



手机网页支付接入与使用规则

手机网页即时到账授权接口

(alipay.wap.trade.create.direct)

手机网页即时到账交易接口

(alipay.wap.auth.authandexecute)

附录文档

版本号：1.0

目 录

1 文档说明	4
1.1 文档说明	4
1.2 业务术语	4
2 责任归属	4
3 技术接入规则	5
4 接口使用规则	10
5 集成流程详解	11
5.1 接入准备	11
1. 确认是否已开通手机网页支付功能	11
2. 确认签约支付宝账号	12
3. 确认网站的开发语言	12
4. 确认网站编码格式	12
5. 确认网站签名方式	12
6. 确认开发环境	13
5.2 集成	13
5.2.1 接口代码示例配置运行	13
1. JAVA配置流程（以eclipse为例）	13
2. PHP配置流程	14
3. ASP.NET(C#)配置流程	16
5.2.2 接口代码示例运行逻辑详解	17
1. JAVA（以eclipse为例）	17
2. PHP	23
3. ASP.NET(C#)	30

6 测试流程规则	37
7 附录	38
7.1 如何获得PID与密钥	38
7.2 RSA密钥生成与使用	40
7.2.1 生成商户密钥	40
1. 打开openssl密钥生成软件	40
2. 生成RSA私钥	41
3. 生成RSA公钥	42
4. 生成PKCS8 编码的私钥	43
7.2.2 RSA密钥使用逻辑	45
1. PHP开发语言使用方法	45
2. JAVA和ASP.NET(C#)开发语言使用方法	46
7.3 业务数据传递	47
7.4 如何增加扩展业务参数	47
1. JAVA修改流程	47
2. PHP修改流程	48
3. ASP.NET(C#)修改流程	48
7.5 如何更新订单	49

1 文档说明

1.1 文档说明

本文档是《手机网页即时到账接口》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。

1.2 业务术语

表1-1 业务术语

术语	解释
请求	通过 HTTP 协议把需要传输的数据发送给接收方的过程。
返回	支付宝根据得到的数据处理完成后，支付宝将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（手机网页即时到账接口.pdf）、代码示例、本文档（手机网页支付接入与使用规则.pdf）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、安全校验码 key、商户的公私钥、支付宝公钥的隐私性	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。
组成	手机网页支付产品由手机网页即时到账授权接口 (alipay.wap.trade.create.direct)、手机网页即时到账交易接口 (alipay.wap.auth.authandexecute) 两个接口组成。	使用时少了其中任何一个接口, 都无法正常使用该产品。
	必须先调用手机网页即时到账授权接口 (alipay.wap.trade.create.direct) 获得授权令牌, 再调用手机网页即时到账交易接口 (alipay.wap.auth.authandexecute) 完成付款。	手机网页即时到账交易接口 (alipay.wap.auth.authandexecute) 依赖授权令牌 (request_token) 才能完成付款, 而该授权令牌需要通过手机网页即时到账授权接口 (alipay.wap.trade.create.direct) 获取。
请求时签名逻辑	要请求的所有参数, 需要根据参数名=参数值的格式, 由字母 a 到 z 的顺序进行排序, 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。	避免报异常错误, 如: 签名不正确。
	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。	避免报异常错误, 如: 签名不正确。
	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

类型	细则	原因
	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 <code>_input_charset</code> （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 <code>utf-8</code> ，即本产品不支持 <code>GBK</code> 编码格式。	避免报异常错误，如：签名不正确。
	<code>seller_email</code> 是收款时的支付宝账号，需要与 <code>partner</code> 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 <code>RSA</code> 、 <code>MD5</code> 两种。如果要使用 <code>RSA</code> 签名时，签名方式要设置成 <code>0001</code> 。	避免签名不成功
	参数 <code>req_data</code> 中节点信息都须按照 <code>XML</code> 节点格式设置，无顺序要求。不可空的节点信息必须设置。可空的节点支持不设置，也支持设置了但是空值。	如果设置不符合 <code>XML</code> 格式或不可空节点信息设置疏漏，请求时不会报错，但是返回时支付宝会提示 <code>0004, req_data illegal</code> 。
接口构造	必须使用支付宝的网关发送请求信息给支付宝，请求网关： http://wappaygw.alipay.com/service/rest.htm 。	避免被钓鱼网站利用
	发送给支付宝的请求，请求参数不仅包含参与签名的参数，还包含参数 <code>sign</code> 。	避免报异常错误，如：签名不正确。

