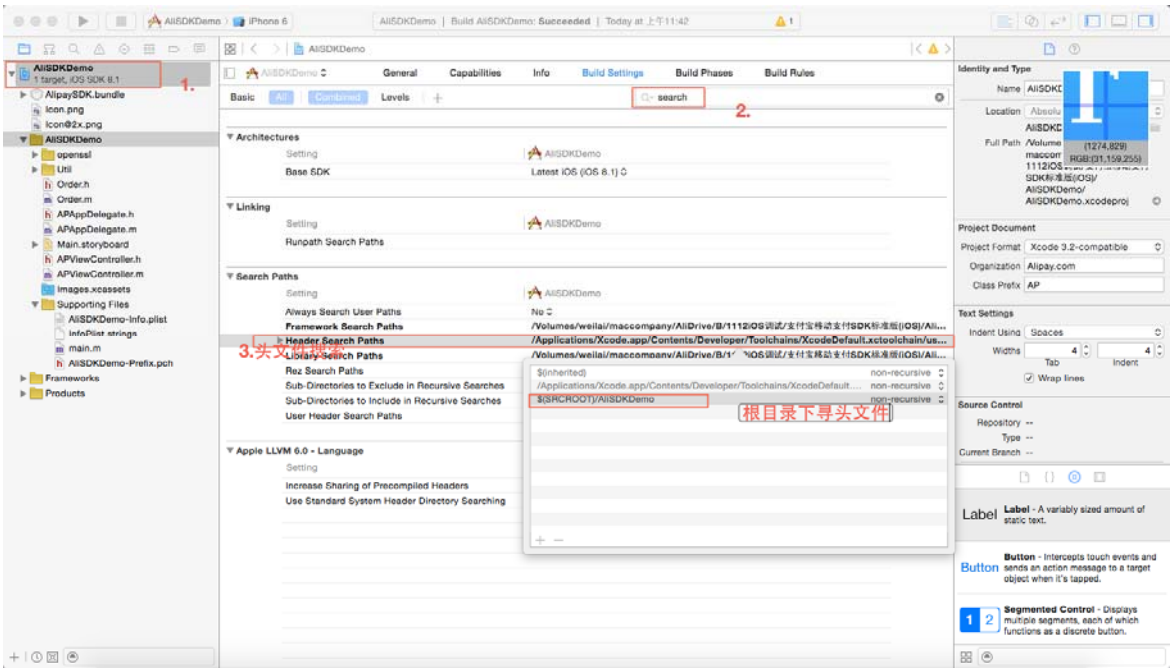

图5-2 增加头文件信息

(5) 点击项目名称，点击“Build Phases”选项卡，在“Link Binary with Librarles”选项中，新增“AlipaySDK.framework”和“SystemConfiguration.framework”两个系统库文件。如果商户项目中已有这两个库文件，可不必再增加。

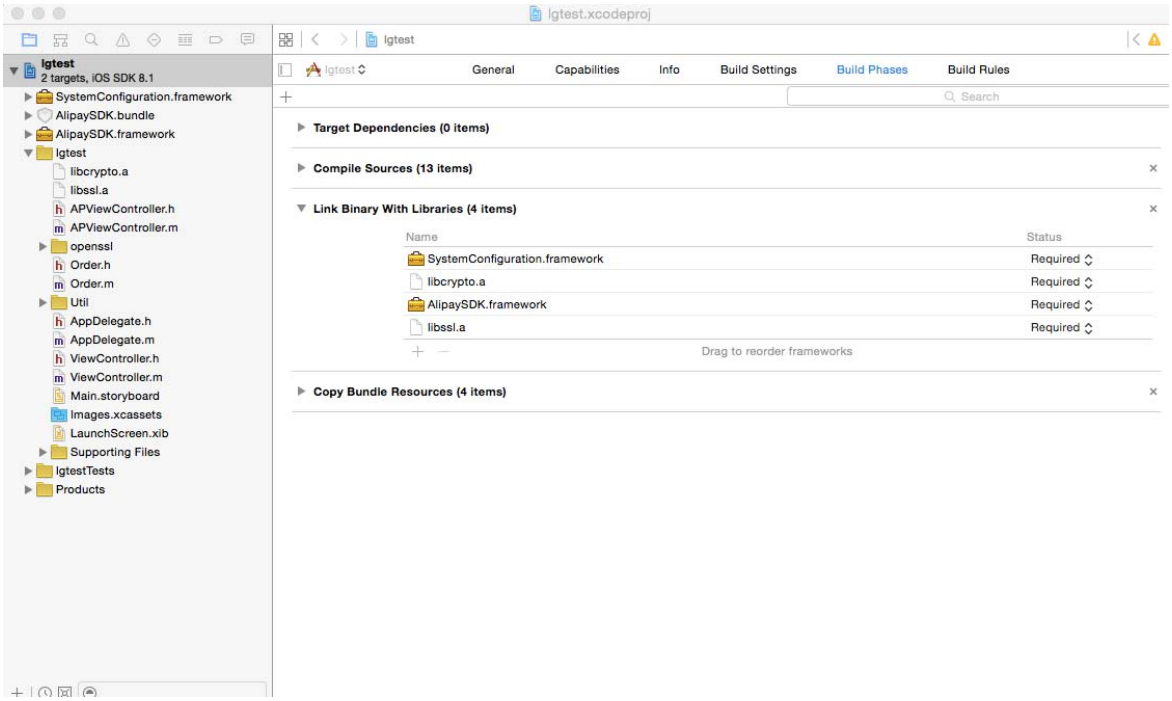

图5-3 增加系统库文件

(6) 点击项目名称，点击“Info”选项卡，在“URL Types”选项中，点击“+”，在“URL Schemes”中输入“alisdkdemo”。“alisdkdemo”来自于文件“APViewController.m”的 NSString *appScheme = @"alisdkdemo";。

