安全支付服务 iOS

应用开发指南

文件版本：3.0.1.1

2012-05-09

前言

1. 面向读者

本文档主要面向需要接入支付宝安全支付的商户的开发人员。

2. 读者所需技能

读者需有 iOS 程序开发背景，掌握 objective-C 和 Xcode 程序开发。

3. 开发环境要求

OS: Mac OS X 10.6 或以上版本

SDK: iOS 4.0 或以上版本

备注：基于 iOS 开发需要 apple 开发者账号和证书，请先去 apple 官方了解相关事项。

目录

第一章安全支付服务简介

1.1 安全支付服务介绍

iOS 安全支付服务主要用来向第三方应用程序提供便捷、安全以及可靠的支付服务。安全支付服务内嵌在 iOS 支付宝客户端中，保证了在不同场合下商户接入场景。本文主要描述安全支付服务应用开发接口的使用方法，供合作伙伴的开发者接入使用。

1.2 安全支付服务业务流程

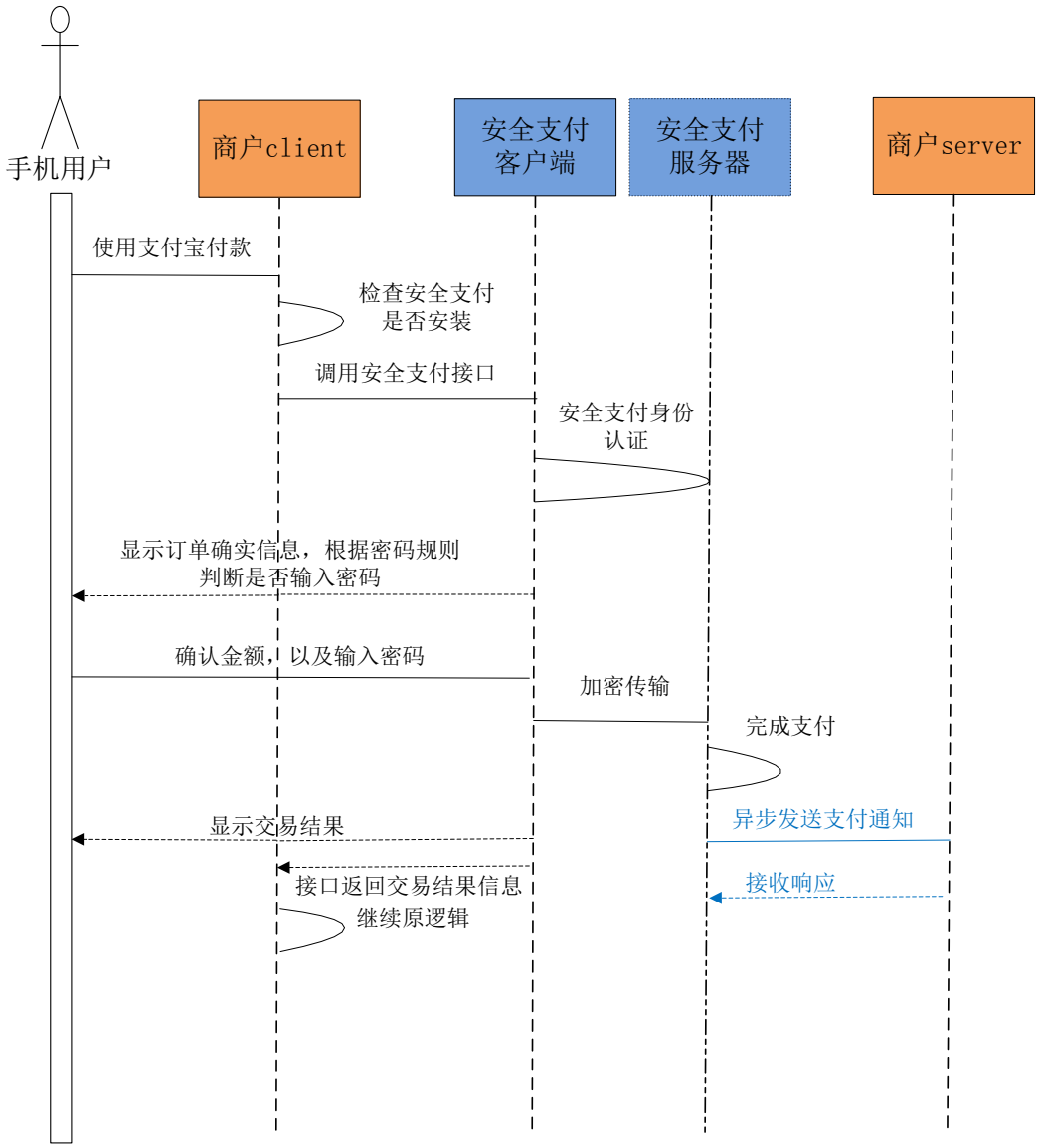


图 1-1 安全支付业务流程图

1.3 调用安全支付数据流程图

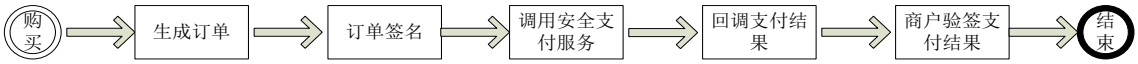


图 1-2 安全支付数据流程图

第二章安全支付接入流程

2.1 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。

2.1.1 商户签约

首先，商户需要在 https://ms.alipay.com 进行注册，并签约安全支付服务。签约成功后可获取支付宝分配的合作商户 ID(PartnerID)，账户 ID(SellerID)，如图：

图 2-1 商户 ID 获取示意图

签约过程中需要任何帮助请致电：0571-88158090（支付宝商户服务专线）

2.1.2 密钥配置

签约成功后，商户可登陆 https://ms.alipay.com 获取商户账号对应的支付宝公钥，具体获取步骤请见 3.2 RSA 密钥详解

接着，商户生成商户公钥和商户私钥（具体生成步骤请见 3.2 RSA 密钥详解），并登录 https://ms.alipay.com，上传商户公钥（具体上传步骤请见 3.2 RSA 密钥详解）。

至此，接入前期准备工作完成，下一节将使用 demo 测试准备工作是否正确。

2.2Demo

为了便于商户的接入，我们提供了安全支付 demo。通过本 demo，商户可测试 2.1 节的前期准备工作是否正确完成，同时还可参考 demo 的代码完成接入。

步骤 1:

解压下载的安全支付开发资料压缩包 WS_SECURE_PAY(20110907)（注：凡文件名后有日期，表示编写该文档时的最新版本号，今后有更新，该文档不会及时更新，后面不再说明），进入目录“*WS_SECURE_PAY(20110907)\iOS(20110907)\AlipayDemo*”，双击打开 *AlixPayDemo.xcodeproj* 该工程文件。

项目结构如图：

图 2- 2 Demo 项目结构图

步骤 2:

编译一下，保证 demo 编译成功。

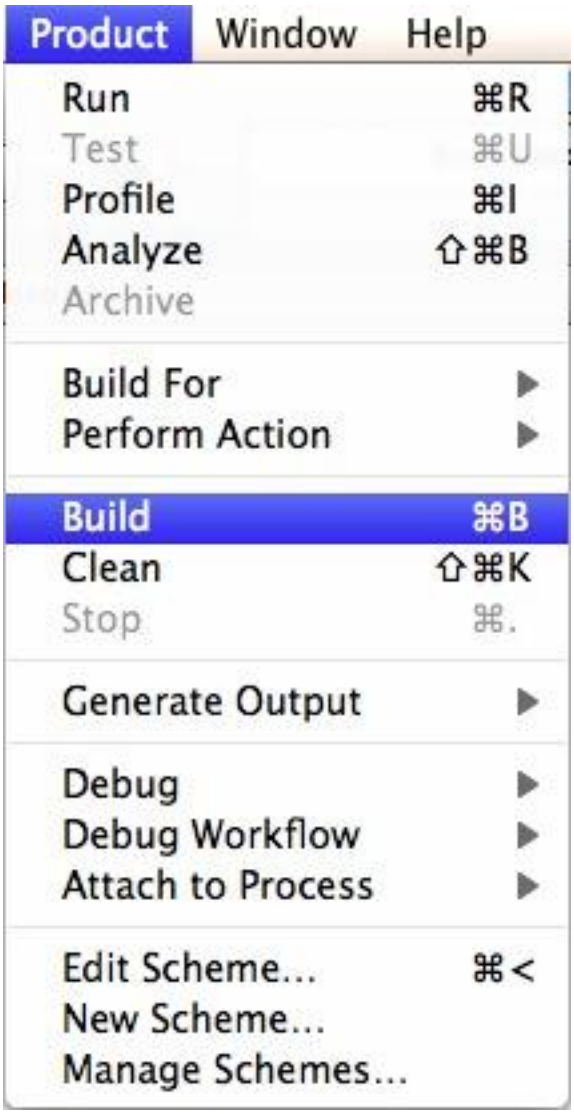


图 2- 3

步骤 3:

展开 Resources 目录，打开 AlixPayDemo-Info.plist 文件并添加商户账号信息，如下图：

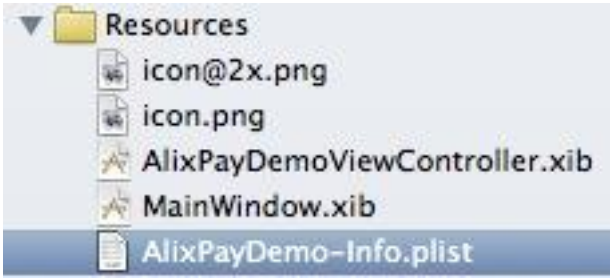


图 2- 4

Key	Type	Value
Localization native development region	String	English
Bundle display name	String	${PRODUCT_NAME}
Executable file	String	${EXECUTABLE_NAME}
Icon file	String	
Bundle identifier	String	com.yourcompany.${PRODUCT_NAME:rfc1034identifier}
InfoDictionary version	String	6.0
Bundle name	String	${PRODUCT_NAME}
Bundle OS Type code	String	APPL
Bundle creator OS Type code	String	????
Bundle version	String	1.0
Application requires iPhone environmer	Boolean	YES
Main nib file base name	String	MainWindow
Partner	String	2088102000947391
Seller	String	2088102000947391
RSA private key	String	MIICdwIBADANBgkqhkiG9w0BAQEFAASCAmEwggJdAgEAA
RSA public key	String	MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCFap2x
▶ URL types	Array	(1 item)

图 2- 4 AlixPayDemo-Info.plist 配置信息

参数详解：

Partner（合作商户 ID）

Seller（账户 ID）

RSA public key（支付宝公钥）

RSA private key（商户私钥）

备注：以上信息仅做参考，商户在运行之前请把以上 4 个参数改为自己的信息。

步骤 4：（该步骤意在提示开发者不能用模拟器测试安全支付服务，如果需要正确运行请看步骤五）

Run 项目，本 demo 由于需要调用支付宝客户端所以无法做到模拟器上运行，需要真机测试，且保证设备安装有支付宝客户端，界面如下：

图 2- 5

点击选中其中一项，会启动安全支付服务，如果是在模拟器中测试，由于模拟器无法安装支付宝客户端，并且跳转 appStore 失败，所以出现以下情况：

图 2- 6

图 2- 7

步骤 5：（真机部署）

连接 iPhone 手机后，Xcode 修改启动选项

图 2- 8

按照真机调试的要求修改 AlixPayDemo-Info.plist 中 Bundle identifier 信息和选择自己的开发者证书对应的 profile，关于真机调试详细说明请参见 developer.apple.com

图 2- 9

Run 部署到真机，商户产品列表请看“图 2- 5”，点选一条商品信息，正常进入资金渠道页面(如果支付宝客户端绑定支付宝账号，进入收银台页面)，则说明前期准备工作完成。下图为资金渠道页面

图 2- 8

2.3 安全支付集成

本章指导在商户项目中集成安全支付，关键代码以 Demo 为例，可参见代码注释。

说明：

1) 为尽量避免接入安全支付服务代码与外部商户自身代码产生冲突，此次更新版本提供 libalixpay.a 库文件函数实现源码，外部商户可以选择源码代替 libalixpay.a 库文件，自行优化。库文件接入良好的外部商户仍可以使用 libalixpay.a 文件，不过请避免 Alixpay.m，AlixpayOrder.m，AlixpayResult.m 三个源文件与 libalixpay.a 的同时使用。
2) Libalixpay.a 库文件工程目录