类型	细则	原因
	<p>发送给支付宝的请求，如果使用 form 表单传输，需要按照以下要求编写：</p> <ul style="list-style-type: none"> • action的值必须为 “http://wappaygw.alipay.com/service/rest.htm?_input_charset=该值”，如： http://wappaygw.alipay.com/service/rest.htm?_input_charset=utf-8。 • 不允许写成完整的请求链接地址，即禁止 http://wappaygw.alipay.com/service/rest.htm?后 带有所有要请求给支付宝的请求参数数据； • <form>与</form>之间需包含所有要请求给支付宝的参数，且每个参数的格式为<input type="hidden" name="参数名" value="参数值" />； • 在众多请求参数中，请求参数_input_charset（编码格式）必须存在于 form 表单中，即 form 表单中必须含有<input type="hidden" name="_input_charset" value="参数值">； • <form>与</form>之间包含的数据只允许是要请求给支付宝的参数，禁止出现商户自行命名，不在接口技术文档请求参数列表中的其他数据； • form 表单的 method 属性，可自行选择 get、post 两种。但是，手机网页即时到账交易接口 (alipay.wap.auth.authandexecute)只支持 get 方式传输。 	<ul style="list-style-type: none"> • 避免请求支付宝时报错，错误码为签名不正确； • 手机网页即时到账交易接口 (alipay.wap.auth.authandexecute)如果用 POST 提交，页面显示会为空白。
数据传输	必须使用 https 协议	避免接口无法正常使用
	在手机网页即时到账授权接口 (alipay.wap.trade.create.direct)中，建议使用模拟远程传输方式提交信息及获取返回信息。并且，提交方式仅支持 POST。	避免报异常错误（如：签名不正确），得不到授权令牌。
	在手机网页即时到账交易接口 (alipay.wap.auth.authandexecute)中，提交方式仅支持 GET。	如果用 POST 提交，页面显示会为空白。

类型	细则	原因
通知返回验证	在手机网页即时到账交易接口 (alipay.wap.auth.authandexecute) 页面跳转同步通知中, 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
	在手机网页即时到账交易接口 (alipay.wap.auth.authandexecute) 服务器异步通知中, 需要 XML 解析明文状态下的参数 notify_data, 获得节点 notify_id 的值。	避免无法正常获得通知 ID (notify_id), 导致校验是否是支付宝发送的验证失败。
	在手机网页即时到账交易接口 (alipay.wap.auth.authandexecute) 服务器异步通知中, 如果有设置服务器异步通知路径, 则必须使用获取到的参数 notify_id 再次请求支付宝, 获取是否是支付宝发送的验证结果。该请求链接是: https://mapi.alipay.com/gateway.do?partner=合作者身份 ID¬ify_id=通知 ID 的值。	验证是否是支付宝发来的请求
	在手机网页即时到账交易接口 (alipay.wap.auth.authandexecute) 服务器异步通知中, 对明文状态下的服务器异步通知参数 (除去参数 sign) 不需要先排序, 而是直接依照通知返回的参数顺序, 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, 从而得到待签名字符串。	避免验证签名不正确
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
	在手机网页即时到账授权接口 (alipay.wap.trade.create.direct) 中, 需要先对获得的返回数据做解码 (URLDECODE), 再对这些数据做字符串切割, 获得参数 res_data 的值。此时, 如果该接口设置的签名方式 sec_id 是 0001 (RSA), 还需要对参数 res_data 做 RSA 解密, 签名方式 sec_id 是 MD5 时不需要该步骤。最后对变成明文的参数 res_data 的值做 XML 解析, 获得授权令牌 request_token。	返回数据是一串编码过的字符串。需要对字符串做解码 (URLDECODE)、切割、RSA 解密、XML 解析等步骤。其中任何一步出现问题, 则无法获得授权令牌 request_token。
	在手机网页即时到账交易接口 (alipay.wap.auth.authandexecute) 页面跳转同步通知中, 可直接用 GET 方式获得返回数据。	页面跳转同步通知以纯 URL 带参数的模式返回信息