⚠ 注意：

这里的 URL Schemes 中输入的 alisdkdemo，为测试 demo，实际商户的 app 中要填写独立的 scheme，建议跟商户的 app 有一定的标示度，要做到和其他的商户 app 不重复，否则可能会导致支付宝返回的结果无法正确跳回商户 app。

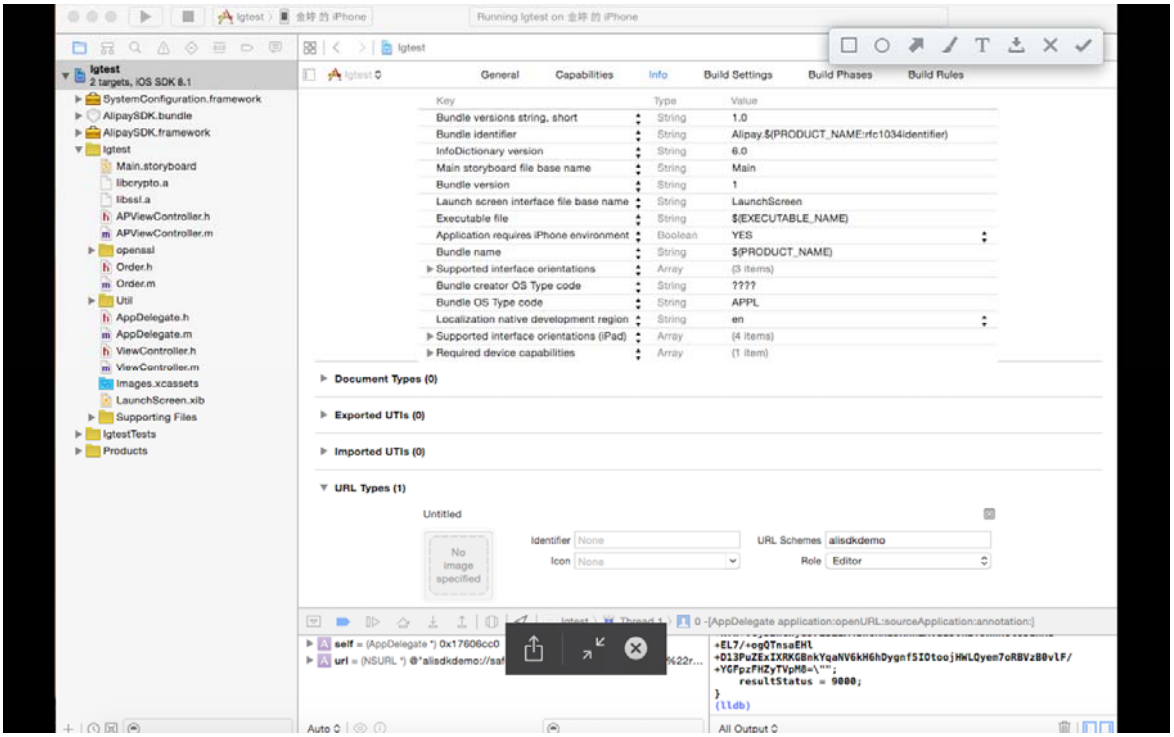

图5-4 配置 URL Schemes

3. 配置基本信息

打开“APViewController.m”文件，对以下三个参数进行编辑

```
NSString *partner = @"";
NSString *seller = @"";
NSString *privateKey = @"";
```

表5-1 IOS 基本信息配置

参数	含义
partner	合作身份者ID，以 2088 开头由 16 位纯数字组成的字符串。请参考“7.1 如何获得PID与密钥”。
seller	支付宝收款账号，手机号码或邮箱格式。
private_key	商户方的私钥，pkcs8 格式。请参考“7.2 RSA密钥生成与使用”。

⚠ 注意：

这些参数配置是为客户端签名功能服务的，仅作为示例使用。商户在接入支付宝产品时，请把这些信息通过商户项目自己的服务端传递。

4. 代码示例运行逻辑

步骤1： 调用 order.m 里的函数 description 将商品信息拼接成字符串作为待签名字符串，如：

```
"partner=\"2088101568353491\"&seller_id=\"2088101568353491\"&out_trade_no=\"YR2VGG3G1I31XDZ\"&subject=\"1\"&body=\"  我  是  测  试  数  据\"&total_fee=\"0.02\"&notify_url=\"http://www.xxx.com\"&service=\"mobile.securitypay.pay\"&payment_type=\"1\"&_input_charset=\"utf-8\"&it_b_pay=\"30m\"&show_url=\"m.alipay.com\""
```

步骤2： 使用类 CreateRSADataSigner，调用 signString 签名函数做签名，如：

```
"GsSZgPloF1vn52XAItRAldwQAbzIgkDyByCxMfTZG%2FMapRoyrNIJo4U1LUGjHp6gdBZ7U8jA1kljLPqkeGv8MZigd3kH25V0UK3Jc3C94Ngxm5S%2Fz5QsNr6wnqNY9sx%2Bw6DqNdEQnnks7PKvvU0zgsynip50lAhJmflmfHvp%2Bgk%3D"
```