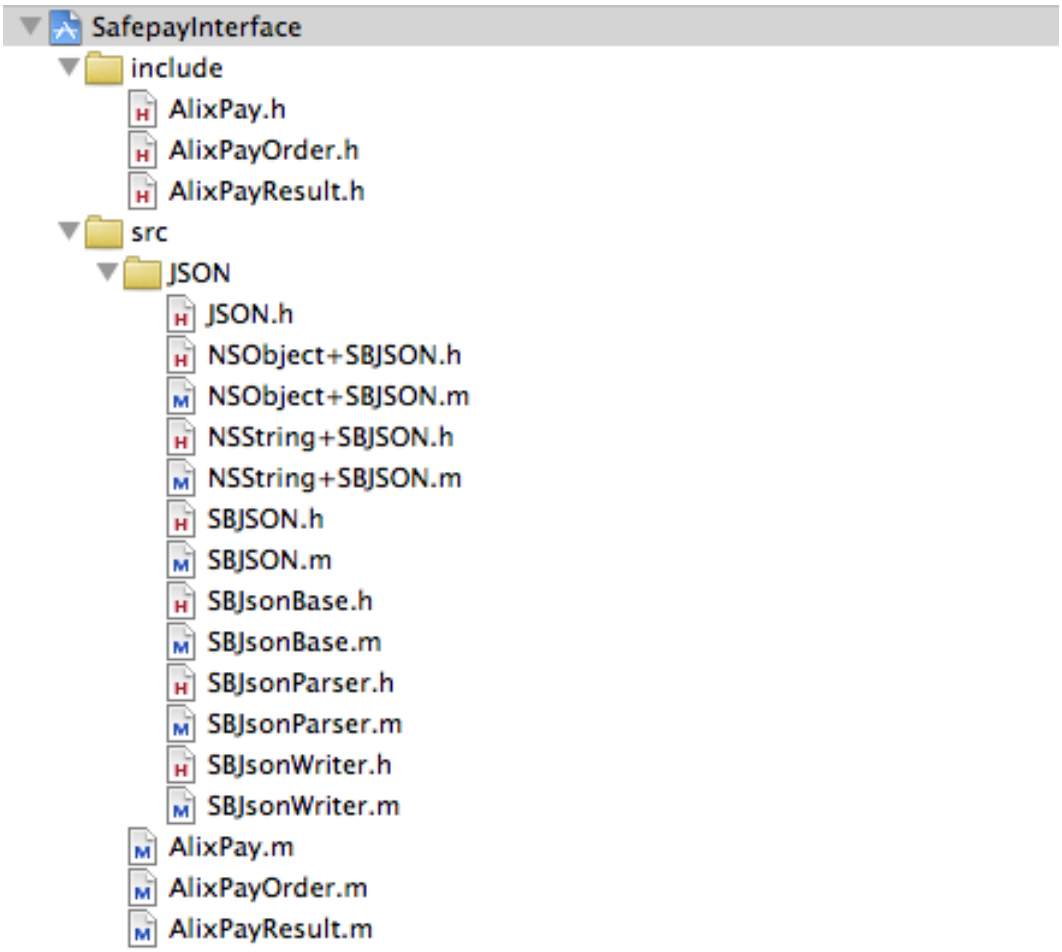


图 2-9

因为安全支付服务接收的数据需要 Json 结构化，请使用源码接入安全支付的外部商户自行添加 SBJson 类库或自行编写相应 Json 转换函数。

3）本次更新将上一版本的 libalixpay.a 和 libalixpay-simulator.a 两个库文件整合成 libalixpay.a 一个库文件，可供真机和模拟器共同使用。使用上一版本库文件接入安全支付服务的外部商户可以酌情更替。

4）选择使用源码接入安全支付的外部商户，默认建议使用 include 中 Alixpay.h，AlixpayOrder.h，AlixpayResult.h 和 scr 中的 Alixpay.m，AlixpayOrder.m，AlixpayResult.m 的接口函数来完成安全支付服务调用。demo 中的 RSA 签名，验签，base64 编码，urlEncode 函数接口也仅供参考，外部商户可以自行进行优化，只要保证产生的商品信息字符串签名，编码正确，且拼装的最终字符串满足文档规范格式。

步骤 1：libs 库文件接入（步骤 1 和步骤 2 请选一种）

复制 *include* 目录下的 *AlixPay.h*，*AlixPayOrder.h*，*AlixPayResult.h* 和 libs 目录下的 libalixpay.a 四个文件到商户的工程下。

步骤 2：libs 源码接入

使用源码接入安全支付服务，默认添加 *include* 中 *Alixpay.h*，*AlixpayOrder.h*，*AlixpayResult.h* 和 *scr* 中的 *Alixpay.m*，*AlixpayOrder.m*，*AlixpayResult.m* 六个文件到工程中。因为传递给安全支付的数据需要 *json* 结构化，所以建议使用 *SBJson* 类库或自行实现相应函

数（*Alixpay* 和 *AlixpayResultx* 相应接口使用 *SBJson* 类库，使用时，请添加 *JSON* 文件夹目录下文件）。

步骤 3：添加 openssl 函数库

Demo 中签名函数作为参考（商户可以使用也可以自行编写签名函数），如需使用，请另添加 Utilities/openssl 文件夹及文件夹目录下文件和 DataSigner.h，DataSigner.m，DataVerifier.h，DataVerifier.m 四个文件到工程中，工程 Targets 的 Build 面板中 Header Search Paths 请添加 openssl 文件夹所在的路径，本 demo 中放在 Utilities 文件夹下。

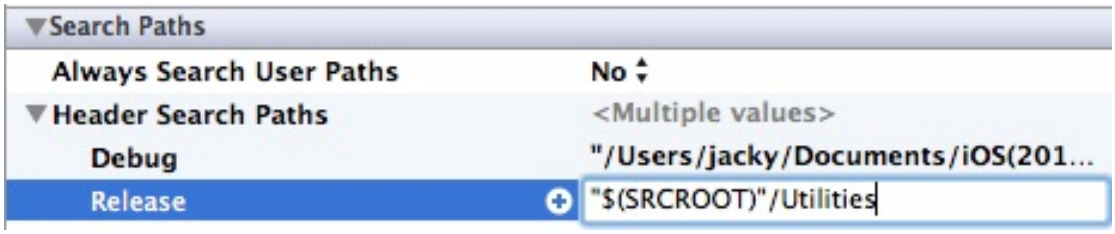


图 2- 10 头文件搜索路径

debug 和 release 都需要如图修改，这个路径就是声明 openssll 所在的目录。

添加 libcrypto.a 和 libssl.a 两个库文件(这两个库文件包含 openssl 相关源码)到工程中，如下图，这样就可以使用 Demo 中 DataSigner，DataVerifier 提供的签名，验签函数。