类型	细则	原因
	在手机网页即时到账交易接口 (alipay.wap.auth.authandexecute) 服务器异步通知中： <ul style="list-style-type: none"> 如果签名方式 <code>sec_id</code> 设置的是 0001 (RSA)，则需要先对参数 <code>notify_data</code> 做 RSA 解密获得明文； 如果签名方式 <code>sec_id</code> 设置的是 MD5，则不需要解密。 	当签名方式是 0001 (RSA) 时，参数 <code>notify_data</code> 是密文状态，需要解密。
	在手机网页即时到账交易接口 (alipay.wap.auth.authandexecute) 服务器异步通知中，如果要获得需要的信息，还需要对参数 <code>notify_data</code> 做 XML 解析获得子节点信息。	参数 <code>notify_data</code> 是 XML 格式的字符串
	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 <code>success</code> 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	用来在必要时检查或跟踪业务处理情况
接入环境	不能把接口嵌入 <code>iframe</code> 框架中	避免接口无法正常使用
错误码处理	请求出错啦	例如 <code>seller_account_name</code> 不是邮箱格式、请求参数缺失等请求参数不符合设置规则。
	为保障用户账户安全，禁止非手机访问。	该笔交易不是在真实手机上创建或执行的。该类情况一般出现在商户测试阶段，通过 <code>pc</code> 浏览器来请求支付宝时报错。
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 接口使用规则

表4-1 接口使用规则

类型	规范点	原因
使用时	对同一笔还没创建支付宝交易的商户订单, 支持重复调用手机网页即时到账授权接口 (alipay.wap.trade.create.direct), 但每一次的 req_id 值必须保证唯一。	每次请求 req_id 的值须保证唯一
	对同一笔还没创建支付宝交易的商户订单, 支持重复调用手机网页即时到账授权接口 (alipay.wap.trade.create.direct), 且每次返回的授权令牌都有效, 有效时间为 24 小时。	当支付宝系统中还没有为商户的某笔订单创建支付宝交易时, 商户方可以对该笔订单重复调用手机网页即时到账授权接口 (alipay.wap.trade.create.direct), 但每次返回的授权令牌不会相同。
	手机网页即时到账交易接口 (alipay.wap.auth.authandexecute) 支持重复调用, 前提是交易基本信息 (买家、卖家、交易金额、超时时间、商品名称等) 在多次调用中保持一致, 且交易尚未完成支付。	防止支付失败
	手机网页即时到账交易接口 (alipay.wap.auth.authandexecute) 重复调用时, 也支持同一个 token (授权令牌) 重复调用, 但需要保证该 token (授权令牌) 对应的交易的买家尚未确定, 即该笔交易还未在支付宝中创建。	避免报 “买家不符” 的错误
	填写买家支付宝账号时, 不能与卖家的支付宝账号相同。	避免报错, 如错误码: 创建交易失败。
	客户端内嵌 webview 方式实现, 需要开启 js 以及 css 设置。	避免出现因内嵌引起的兼容性或者界面布局异常、乱码等问题。
	在请求参数中不能存在影响请求格式的特殊字符, 如: url、xml 格式的特殊字符等。	避免交易失败
	建议商户的业务判断依据以 notify_url (服务器异步通知) 为准, call_back_url 页面文件只做界面展示用途。	避免商户订单数据与支付宝交易数据不同步
	本接口支持的众多支付通道中, 储蓄卡支付和信用卡支付一旦进入手机网银系统页面, 支付宝将无法控制订单的支付效率问题。	各网银系统的运作及各网银与支付宝间的交互因为网络或其他因素导致支付时的速度快慢不一

类型	规范点	原因
业务应用注意事项	支付宝支付手续费扣取模式分为两种： <ul style="list-style-type: none">实时扣费模式：一般都是收款账户扣费，或采用合同指定固定扣款支付宝账户扣费；月结手续费模式：实时交易不扣手续费，月底账单统一结算。	实时扣费时，在扣费支付宝账号的账务明细中可马上查到扣费记录；但月结模式下，则不会查到。
	在集成接口的过程中，如果商户使用 <code>seller_email</code> 作为收款账户，则当需要修改该支付宝账户名称时，商户必须在程序中做相应修改。	避免支付过程中会出现卖家账户不存在的错误
测试时	调试阶段时，建议使用 1 分钱（0.01 元）进行测试	降低测试成本
网络	确保网络顺畅	防止支付时或支付完成后的订单同步出现各类异常

5 集成流程详解

5.1 接入准备

1. 确认是否已开通手机网页支付功能

使用签约支付宝账号登录<https://b.alipay.com/order/serviceIndex.htm>，订单状态为“已完结”表示手机网页支付权限已开通，如下图所示。



图5-1 查询订单状态

如果无法确认，建议联系与支付宝签约协议时的贵公司业务人员。

2. 确认签约支付宝账号

集成时需要使用签约的支付宝账号相关信息，包括：支付宝登录账号、合作者身份ID（PID）、交易安全校验码（KEY）。获得方法请参见“7.1 如何获得PID与密钥”。