步骤3： 把签名结果赋值给参数 sign，并把 sign 加入之前的待签名数组中，此时得到的便是要请求给支付宝的全部数据。

```
"partner=\"2088101568353491\"&seller_id=\"2088101568353491\"&out_trade_no=\"YR2VGG3G1I31XDZ\"&subject=\"1\"&body=\"  我  是  测  试  数  据\"&total_fee=\"0.02\"&notify_url=\"http://www.xxx.com\"&service=\"mobile.securitypay.pay\"&payment_type=\"1\"&_input_charset=\"utf-8\"&it_b_pay=\"30m\"&show_url=\"m.alipay.com\"&sign=\"GsSZgPloF1vn52XAItRAldwQAbzIgkDyByCxMfTZG%2FMapRoyrNIJo4U1LUGjHp6gdBZ7U8jA1kljLPqkeGv8MZigd3kH25V0UK3Jc3C94Ngxm5S%2Fz5QsNr6wnqNY9sx%2Bw6DqNdEQnnks7PKvvU0zgsynip50lAhJmflmfHvp%2Bgk%3D\"&sign_type=\"RSA\""
```

步骤4： 调用(AlipaySDK *)defaultService 类下面的支付接口函数，唤起支付宝支付页面。

```
(void)payOrder:(NSString *)orderStr
     fromScheme:(NSString *)schemeStr
       callback:(CompletionBlock)completionBlock
```

appScheme 为 app 在 info.plist 注册的 scheme。

图5-5 支付宝支付页面

后面的动作全由买家在支付宝收银台中操作完成。如果设备中有支付宝客户端，会优先调用支付宝客户端进行支付，支付完成后会重新唤起商户 app。

步骤5： 当这笔交易被买家支付成功后支付宝收银台上显示该笔交易成功，并提示用户“返回”。此时在 APAppDelegate.m 的 -(BOOL)application:(UIApplication)application openURL:(NSURL)url sourceApplication:(NSString *)sourceApplication annotation:(id)annotation 中调用获取返回数据的代码：

```
[[AlipaySDK defaultService]
processOrderWithPaymentResult:url
standbyCallback:^(NSDictionary *resultDic) {
NSLog(@"result = %@",resultDic);//返回的支付结果
//【由于在跳转支付宝客户端支付的过程中，商户 app 在后台很可能被系统 kill 了，所以 pay 接口的 callback 就会失效，请商户对 standbyCallback 返回的回调结果进行处理，就是在这个方法里面处理跟 callback 一样的逻辑】
}];
```

拿到返回数据：

- 点取消后返回

```
"alisdkdemo://safepay/?%7B%22memo%22:%7B%22result%22:%22%22,%22memo%22:%2
```

```
2%E7%94%A8%E6%88%B7%E4%B8%AD%E9%80%94%E5%8F%96%E6%B6%88%22,%22ResultStatus%22:%226001%22%7D,%22requestType%22:%22safepay%22%7D"
```

对其做 URLDecode

```
"alisdkdemo://safepay/?{"memo":{"result":"","memo":"用户中途取消","ResultStatus":"6001"},"requestType":"safepay"}"
```

- 点确认后返回

```
"alisdkdemo://safepay/?%7B%22memo%22:%7B%22result%22:%22partner=%5C%2220888101568353491%5C%22&seller_id=%5C%2220888101568353491%5C%22&out_trade_no=%5C%22QU6ZOD85K4HVQFN%5C%22&subject=%5C%221%5C%22&body=%5C%22%E6%88%91%E6%98%AF%E6%B5%8B%E8%AF%95%E6%95%B0%E6%8D%AE%5C%22&total_fee=%5C%220.02%5C%22&notify_url=%5C%22http:%5C/%5C/www.xxx.com%5C%22&service=%5C%22mobile.securitypay.pay%5C%22&payment_type=%5C%221%5C%22&_input_charset=%5C%22utf-8%5C%22&it_b_pay=%5C%2230m%5C%22&show_url=%5C%22m.alipay.com%5C%22&success=%5C%22true%5C%22&sign_type=%5C%22RSA%5C%22&sign=%5C%22pg16DPA%5C/cIRg1iUFCl8lYZG54de+kfw+vCj32hGWye97isZ1A4bW6RNaDXHhZXVaI5Vk2YDxhNUl85EHRd+EL7%5C/+ogQTnsaEHl+D13PuZExIXRKGBnkYqaNV6kH6hDygnf5IOtoojHWLQyem7oRBVzB0vlF%5C/+YGFpzFHZyTVpM8=%5C%22%22,%22memo%22:%22%22,%22ResultStatus%22:%229000%22%7D,%22requestType%22:%22safepay%22%7D"
```

对其做 URLDecode

```
"alisdkdemo://safepay/?{"memo":{"result":"partner=\"2088101568353491\"&seller_id=\"2088101568353491\"&out_trade_no=\"QU6ZOD85K4HVQFN\"&subject=\"1\"&body=\"我是测试数据\"&total_fee=\"0.02\"&notify_url=\"http:\/\/www.xxx.com\"&service=\"mobile.securitypay.pay\"&payment_type=\"1\"&_input_charset=\"utf-8\"&it_b_pay=\"30m\"&show_url=\"m.alipay.com\"&success=\"true\"&sign_type=\"RSA\"&sign=\"pg16DPA\/cIRg1iUFCl8lYZG54de+kfw+vCj32hGWye97isZ1A4bW6RNaDXHhZXVaI5Vk2YDxhNUl85EHRd+EL7\/+ogQTnsaEHl+D13PuZExIXRKGBnkYqaNV6kH6hDygnf5IOtoojHWLQyem7oRBVzB0vlF\/+YGFpzFHZyTVpM8=\"","memo":"","ResultStatus":"9000"},"requestType":"safepay"}"
```