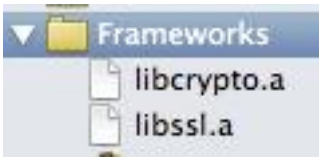


图 2- 11

步骤 4：订单数据生成

在调用安全支付时，需要提交订单信息 *orderSpec*，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。示例如下：（红色参数表示该参数值需与示例一致，不可自定义；蓝色参数表示值可自定义。具体参数说明请见订单信息描述）

```
partner="2088002007260245"&seller="2088002007260245"&out_trade_no=
"500000000006548"&subject="商品名称"&body="这是商品描述
"&total_fee="30"&notify_url="http://notify.java.jpxx.org/index.jsp
"&sign="kU2Fa3x6V985g8ayTozI1eJ5fHtm8%2FJGeJQf9in%2BcVmRJjHaExbirn
GGKJ%2F7B63drqc4Kjlk%2FSg6vtSIkOtdvVBrRDpYaKxXVqkJTzRYgUwrrpMudbIj
9aMS2O3dHG0GPyL4Zb6jKDYXHabGG0aBJY3QA7JuTJ23t6SqV%2B5f1xg%3D"&sign
_type="RSA"
```

商户可以使用或参照 AlixpayOder 的*-(NSString*)description* 函数完成订单数据信息字符串拼接。

商户可以使用或参照 DataSigner 的*-(NSString*)signstring:(NSString*)*string 函数完成订单信息签名。

其中 *sign* 值的生成，请见对商品信息进行 *RSA* 签名。需要特别注意的是：对数据签名

后得到的 *sign* 值必须进行 *URLEncode*，之后才可作为参数。

步骤 6：调用安全支付服务

默认在需要调用安全支付服务的地方调用 *Alixpay *alixpay = [Alixpay shared]*；并将签完名，*base64* 编码且 *UrlEncode* 的商品信息字符串（Demo 中使用函数 -(NSString*)signstring:(NSString*)string 完成对订单信息的上述三步处理），然后通过 *-(int)pay: (NSString *)orderStringapplicationScheme: (NSString*)scheme* 函数来唤起安全支付服务。该函数方法通过指定 Url 然后打开的方式调用安全支付服务，详细信息参加 Alixpay.m。

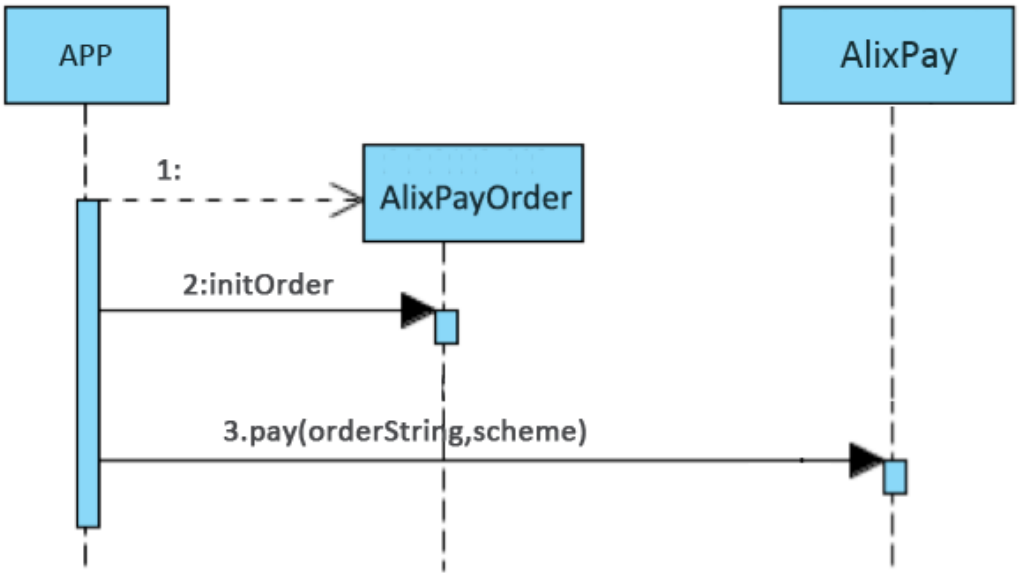


图 2- 12

步骤 7：支付结果获取和处理

调用安全支付后，将通过两种途径获得支付结果：

1、*AliXpay* 方法的返回：

在程序的 *ApplicationDelegate* 文件中需要添加支付结果回调处理函数，从安全支付页面返回外部商户程序时 *iOS4.0* 以上系统版本会响应 *-(BOOL)application: (UIApplication *)application handleOpenURL: (NSURL *)url* 函数，而 *iOS4.0* 以下版本实则是重新启动外部商户程序，进入 *-(BOOL)application: (UIApplication *)application didFinishLaunchingWithOptions: (NSictionary *)launchOptions* 函数，外部商户根据兼容版本情况在相关位置添加回调处理函数 *-(BOOL)parseURL: (NSURL *)url application: (UIApplication *)application* 函数，处理安全支付回调信息。详细参见 Demo。