3. 确认网站的开发语言

开发语言是用指定的一种语言来开发网站。支付宝提供的代码示例目前包含 PHP、ASP.NET、JAVA 三种。

4. 确认网站编码格式

查询网站使用的是何种编码格式，可通过在网站网页上点击右键，找到“编码”，打开扩展箭头后查看，默认选中的编码即为网站的编码格式。支付宝提供的代码示例目前仅包含 UTF-8 一种。

5. 确认网站签名方式

签名方式指的是对字符串做加密与解密时的方法。支付宝提供的代码示例目前包含 RSA、MD5 两种。

6. 确认开发环境

- 本机电脑调试环境
各开发语言对应的开发环境、调试工具等。
- 服务器调试环境
 - 服务器需要支持各开发语言对应的环境，如：支持 PHP 开发语言的 SSL 服务、CURL 服务等。
 - 能够把本地文件上传到服务器上的通道与权限，如：FTP，SFTP 传输。

5.2 集成

5.2.1 接口代码示例配置运行

解压下载到的接口资料压缩文件（文件名是 WS_WAP_PAYWAP.zip），根据商户自身情况选择 JAVA（文件夹名为 WS_WAP_PAYWAP-JAVA-UTF-8）、PHP（文件夹名为 WS_WAP_PAYWAP-PHP-UTF-8）、C#（文件夹名为 WS_WAP_PAYWAP-CSHARP-UTF-8）三种开发语言代码示例文件夹中的一个。

1. JAVA配置流程（以eclipse为例）

步骤1： 导入代码

启动 JAVA 开发软件，把“WS_WAP_PAYWAP-JAVA-UTF-8”文件夹中的“src\com”文件夹、“WebRoot\images”文件夹、“WebRoot\log”文件夹、“WebRoot\WEB-INF\lib”文件夹、“WebRoot\alipayapi.jsp”、“WebRoot\call_back_url.jsp”、“WebRoot\notify_url.jsp”全部导入到开发环境中。“src\com”文件夹中的文件是类文件，无法与支付宝 PC 端的其他接口混用。

步骤2： 配置基本信息

在 JAVA 开发环境中，打开“src\com\alipay\config”类目下“AlipayConfig.java”文件，配置 partner（合作身份者 ID）、key（交易安全校验码）、private_key（商户私钥）、public_key（支付宝公钥）、sign_type（签名类型）等基本信息参数。

表5-1 JAVA 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”。

参数	含义
key	交易安全检验码，由数字和字母组成的 32 位字符串。当sign_type设置为MD5 时，该参数设置才有效。请参考“7.1 如何获得PID与密钥”。
private_key	商户方的私钥。当sign_type设置为 0001 时，该参数设置才有效。请参考“7.2 RSA密钥生成与使用”。
public_key	支付宝的公钥。当sign_type设置为 0001 时，该参数设置才有效。请参考“7.2 RSA密钥生成与使用”。
sign_type	签名方式。在手机网页支付的接口中仅支持两种签名方式，MD5 和 RSA。对该参数设置的值分别是 MD5、0001。
input_charset	字符编码格式。在手机网页支付的接口中仅支持 utf-8。无需修改。

步骤3： 配置业务信息

打开“alipayapi.jsp”文件。配置req_id（请求号）、notify_url（服务器异步通知页面路径）、call_back_url（页面跳转同步通知页面路径）、seller_email（卖家支付宝帐户）、out_trade_no（商户订单号）、subject（订单名称）、total_fee（付款金额）这几个常用业务参数信息。请参考“7.3 业务数据传递”和“7.4 如何增加扩展业务参数”。

步骤4： 编写同步通知业务逻辑

打开“call_back_url.jsp”文件。在注释指定位置“//请在这里加上商户的业务逻辑程序代码”写入商户的业务逻辑代码。请参考“7.5 如何更新订单”。

步骤5： 编写异步通知业务逻辑

打开“notify_url.jsp”文件（notify_url.jsp需保证无任何HTML代码）。在注释指定位置“//请在这里加上商户的业务逻辑程序代码”写入商户的业务逻辑代码。请参考“7.5 如何更新订单”。需要注意的是，签名方式为 0001 时，需要先解密，MD5 时则不需要。解密后，需要XML解析方可获取参数信息。