之后，对这些数据做处理。

⚠ 注意：

- 由于在跳转支付宝客户端支付的过程中，商户 app 在后台很可能被系统 kill 了，所以 pay 接口的 callback 就会失效，请商户对 standbyCallback 返回的回调结果进行处理；
- 同步返回数据时，建议通过服务端的验签功能代码做验签处理，之后再对返回的数据做业务逻辑处理；
- 须以服务器异步通知的结果数据为准，并对其做业务逻辑处理；
- SDK 付款有两种模式：如果外部存在支付宝钱包，则直接跳转到支付宝钱包付款；不存在的场景下，在 SDK 内部进行 H5 支付。测试同学需要关注这两类测试场景。

5.2.2 Android

1. 开发包

解压接口压缩文件（文件名是 WS_MOBILE_PAY_SDK_BASE.zip），找到安卓的压缩文件（文件名是支付宝钱包支付开发包标准版(Android).zip）。标准开发包以 jar 包方式提供给商户应用工程集成，打开 alipay-sdk-common 文件夹获取 alipaySDK-20150602.jar，后 8 位数字标识发布日期，商户可根据日期时间判断 SDK 版本的新旧。

2. 导入开发资源

(1) 将 alipaySDK-20150602.jar 包放入商户应用工程的 libs 目录下，如下图。

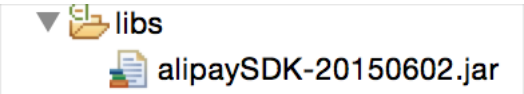

图5-6 libs 目录结构

(2) 进入商户应用工程的 Java Build Path，将 libs 目录下的 alipaySDK-20150602.jar 导入，如下图。

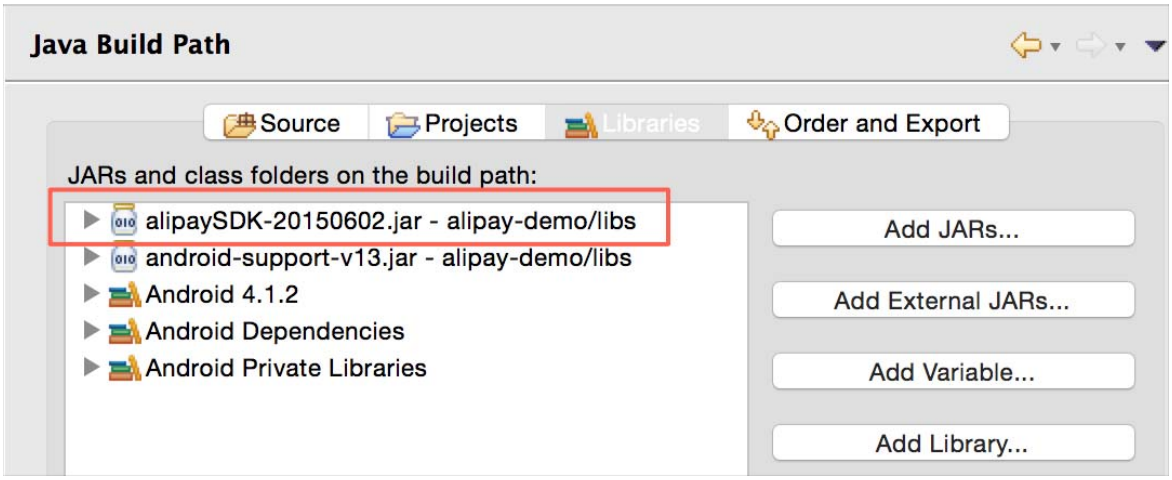

图5-7 导入 jar

(3) 选中 Order and Export，勾选 alipaySDK-20150602.jar，如下图。

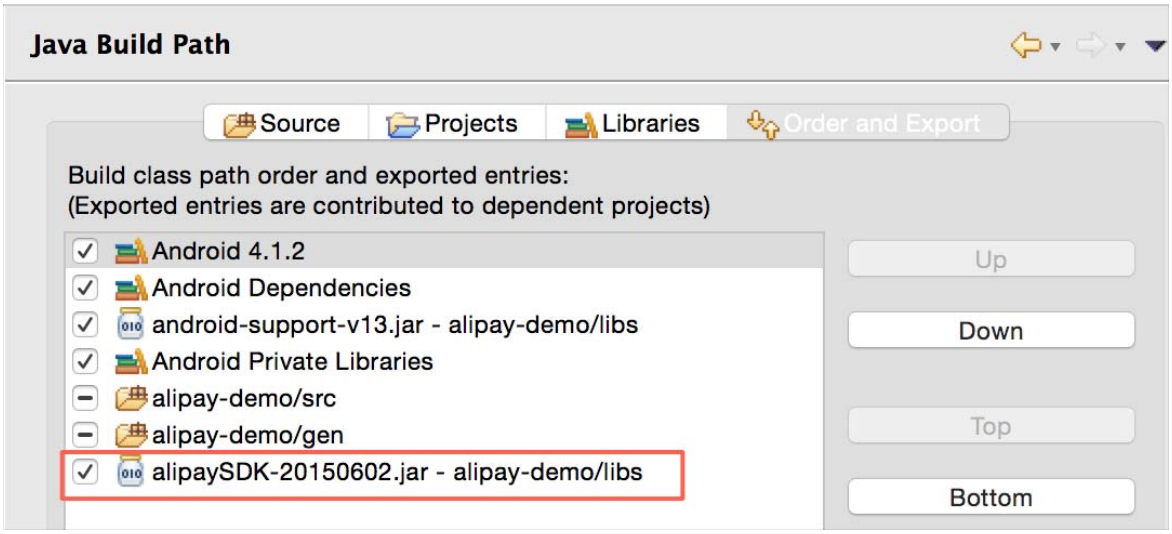

图5-8 勾选 3 个 jar 包

3. 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.sdk.app.H5PayActivity"
          android:configChanges="orientation|keyboardHidden|navigation"
          android:exported="false"
          android:screenOrientation="behind" >
</activity>
<activity
          android:name="com.alipay.sdk.auth.AuthActivity"
          android:configChanges="orientation|keyboardHidden|navigation"
          android:exported="false"
```