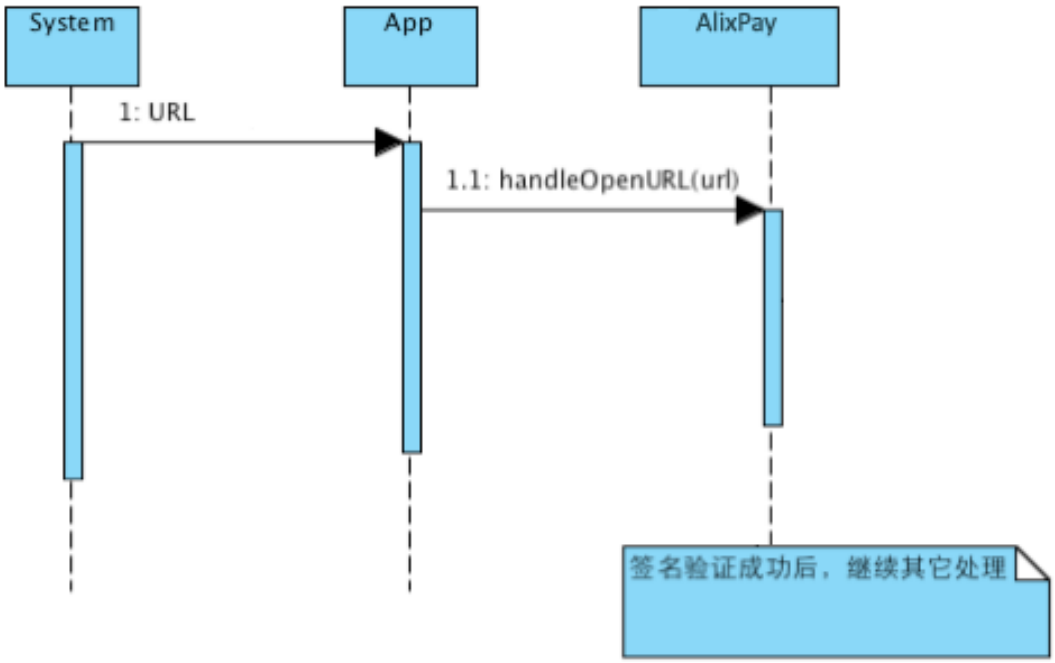


图 2- 13

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安全支付，即调用安全支付服务时拼装商品信息中的 *notify_url*。

支付宝服务器在支付完成后，会用 POST 方法调用 *notufy_url*，以 xml 为数据格式传输支付结果，详见

第三章 RSA 详解

*以下内容加*号为重点*

3.1 RSA 和 OpenSSL 介绍

3.1.1 什么是 RSA

RSA 是一种非对称的签名算法，即签名密钥（私钥）与验签密钥（公钥）是不一样的，私钥用于签名，公钥用于验签。

在与支付宝交易中，会有 2 对公私钥，即商户公私钥，支付宝公私钥。

3.1.2 为什么要用 RSA

使用这种算法可以起到防止数据被篡改的功能，保证支付订单和支付结果不可抵赖(商户私钥只有商户知道)。

3.1.3 什么是 OpenSSL

一句话概括：OpenSSL 是基于众多的密码算法、公钥基础设施标准以及 SSL 协议安全开发包。

3.1.4 为什么要用 OpenSSL

通过 OpenSSL 生成的签名和内置的算法可以做到跨平台，这样在不同的开发语言中均可以签名和验签。

3.2 RSA 密钥详解*

3.2.1 找到生成 RSA 密钥工具

1. 下载开发指南和集成资料，如下图，您能看到此文档说明指南和集成包已经下载了。

图 3- 1 文档下载

2. 解压下载的压缩包(WS_SECURE_PAY)，找到并解压 openssl-0.9.8k_WIN32(RSA 密钥生成工具).zip 工具包

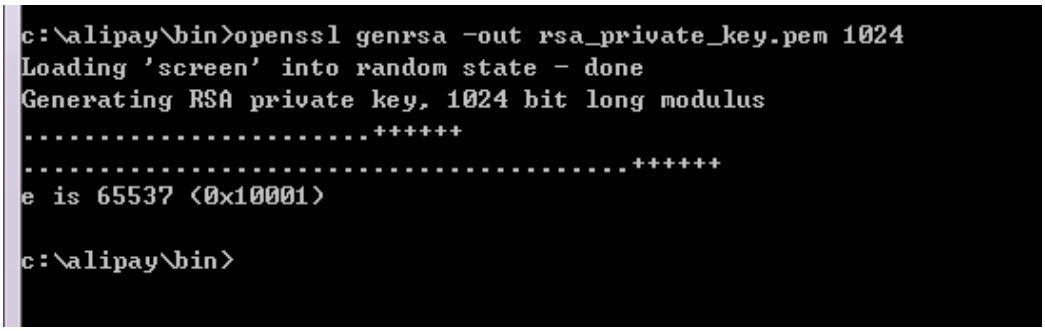
名称	大小	压缩后大小	类型
..			Folder
Android(20110907)			Folder
iOS(20110805)			Folder
J2ME(20110615)			Folder
Qt for Symbian			Folder
Symbian S60 v5 & Symbian ^3 (20110907)			Folder
Symbian_S60_v3(20110907)			Folder
openssl-0.9.8k_WIN32(RSA密钥生成工具).zip	1,309,693	1,303,238	WinRAR ZIP 压缩...
通知验签示例.zip	56,466	56,182	WinRAR ZIP 压缩...
安全支付服务端通知描述文档(20110217).pdf	314,249	279,647	文件 pdf

图 3- 2openssl

3.2.2 生成商户密钥并获取支付宝公钥

(1) 生成原始 RSA 商户私钥文件

假设解压后的目录为 c:\alipay，命令行进入目录 C:\alipay\bin，执行“*openssl genrsa -out rsa_private_key.pem 1024*”，在 C:\alipay\bin 下会生成文件 rsa_private_key.pem，其内容为原始的商户私钥（请妥善保存该文件），以下为命令正确执行截图：



```
c:\alipay\bin>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)

c:\alipay\bin>
```

图 3- 3 生成原始 RSA 商户私钥文件

(2) 将原始 RSA 商户私钥转换为 pkcs8 格式

命令行执行“ *openssl pkcs8 -topk8 -inform PEM -in rsa_private_key.pem -outform PEM -nocrypt*”得到转换为 pkcs8 格式的私钥。复制下图红框内的内容至新建 txt 文档，去掉换行，最后另存为“private_key.txt”（请妥善保存，签名时使用）。