2. PHP配置流程

步骤1： 导入代码

启动 PHP 开发软件，把“WS_WAP_PAYWAP-PHP-UTF-8”文件夹中除去“readme.txt”与“openssl”文件夹外的其他所有文件全部导入到开发环境中。“lib”文件夹中的文件是类文件，无法与支付宝 PC 端的其他接口混用。

步骤2： 配置基本信息

在 PHP 开发环境中，打开“alipay.config.php”文件，配置 partner（合作身份者 ID）、key（交易安全校验码）、private_key（商户私钥）、public_key（支付宝公钥）、sign_type（签名类型）等基本信息参数。

表5-2 PHP 基本信息配置

参数	解释
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”。
key	交易安全检验码，由数字和字母组成的 32 位字符串。当sign_type设置为MD5时，该参数设置才有效。请参考“7.1 如何获得PID与密钥”。
private_key	商户方的私钥。当sign_type设置为 0001 时，该参数设置才有效。请参考“7.2 RSA密钥生成与使用”。
public_key	支付宝的公钥。当sign_type设置为 0001 时，该参数设置才有效。请参考“7.2 RSA密钥生成与使用”。
sign_type	签名方式。在手机网页支付的接口中仅支持两种签名方式，MD5 和 RSA。对该参数设置的值分别是 MD5、0001。
input_charset	字符编码格式。在手机网页支付的接口中仅支持 utf-8。无需修改。
cacert	ca 证书路径地址，用于 curl 中 ssl 校验。需要保证 cacert.pem 文件在当前文件夹目录中。无需修改。
transport	访问模式，根据自己的服务器是否支持 ssl 访问，若支持请选择 https；若不支持请选择 http。该参数只关联到同步通知或异步通知时，验证是否是支付宝发来的请求功能中的支付宝请求地址。无需修改。

步骤3： 配置业务信息

打开“alipayapi.php”文件。配置req_id（请求号）、notify_url（服务器异步通知页面路径）、call_back_url（页面跳转同步通知页面路径）、seller_email（卖家支付宝帐户）、out_trade_no（商户订单号）、subject（订单名称）、total_fee（付款金额）这几个常用业务参数信息。请参考“7.3 业务数据传递”和“7.4 如何增加扩展业务参数”。

步骤4： 编写同步通知业务逻辑

打开“call_back_url.php”文件。在注释指定位置“//请在这里加上商户的业务逻辑程序代码”写入商户的业务逻辑代码。请参考“7.5 如何更新订单”。

步骤5： 编写异步通知业务逻辑

打开“notify_url.php”文件（notify_url.php需保证无任何HTML代码）。在注释指定位置“//请在这里加上商户的业务逻辑程序代码”写入商户的业务逻辑代码。请参考“7.5 如何更新订单”。需要注意的是，签名方式为 0001 时，需要先解密，MD5 时则不需要。解密后，需要XML解析方可获取参数信息。

3. ASP.NET(C#)配置流程

步骤1： 导入代码

启动.NET 开发软件，把“WS_WAP_PAYWAP-CSHARP-UTF-8”文件夹中除去“readme.txt”与“Web.Config”外的其他所有文件全部导入到开发环境中。为防止占用“default.aspx”的名称，可以把“default.aspx”与“default.aspx.cs”两个文件中的核心代码拷贝到新页面文件中。 “app_code”文件夹中的文件是类文件，无法与支付宝 PC 端的其他接口混用。

步骤2： 配置基本信息

在.NET 开发环境中，打开“app_code”类目下的“AlipayConfig.cs”文件，配置 partner（合作身份者 ID）、key（交易安全校验码）、private_key（商户私钥）、public_key（支付宝公钥）、sign_type（签名类型）等基本信息参数。

表5-3 ASP.NET(C#)基本信息配置

参数	解释
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”。
key	交易安全校验码，由数字和字母组成的 32 位字符串。当sign_type设置为MD5 时，该参数设置才有效。请参考“7.1 如何获得PID与密钥”。
private_key	商户方的私钥。当sign_type设置为 0001 时，该参数设置才有效。请参考“7.2 RSA密钥生成与使用”。
public_key	支付宝的公钥。当sign_type设置为 0001 时，该参数设置才有效。请参考“7.2 RSA密钥生成与使用”。
sign_type	签名方式。在手机网页支付的接口中仅支持两种签名方式，MD5 和 RSA。对该参数设置的值分别是 MD5、0001。
input_charset	字符编码格式。在手机网页支付的接口中仅支持 utf-8。无需修改。