```
          android:screenOrientation="behind" >
 </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" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission  android:name="android.permission.WRITE_EXTERNAL_STORAGE"
/>
```

4. 添加混淆规则

在商户应用工程的 proguard-project.txt 里添加以下相关规则：

```
-libraryjars libs/alipaySDK-20150602.jar

-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.app.AuthTask{ public *;}
```

至此，开发包开发资源导入完成。

5. 订单数据生成

在调用开发包支付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“&”分隔，所有参数不可缺。

6. 支付接口调用

需要在新线程中调用支付接口。（可参考 alipay_demo 实现）

获取 PayTask 支付对象调用支付，代码示例：

```
final String orderInfo = info;   // 订单信息

     Runnable payRunnable = new Runnable() {

         @Override
         public void run() {
             PayTask alipay = new PayTask(DemoActivity.this);
             String result = alipay.pay(orderInfo);

             Message msg = new Message();
             msg.what = SDK_PAY_FLAG;
             msg.obj = result;
```

```
                mHandler.sendMessage(msg);
            }
        };
        // 必须异步调用
        Thread payThread = new Thread(payRunnable);
        payThread.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

商户应用客户端通过当前调用支付的 Activity 的 Handler 对象，通过它的回调函数获取支付结果。（可参考 alipay_demo 实现）

代码示例：

```
private Handler mHandler = new Handler() {
     public void handleMessage(Message msg) {
         Result result = new Result((String) msg.obj);
         Toast.makeText(DemoActivity.this, result.getResult(),
                     Toast.LENGTH_LONG).show();
     };
 };
```

- 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。

8. 查询有效账户接口调用

调用 PayTask 对象的 checkAccountIfExist()方法查询。（可参考 alipay_demo 实现）

代码示例：

```
Runnable checkRunnable = new Runnable() {

         @Override
         public void run() {
             PayTask payTask = new PayTask(DemoActivity.this);
             boolean isExist = payTask.checkAccountIfExist();

             Message msg = new Message();
             msg.what = SDK_CHECK_FLAG;
             msg.obj = isExist;
```

```
                mHandler.sendMessage(msg);
            }
        };

        Thread checkThread = new Thread(checkRunnable);
        checkThread.start();
```

9. 获取当前开发包版本号

调用 PayTask 对象的 getVersion()方法查询。

代码示例：

```
PayTask payTask = new PayTask(activity);
String version = payTask.getVersion();
```

⚠ 注意：

SDK 付款有两种模式：如果外部存在支付宝钱包，则直接跳转到支付宝钱包付款；没有支付宝钱包的场景下，将触发在 SDK 内部进行 H5 支付。商户在测试集成支付是否正常的时候，建议测试（存在、没有）支付宝钱包的场景。对于测试过程中出现的异常，请联系支付宝技术支持进行处理。

6 测试流程规则

表6-1 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

步骤	调试内容	备注
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。

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支付宝网站，点击“签约管理”栏目下的“签约订单”。

图7-1 我的商家服务

步骤2： 在跳转后的页面中点击“查看PID|KEY”，在新打开的页面中（https://b.alipay.com/order/pidAndKey.htm ），可查看到签约支付宝账号、合作者身份ID（PID）。

图7-2 查询 PID

步骤3： 输入支付密码，查询 key、支付宝公钥。

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。

步骤4： 上传 RSA 公钥

在“合作伙伴密钥管理”下，点击“RSA 加密”后的“添加密钥”，把自己的公钥复制进去，如下图所示。

图7-4 上传 RSA 公钥

⚠ 注意：

上传的公钥是一行格式，不允许有注释、空格、换行等！

转换前 pem 文件格式：

```
-----BEGIN PUBLIC KEY-----
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDQWiDVZ7XYxa4CQsZoB3n7bfxL
DkeGKjyQPt2FUtm4TWX9OYrd523iw6UUqnQ+Evfw88JgRnhyXadp+vnPKP7unorm
YQAfsM/CxzrfMoVdtwSiGtIJB4pfyRXjA+KL8nIa2hdQy5nLfgPVGZN4WidfUY/Q
pkddCVXnZ4bAUaQjXQIDAQAB
-----END PUBLIC KEY-----
```

转换后的字符串：

```
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDQWiDVZ7XYxa4CQsZoB3n7bfxLDkeGKjyQP
t2FUtm4TWX9OYrd523iw6UUqnQ+Evfw88JgRnhyXadp+vnPKP7unormYQAfsM/CxzrfMoVdtw
SiGtIJB4pfyRXjA+KL8nIa2hdQy5nLfgPVGZN4WidfUY/QpkddCVXnZ4bAUaQjXQIDAQAB
```