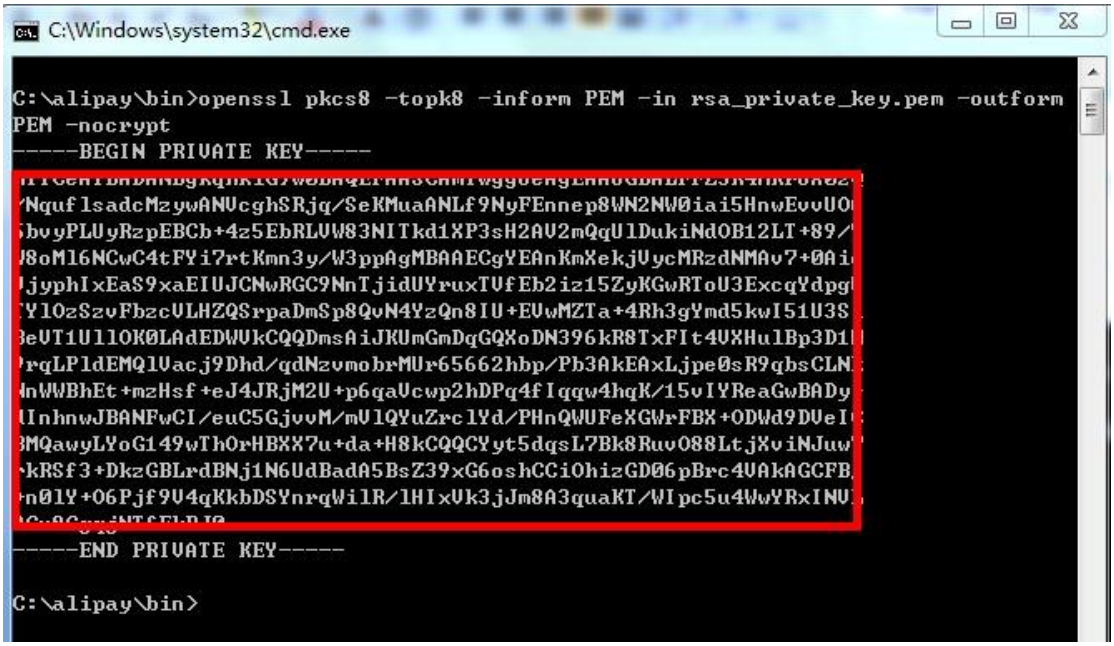


图 3-4 转换私钥格式

(3) 生成 RSA 商户公钥

命令行执行“ *openssl rsa -in rsa_private_key.pem -pubout -out rsa_public_key.pem*”，在 C:\alipay\bin 文件夹下生成文件 rsa_public_key.pem。接着用记事本打开 rsa_public_key.pem，复制全部内容至新建的 txt 文档，删除文件头“-----BEGIN PUBLIC KEY-----”与文件尾“-----END PUBLIC KEY-----”及空格、换行，如下图。最后得到一行字符串并保存该 txt 文件为“public_key.txt”。

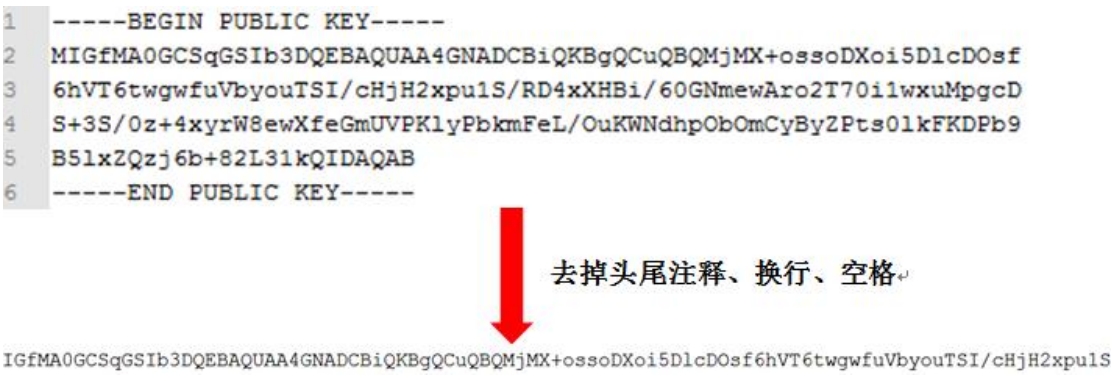


图 3-5 生成公钥

(4) 上传商户公钥至支付宝

浏览器访问 https://ms.alipay.com/index.htm 并用签约帐号登录，点击菜单栏“我的产品”，右侧点击“密钥管理”，见下图红色框内

图 3- 6 商户公钥上传

点击“上传”，选择步骤(3)生成的“public_key.txt”并完成上传。

(5) 获取 RSA 支付宝公钥

成功上传公钥至支付宝后，页面显示如下：

图 3- 7 支付宝公钥获取

其中红色框内部分即支付宝公钥，请复制至新建 txt 文档，去掉换行和空格，妥善保存（用于验签收到的支付宝通知）。

3.3 RSA 签名和验签 *

建议：签名和验签尽量在商户服务器端进行，同时一些敏感数据（如公私钥等）也应存储在服务器端，避免可能的安全隐患。

3.3.1 RSA 签名

(1) *Demo* 中提供 *RSA* 签名参考，如需使用，请添加 *demo* 中 *DataSigner.h*，*DataSigner.m*，*DataVerifier.h*，*DataVerifier.m* 四个文件以及相关 *src* 文件夹，*openssl* 文件夹目录下文件和 *libssl.a* 和 *libcrypto.a*，并在 *Targets* 的 *Bulid* 面板中 *Header Search Path* 填写相关头文件路径。

(2) 生成商品订单:

将商品信息拼接成字符串，可以参照 *demo* 中 *AlixpayOder* 中*-(NSString*)descrption* 函数。例子：出售商品（*subject*）"Iphone4"，价格（*total_fee*）"1"元，外部交易号（*out_trade_no*）"zzzz"，商品描述（*body*）为"秒杀"，订单支付完成通知 URL（*notify_url*）为"http://notify.java.jpxx.org/index.jsp"

则生成如下商品信息字串：

```
partner="xxxx"&seller="yyyy"&out_trade_no="zzzz"&subject="Ipone4"&body="秒杀"&total_fee="1"&notify_url="http%3A%2F%2Fnotify.java.jpxx.org%2Findex.jsp"
```

备注：notify_url 的值需要进行 URL 编码。

(3) 对商品信息进行 *RSA* 签名：

可参照 demo 中 *-(NSString*)signString:(NSString*)string* 函数，将签名字符串和商品信息字符串以及签名类型按格式拼接。

参数说明：（参考名称对照 Demo 里的参考函数参数）

string：待签名的字符串数据(下图中红色部分)

返回值：签名，base64 编码以及 UrlEncode 处理后的字符串（下图中蓝色部分）

签名后的订单字符串示例：

```
partner="xxxx"&seller="yyyy"&out_trade_no="zzzz"&subject="Ipone4"&body="秒杀"&total_fee="1"&notify_url="http%3A%2F%2Fnotify.java.jpxx.org%
```

```
2Findex.jsp"&sign_type="RSA"&sign="O0I1APPVQcK5bbSgdeFx9HB3Yu/U2+ak
TZ3T0/P7v3g7XD7TsQCprb6O9Nybr8CDIrztdUseQN/TCXuEvCU2cvCt1xX9UUyI6fOx
XxQFlDWx7IE2S7Zo5wOeVWmMBnCQCV8iDjcNxGHwhtCT09bVVf0wbaOiHXvAYzWlvPhy
R+0="
```