步骤3： 配置业务信息

打开“default.aspx.cs”文件。配置req_id（请求号）、notify_url（服务器异步通知页面路径）、call_back_url（页面跳转同步通知页面路径）、seller_email（卖家支付宝帐户）、out_trade_no（商户订单号）、subject（订单名称）、total_fee

（付款金额）这几个常用业务参数信息。请参考“7.3 业务数据传递”和“7.4 如何增加扩展业务参数”。

步骤4: 编写同步通知业务逻辑

打开“call_back_url.aspx”、“call_back_url.aspx.cs”文件。call_back_url.aspx 为前端页面可显示一些信息供用户查看，call_back_url.aspx.cs 为后台业务代码编写处。

在注释指定位置“//请在这里加上商户的业务逻辑程序代码”写入商户的业务逻辑代码。请参考“7.5 如何更新订单”。

步骤5: 编写异步通知业务逻辑

打开“notify_url.aspx.cs”文件。notify_url.aspx 需保证无任何 HTML 代码，notify_url.aspx.cs 为后台业务代码编写处。

在注释指定位置“//请在这里加上商户的业务逻辑程序代码”写入商户的业务逻辑代码。请参考“7.5 如何更新订单”。需要注意的是，签名方式为 0001 时，需要先解密，MD5 时则不需要。解密后，需要XML解析方可获取参数信息。

5.2.2 接口代码示例运行逻辑详解

1. JAVA（以eclipse为例）

步骤1: 商户为手机网页即时到账授权接口(alipay.wap.trade.create.direct)的请求参数配置好数据后，组装进数组 sParaTempToken 中，再对该数组做空值过滤（调用 AlipayCore 类中的 paraFilter 函数）。

```
{req_data=<direct_trade_create_req><notify_url>http://www.xxx.com/WS_WAP_PAYWAP-JAVA-UTF-8/notify_url.jsp</notify_url><call_back_url>http://127.0.0.1:8080/WS_WAP_PAYWAP-JAVA-UTF-8/call_back_url.jsp</call_back_url><seller_account_name>alipayrisk10@alipay.com</seller_account_name><out_trade_no>cs201306170001</out_trade_no><subject>测试</subject><total_fee>0.01</total_fee></direct_trade_create_req>, service=alipay.wap.trade.create.direct, sec_id=0001, partner=2088501624560335, _input_charset=utf-8, req_id=201306170001, v=2.0, format=xml}
```

步骤2: 对该数组做参数名首字母升序的排序动作，此时的数组为待签名数组，并把待签名数组转变成待签名字符串（调用AlipayCore类中的createLinkString函数），排序规则请参见“3 技术接入规则”中的“请求时签名逻辑”。

```
_input_charset=utf-8&format=xml&partner=2088501624560335&req_data=<direct_trade_create_req><notify_url>http://www.xxx.com/WS_WAP_PAYWAP-JAVA-UTF-8/notify_url.jsp</notify_url><call_back_url>http://127.0.0.1:8080/WS_WAP_PAYWAP-JAVA-UTF-8/call_back_url.jsp</call_back_url><seller_account_name>al
```

```
ipayrisk10@alipay.com</seller_account_name><out_trade_no>cs201306170001</out_trade_no><subject>测试</subject><total_fee>0.01</total_fee></direct_trade_create_req>&req_id=201306170001&sec_id=0001&service=alipay.wap.trade.create.direct&v=2.0
```

步骤3: 对该字符串做签名（调用 AlipaySubmit 类中的 buildRequestMysign 函数），规则参见“3 技术接入规则”中的“请求时签名逻辑”。

```
O8DH9HG2yRkbUrIz0jtKaPZRcWqggcuGEtI6576Yibh60iX7azXTq2D4ZYWaX16kBy8MNSE4PBuPcrxe66kQW/Ufhoa/f+0jeUx5RC3/A0as+/vOOCmQnO0VTyRTp2DHLvcL3k9EKow6FsFuQH ZPpHsQnzzRUSlQ+vGI+dPYUSU=
```