步骤5： 点击“确认上传”，提示：上传成功，说明已经成功上传。

图7-5 上传成功提示

> 说明：
> 如果需要修改公钥，只需要把新的公钥复制进去，点击“修改”即可！

步骤6： 查看支付宝公钥：点击“RSA 加密”后面的“查看支付宝公钥”即可查看到对应的支付宝公钥。

图7-6 查看支付宝公钥

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：

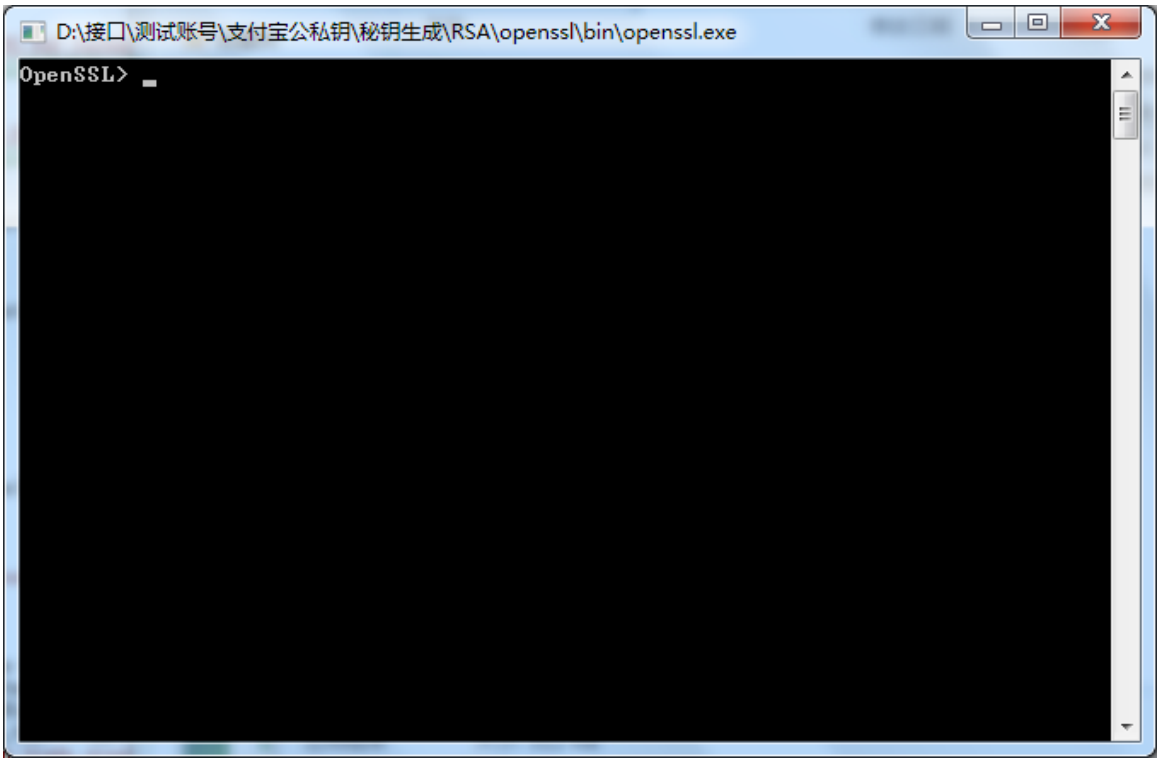

图7-7 执行 openssl.exe 文件

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：