3.3.2 RSA 验签

(1) 使用 RSA 库进行验签：（参数名称为 Demo 里的以上函数名对应的参数名）

验签可以参照demo中*-(BOOL)verifyString: (NSString*)string withSign: (NSString*)signString*函数实现。

参数说明：

string：安全支付返回的待验签字符串(下图中红色部分)

signString：安全支付返回的签名字符串(下图中蓝色部分)

返回值：验签成功则返回YES，反之返回NO

以下是一个订单支付成功完整信息的示例：

```
resultStatus={9000};
result={partner="2088002007260245"&seller="2088002007260245"&out_tra
de_no="600000000006891"&subject="商品名称"&body="这是商品描述
"&total_fee="1"&notify_url="http%3A%2F%2Fnotify.java.jpxx.org%2F
index.jsp"&success="true"&sign_type="RSA"&sign="O0I1APPVQcK5bbSgdeF
x9HB3Yu/U2+akTZ3T0/P7v3g7XD7TsQCprb6O9Nybr8CDIrztdUseQN/TCXuEvCU2cvC
t1xX9UUyI6fOxXxQFlDWx7IE2S7Zo5wOeVWmMBnCQCV8iDjcNxGHwhtCT09bVVf0wbaO
iHXvAYzWlvPhyR+0="}
```

备注：安全支付服务返回的URL解析前需要*UrlDeCode*，参照*demo*中*-(AlixpayResult*)resultFromURL: (NSURL*)url*函数。

第四章 通知结果

4.1 AlixPay 方法返回的结果

详情请查看集成步骤 5，AlixPay 回调。

4.2 notify_url 通知说明

4.2.1 什么是 Notify_url

支付宝通过访问商户提供的地址的形式，将交易状态信息发送给商户服务器。商户通过支付宝的通知判断交易是否成功，具体如下：

商户地址：提供一个 http 的 URL(例:http://www.partnertest.com/servlet/NotifyReceiver)，支付宝将以 POST 方式调用该地址。

通知触发条件：交易状态发生改变，如交易从“创建”到“成功”或“关闭”。

商户返回信息：商户服务器收到通知后需返回纯字符串“success”，不能包含其他任何 HTML 等语言的文本。

通知重发：若支付宝没有收到商户返回的“success”，将对同一笔订单的通知进行周期性重发 (间隔时间为：2 分钟,10 分钟,10 分钟,1 小时,2 小时,6 小时,15 小时共 7 次)。

交易判断条件：收到 trade_status=TRADE_FINISHED（如果签有高级即时到帐协议则 trade_status=TRADE_SUCCESS）的请求后才可判定交易成功（其它 trade_status 状态请求可以不作处理）

4.2.2 Notify_url 接收数据示例

```
notify_data=<notify><partner>2088201564809153</partner><discount>0.00</discount><payment_type>1</payment_type><subject>测试商品</subject><trade_no>2012041821018998</trade_no><buyer_email>xxxx@xx.com</buyer_email><gmt_create>2012-04-18 11:06:52</gmt_create><quantity>1</quantity><out_trade_no>0418110644-1034</out_trade_no><seller_id>2088201564809153</seller_id><trade_status>TRADE_SUCCESS</trade_status><is_total_fee_adjust>N</is_total_fee_adjust><total_fee>0.01</total_fee><gmt_payment>2012-04-18 11:06:52</gmt_payment><seller_email>alipay@alipay.com</seller_email><price>0.01</price><buyer_id>2088302175666987</buyer_id><use_coupon>N</use_coupon></notify>&sign=ZPZULntRpJwFmGNIVKwjLEF2Tze7bqs60rxQ22CqT5J1UlvGo575QK9z/+p+7E9cOoRoWzqR6xHZ6WVv3dloyGKDR0btvrdqPgUAoeaX/YOWzTh00vwcQ+HBtXE+vPTfAqjCTxiiSJEOY7ATCF1q7iP3sfQxhS0nDUug1LP3OLk=
```

参数说明（注意，具体接收到的数据可能与例子有细微出入，仅确保下表内参数不变）：

notify_data：待验签数据(红色部分)，主要参数说明请见下表

sign：签名(蓝色部分)

备注：在调用验签方法时，需要将"*notify_data*="这几个字符加上，一并验签，以上红色部分

具体的验签方法请参考 3.3.2 RSA 验签

Notify_data 参数说明

参数名	说明
trade_status	用于判断交易状态，值有： TRADE_FINISHED：表示交易成功完成 WAIT_BUYER_PAY：表示等待付款 TRADE_SUCCESS：表示交易成功（高级即时到帐）
total_fee	交易金额
subject	商品名称
out_trade_no	外部交易号（商户交易号）
trade_no	支付宝交易号
gmt_create	交易创建时间
gmt_payment	交易付款时间 若交易状态是“WAIT_BUYER_PAY”则无此参数

第五章 常见问答

1. 客户端验签，报“订单信息被篡改”是什么问题？

可能有以下2种情况

a) 有可能数据在传输过程中被黑客截取和篡改

b) 检查plaintext(待签名的字符串)中是否有以下四个符号，如果参数当中包含了这四个字符也会报“订单信息被篡改”：

+加号

&连接符

“双引号

=等号

2. 客户端调用安全支付时对 body 和 subject 进行 URLEncode 会报签名错误，到底哪些需要 URLEncode？

调用安全支付接口时，只需要对参数sign进行URLEncode，其他参数都不能URLEncode，安全支付服务插件会对所有参数进行URLEncode，所以不用担心中文乱码

3. 上传商户公钥报格式错误怎么办？

首先确认上传的位置是否是RSA的下面，注意不要是DSA，无线目前不支持DSA加密；另外请检查上传的文件中是否去除注释、空格、换行等，必须是一行的字符串

附录 A 错误代码列表

以下为安全支付服务所定义的错误代码：

表 A-1 系统定义的错误代码表

错误代码	含义
9000	操作成功。
4000	系统异常。
4001	数据格式不正确。
4003	该用户绑定的支付宝账户被冻结或不允许支付。
4004	该用户已解除绑定。
4005	绑定失败或没有绑定。
4006	订单支付失败。
4010	重新绑定账户。
6000	支付服务正在进行升级操作。
6001	用户中途取消支付操作。

附录 B 安全支付服务接口

1 AlixPay Class Reference

Inherits from	NSObject
Declared in	AlixPay.h

2 pay:applicationScheme:

定义

订单支付接口。

```
+ (BOOL)pay:(NSString *)orderString
applicationScheme:(NSString *)scheme;
```

参数

orderString

包含商户信息以及订单信息的字符串。格式为：

<orderInfo>&sign="<signString>"&signType="<signType>"

其中<orderInfo>的值为 SPOrder 类的 description 方法返回的字符串

<signString>的值为对<orderInfo>进行 RSA 签名后的结果

<singType>的值默认为: @"RSA"

applicationScheme

应用程序注册的 scheme，写在相应 info.plist 中 URL types 字段。

返回值

kSPErrorOK，接口调用成功。

kSPErrorAlipayClientNotInstalled，没有安装支付宝客户端。

kSPErrorSignError，签名错误。

说明

这是一个异步方法，可以在 UI 线程中调用。

安全支付结束时，会生成一个如下的 URL：

<yourApplicationScheme>://safepay/?<query>

并且会对这个 URL 调用如下的方法：

[[UIApplication sharedApplication] openURL:url];

返回外部商户程序，需要处理该 URL

3 handleOpenURL:

定义

当安全支付回调，处理 applicationDelegate 中系统函数 application:handleOpenURL 中的 URL。

- (AlixPayResult *)handleOpenURL:(NSURL *)url;

参数

application:handleOpenURL 中的 URL，这个 URL 包含安全支付服务信息。

返回值

如果由安全支付服务打开外部商户程序，返回一个 AlixPayResult 的实例。
如果该 URL 不是由安全支付回调打开的，则返回 nil。

说明

安全支付完成后（不管是否操作成功），第三方应用都会被安全支付重新唤起，这时第三方应用需要接收回调的 url，并将返回的 url 交由 handleOpenURL 去解析安全支付服务回调信息。

4AlixPayOrder Class Reference:

Inherits from	**NSObject**
Declared in	**AlixPayOrder.h**

对订单信息的描述。参考**附录 C** 查看各个属性的含义。

4.1 partner

商家编号
@property(nonatomic, copy) NSString * partner;

4.2 seller

商家支付宝账号 ID。
@property(nonatomic, copy) NSString * seller;

4.3 tradeNO

订单号。
@property(nonatomic, copy) NSString * tradeNO;

4.4 productName

商品名称。
@property(nonatomic, copy) NSString * productName;

4.5 productDescription

商品的具体描述信息。
@property(nonatomic, copy) NSString * productDescription;

4.6 amount

商品金额（精确到小数点后两位）。
@property(nonatomic, copy) NSString * amount;

4.7 notifyURL

商户提供的 URL。
@property(nonatomic, copy) NSString * notifyURL;

说明
订单支付结束时，支付宝服务端会回调这个 URL，通知商家本次支付的结果，非必需。

注：AlixpayOrder 的作用是避免遗漏商品信息，以及避免商品信息拼接签名产生不避免错误。

5 AlixPayResult Class Reference

Inherits from	NSObject
Declared in	AlixPayResult.h

AlixPayResult 表示安全支付返回的结果，都是只读信息。

5.1 statusCode

返回安全支付的操作状态。

- (int)statusCode

返回值
安全支付的操作状态码。状态码的含义请参考**表 B**。

5.2 statusMessage

返回安全支付的操作状态信息。

- (NSString *)statusMessage

返回值
安全支付的操作状态的字符串描述。

5.3resultString

安全支付返回的结果

- (NSString *)resultString

返回值
安全支付返回的原始字符串。

5.4 signString

对 resultString 的签名字符串，方便外部商户验签结果安全性。

- (NSString *)signString

返回值
对 resultString 的签名字符串。

5.5 signType

签名类型，目前为 RSA 类型。

- (int)signString

返回值
签名类型。

注：安全支付服务回调URL信息

字段名称	描述	属性	备注
resultStatus	本次操作的状态返回值。	字符串	用来标识本次调用结果状态码。
result	本次操作返回的结果数据。	字符串	订单支付结果信息。字符串格式，形式一般如下： partner=""&seller=""&out_trade_no=""&subject=""&body=""&total_fee="30"&notify_url=""&success="true"&sign_type="RSA"&sign="xxx" 注:&success="true"&sign_type="RSA"&sign="xxx"之前的部分为商户的原始数据。 success，用来标识本次支付成功失败信息。 sign="xxx"为支付宝对本次支付结果(标红部分)的签名，商户可以使用签约时支付宝提供的公钥进行验证。

特别说明：

调用本方法的应用程序，需要通过resultStatus以及result字段的值来综合判断并确定支付结果。在resultStatus=9000，并且success="true"以及sign="xxx"校验通过的情况下，证明支付成功。其它情况归为失败。较低安全级别的场合，也可以只通过检查resultStatus以及success="true"来判定支付结果。比如，以下是一个订单支付成功完整信息的示例：

```
resultStatus={9000};
result={partner="2088002007260245"&seller="2088002007260245"&out_trade_no="6000
00000006891"&subject="商品名称"&body="这是商品描述
"&total_fee="1"&notify_url="http://notify.java.jpxx.org/index.jsp"&success="tru
e"&sign_type="RSA"&sign="O0I1APPVQcK5bbSgdeFx9HB3Yu/U2+akTZ3T0/P7v3g7XD7TsQCprb
```

```
6O9Nybr8CDIrztdUseQN/TCXuEvCU2cvCt1xX9UUyI6fOxXxQFlDWx7IE2S7Zo5wOeVWmMBnCQCV8iD
jcNxGHwhtCT09bVVf0wbaOiHXvAYzWlvPhyR+0="}
```

附录 C 订单参数信息描述

表 C-1 订单参数说明表

键	说明
partner	商家编号。 应用开发商与支付宝签约接入支付服务时，由支付宝分配。商户可以用签约支付宝账号登录 https://ms.alipay.com 获取。
seller	商家支付宝账号 id。订单付款成功后，钱会打到本账号 id 对应的支付宝账号中。商户可以用签约支付宝账号登录 https://ms.alipay.com 获取。
out_trade_no	订单编号。
subject	商品名称。
body	商品的具体描述信息。
total_fee	本次支付的总费用。所有商品的费用总和，以人民币元为单位。如 1.50。精确到小数点后两位。
notify_url	商户提供的 URL。订单支付结束时，支付宝服务端会回调这个 URL，通知商家本次支付的结果。
sign	上述订单信息的签名。
sign_type	签名类型。目前支持 RSA。