步骤4: 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。

```
{req_data=<direct_trade_create_req><notify_url>http://www.xxx.com/WS_WAP_PAYWAP-JAVA-UTF-8/notify_url.jsp</notify_url><call_back_url>http://127.0.0.1:8080/WS_WAP_PAYWAP-JAVA-UTF-8/call_back_url.jsp</call_back_url><seller_account_name>ipayrisk10@alipay.com</seller_account_name><out_trade_no>cs201306170001</out_trade_no><subject>测试</subject><total_fee>0.01</total_fee></direct_trade_create_req>,service=alipay.wap.trade.create.direct,partner=2088501624560335,sec_id=0001,_input_charset=utf-8,req_id=201306170001,sign=O8DH9HG2yRkbUrIz0jtKaPZRcWqggcuGEtI6576Yibh60iX7azXTq2D4ZYWaX16kBy8MNSE4PBuPcrxe66kQW/Ufhoa/f+0jeUx5RC3/A0as+/vOOCmQnO0VTyRTp2DHLvcL3k9EKow6FsFuQH ZPpHsQnzzRUSlQ+vGI+dPYUSU=,format=xml,v=2.0}
```

步骤5: 模拟远程 HTTP 协议请求支付宝（调用 AlipaySubmit 类中的 buildRequest (String ALIPAY_GATEWAY_NEW, String strParaFileName, String strFilePath, Map<String, String> sParaTemp) throws Exception 函数），得到支付宝的处理结果，字符串文本格式。

```
res_data=wzpw16nSd%2B0aYMQaltXjmEdJgeAkLH2dupVrHNNKknd33wwqgbd1hj3gAur%2F1WTLsFWQcw8Q%2Fb8XKk8r0Ti14pTbH9wzhyOeElGfd4Ks1HKkqZFXLX7YVhfBH71HjJiNFBzMZhMAGy%2BhAgiwbw4mvIjQF%2Bu%2BGS3SRK%2B4wy2QmptEEcqVseMgHr56MYefdvxzb%2FT06%2B%2Fg5pdIre94H6ZL0mDR8IobzQnJV1mq%2B9YMXNUSpsRzXCHhcbKEq7xqDVde5eb43LvNKA8ypR9hMiYd5%2FOLAuYnpetM81IzGNC83kioh5hsSoLgEErCrMklxucR7X5U6122APeK6i%2FbT8TA%3D%3D&service=alipay.wap.trade.create.direct&sec_id=0001&partner=2088501624560335&req_id=201306170001&sign=112LB1P66aZArPUJV9Vd3Tfo6m7PLiI8IQRNjjwFjg6NXS4PCjTdY6PlCnB5MmlnNFurI2J5GuL7LUCKVSl058hUpXqRllpoYNorKx%2BUKXYOAmPilRa2zHry%2BjVU1W9qNp4jLI2oDY8lgBbKr%2BkL3BYO0GU0Stv4zAhokAaF9ho%3D&v=2.0
```

步骤6: 对该结果做 URLDECODE 解码工作（调用 urldecode 函数）。

```
res_data=wzpw16nSd+0aYMQaltXjmEdJgeAkLH2dupVrHNNKknd33wwqgbd1hj3gAur/1WTL
```

```
SFWQcw8Q/b8XKk8r0Ti14pTbH9wzhyOeElGfd4Ks1HKkqZFXLX7YVhfBH7lHjJiNFBzMZhMAG
y+hAgiwbw4mvIjQF+u+GS3SRK+4wy2QmptEEcqVseMgHr56MYeFfdvxzbX/T06+/g5pdIre94
H6ZL0mDR8IobzQnJVlmq+9YMXNUSpsRzXCHhcBKEq7xqDVdE5eb43LvNKA8ypR9hMiYd5/OLa
uYnpetM8lIzGNC83kioh5hsSoLgEErCrMklxucR7X5U6l22APeK6i/bT8TA==&service=ali
pay.wap.trade.create.direct&sec_id=0001&partner=2088501624560335&req_id=2
01306170001&sign=1l2LB1P66aZArPUJV9Vd3Tfo6m7PLiI8IQRNjjwFjg6NXS4PCjTdY6Pl
CnB5Mml1NFurI2J5GuL7LUCKVSl058hUpXQrllpoYNorKx+UKXYOAmPilRa2zHry+jVU1W9qN
p4jLI2oDY8lgBbKr+kL3BYO0GU0Stv4zAhokAaF9ho=&v=2.0
```