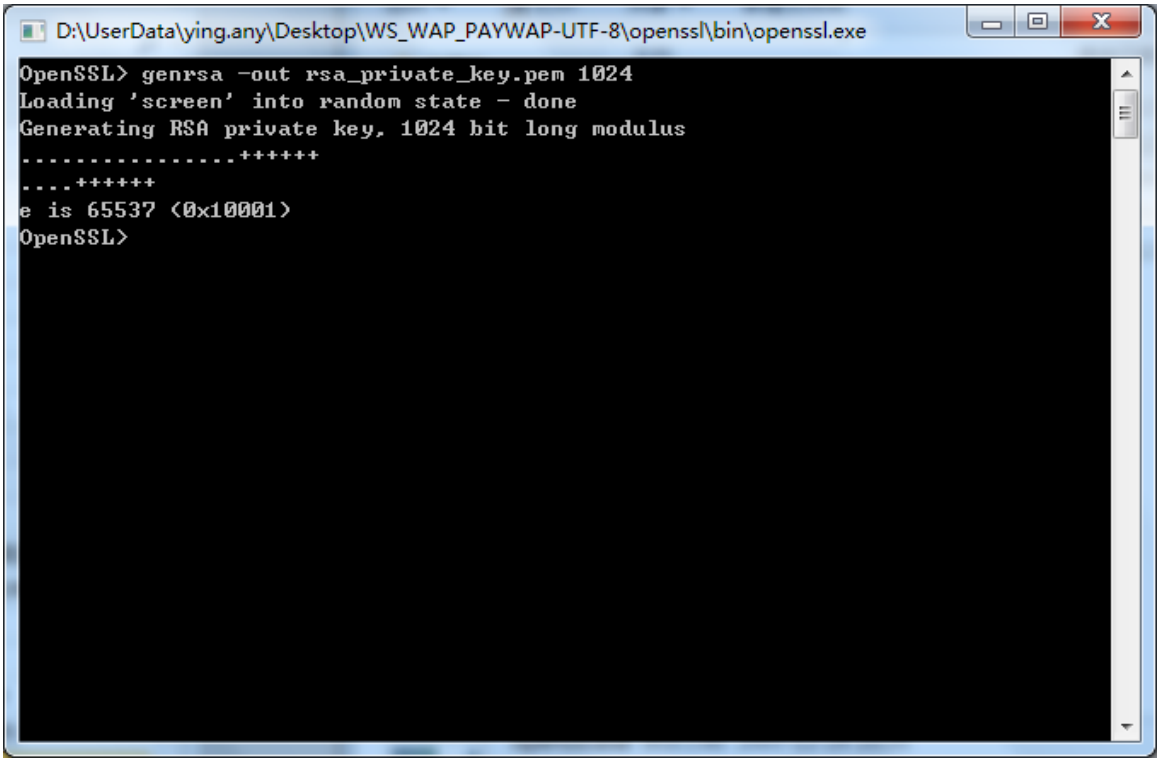

图7-8 生成 RSA 私钥

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：

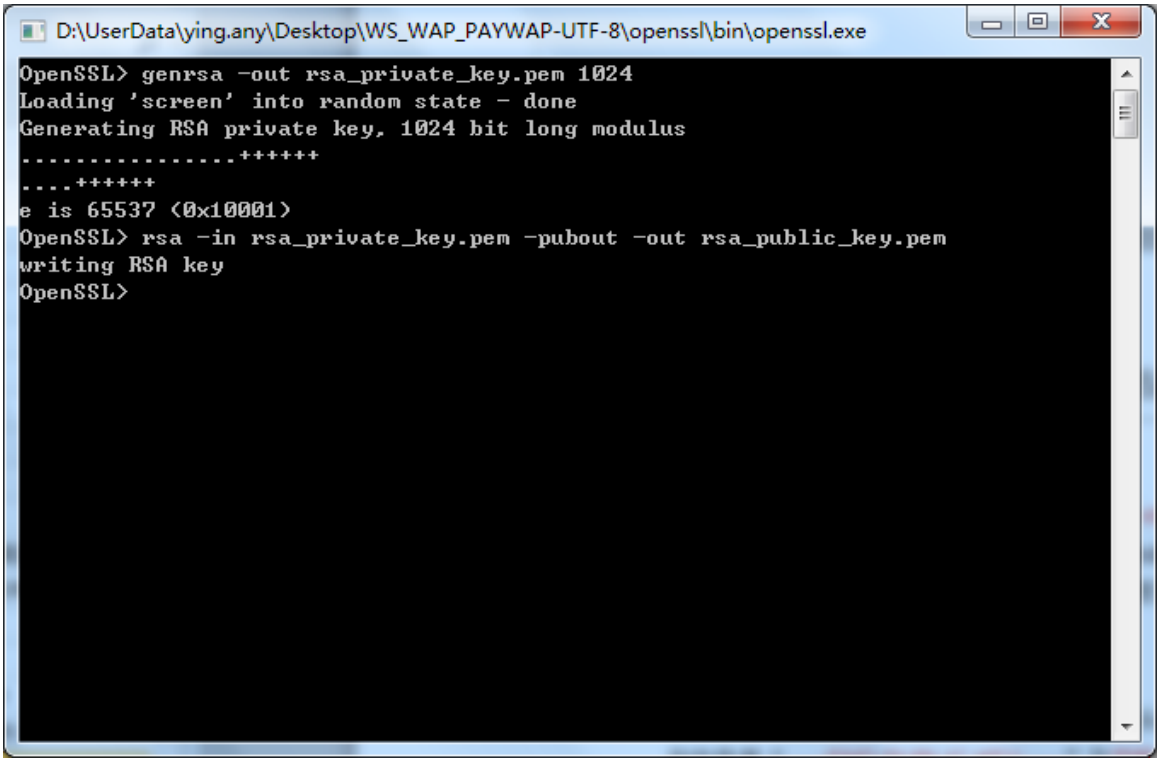

图7-9 生成 RSA 公钥

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：

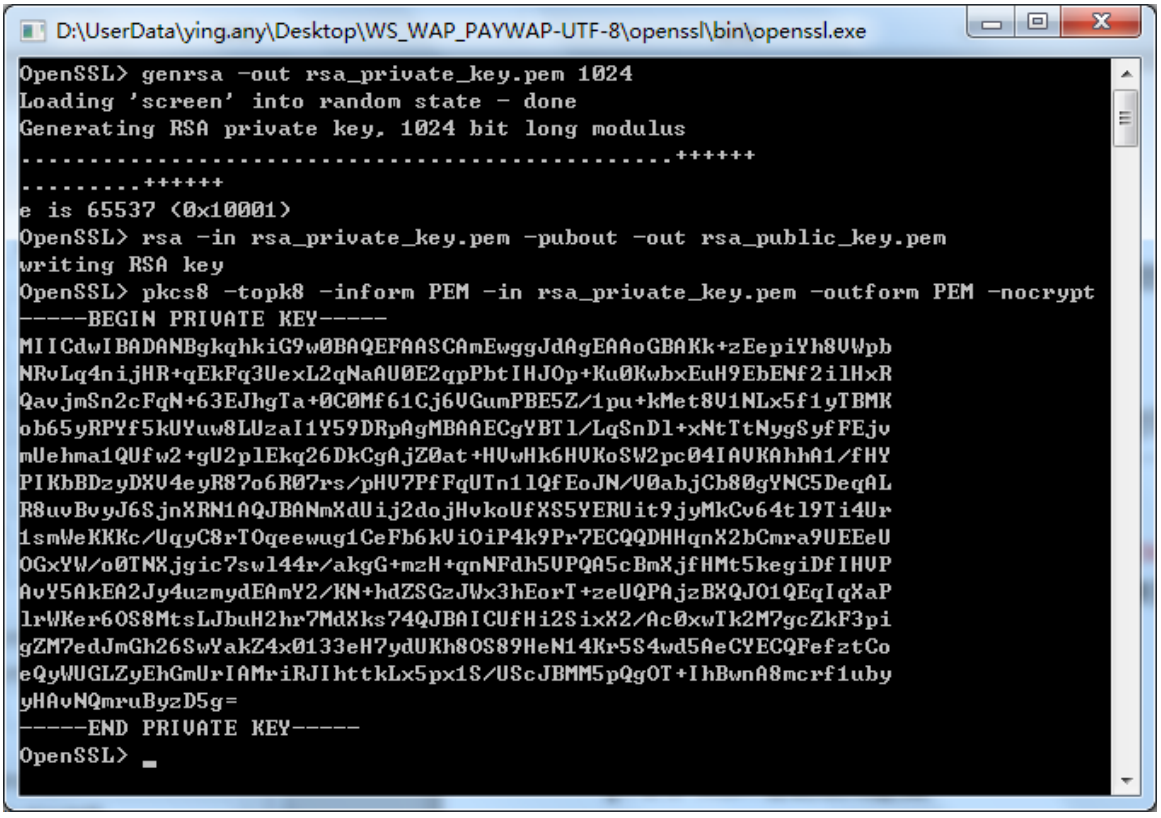

图7-10 生成 PKCS8 编码的私钥

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：

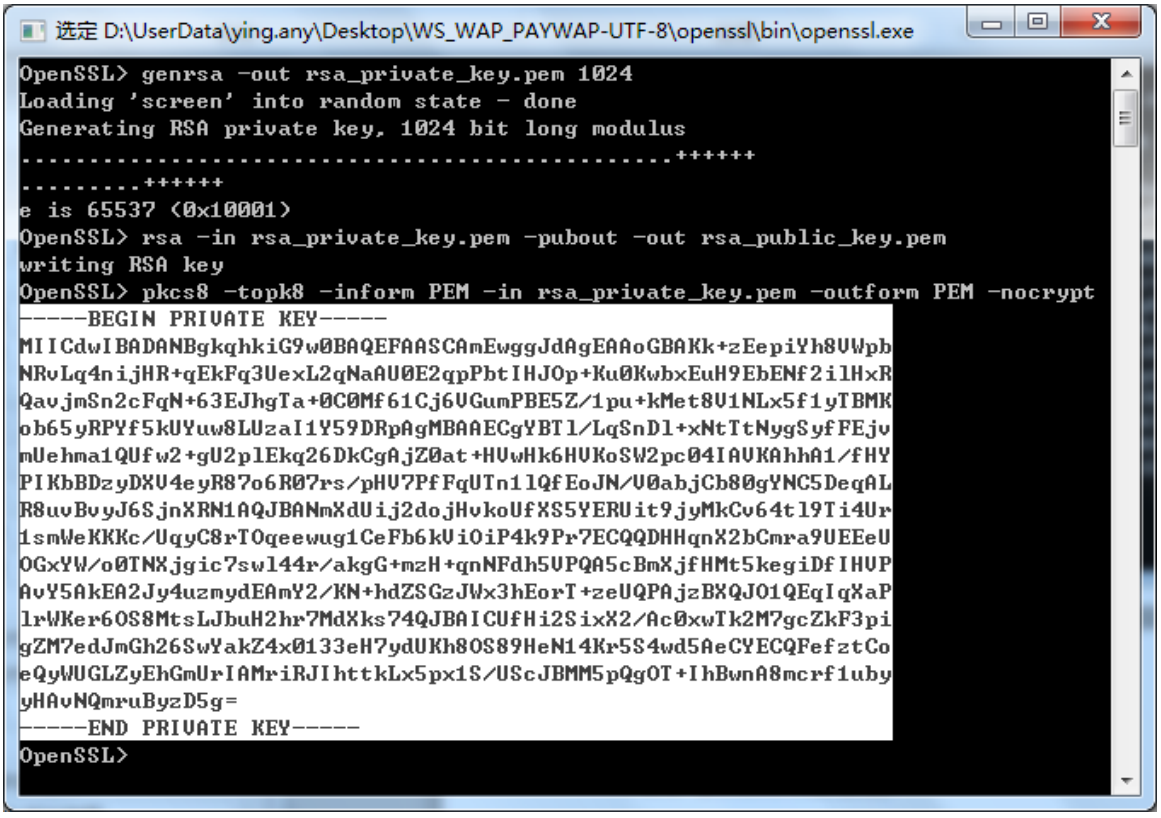

图7-11 选中要复制的文字

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
  - 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：

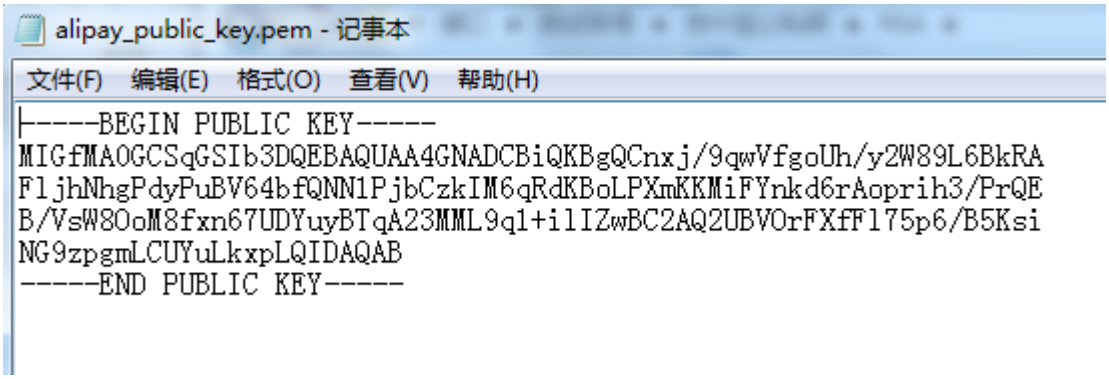

图7-12 支付宝公钥示意图

(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。