步骤7: 对这串结果字符串做字符串切割，获取返回参数的信息。注意，如果签名方式设置的是 0001（RSA），那么还需要对这结果中的返回参数 **res_data** 做解密工作；如果签名方式设置的是 MD5，则结果中的返回参数 **res_data** 直接显示明文。

```
res_data=<?xml version="1.0"
encoding="utf-8"?><direct_trade_create_res><request_token>2013061725bc6b9
78788d6f2459d0af9a49ea2f3</request_token></direct_trade_create_res>&servi
ce=alipay.wap.trade.create.direct&sec_id=0001&partner=2088501624560335&re
q_id=201306170001&sign=1l2LB1P66aZArPUJV9Vd3Tfo6m7PLiI8IQRNjjwFjg6NXS4PCj
TdY6PlCnB5Mml1NFurI2J5GuL7LUCKVSl058hUpXQrllpoYNorKx+UKXYOAmPilRa2zHry+jV
U1W9qNp4jLI2oDY8lgBbKr+kL3BYO0GU0Stv4zAhokAaF9ho=&v=2.0
```

接着把 XML 结构的 **res_data** 的值做 XML 解析，从中获取子节点 **request_token** 的值（这一系列过程需调用 **AlipaySubmit** 类中的 **parseResponse** 函数），这个值就是授权令牌。

```
2013061725bc6b978788d6f2459d0af9a49ea2f3
```

步骤8: 商户为手机网页即时到账交易接口(**alipay.wap.auth.authandexecute**)的请求参数配置好数据后，与步骤一、二、三、四同理，把数据组装进数组 **sParaTemp** 中。

```
{req_data=<auth_and_execute_req><request_token>2013061725bc6b978788d6f245
9d0af9a49ea2f3</request_token></auth_and_execute_req>,
service=alipay.wap.auth.authAndExecute, sec_id=0001,
partner=2088501624560335, _input_charset=utf-8, v=2.0, format=xml}
```

对该数组做空值过滤（调用 **AlipayCore** 类中的 **paraFilter** 函数），对该数组做参数名首字母升序的排序动作，把待签名数组转变成待签名字符串，对该字符串做签名（调用 **AlipaySubmit** 类中的 **buildRequestMysign** 函数），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。

```
{req_data=<auth_and_execute_req><request_token>2013061725bc6b978788d6f245
9d0af9a49ea2f3</request_token></auth_and_execute_req>,
service=alipay.wap.auth.authAndExecute, partner=2088501624560335,
sec_id=0001, _input_charset=utf-8,
sign=jqUVcsnO3CX2c3/b2WqALKL+AJgxnB9rT87LJ31BBHqNd6DixUmt4ht6qz9TqpX9nuI
+EaPnYIhPcF+okas2bGn7ju4V3TS0805g/7mELsuW8/kcG0d/+zLvLqH7tXY0oilb4Bq/fTb8
```

```
FewVuBWSMkZXIzovc9he0cldTk1XPQ=, format=xml, v=2.0}
```

步骤9: 以 GET 方式, 请求支付宝 (调用 AlipaySubmit 类中的 buildRequest(String ALIPAY_GATEWAY_NEW, Map<String, String> sParaTemp, String strMethod, String strButtonName)函数), 此时商户的页面会自动跳转至支付宝收银台。



图5-2 跳转至支付宝收银台

后面的动作全由买家在支付宝收银台中操作完成。

步骤10: 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,并提示用户可返回商户网站。买家点击该功能按钮后,则当前界面会跳转回商户设置的 `call_back_url` 页面文件去,如果长时间没有点击,则当前支付宝收银台界面会自动跳转至商户 `call_back_url` 页面文件去。此时商户的 `call_back_url` 页面会收到一串带有处理结果数据的链接地址。

```
http://127.0.0.1:8080/WS_WAP_PAYWAP-JAVA-UTF-8/call_back_url.jsp?out_trade_no=cs201306170001&request_token=requestToken&result=success&trade_no=2013061741934035&sign=UCUm6GVdi7MEcNnDbD7UnG0UyEyUtuKOSuhPQbrGxUvLUmRJectMm5NxYV%2FtvWT0VCGbM%2F4zaAlYy1EP4bFLisoyprq%2FQ9H5XMn0%2FOZ5E1XclGXid%2FLMbMGoYCqNmI22MxMZ0OmL9UoUSrT2f4%2B8EDCtkDsnUsG82K4GnjSmJ2w%3D&sign_type=0001
```

步骤11: 用 GET 方式获得这些返回数据,与步骤一、二、三、四同理,对数组 `params` 做空值过滤、参数名首字母升序的排序、待签名数组转变成待签名字符串。

```
out_trade_no=cs201306170001&request_token=requestToken&result=success&trade_no=2013061741934035
```

对该字符串做签名验证(这一系列动作为调用 `AlipayNotify` 类中的 `verifyReturn` 函数完成),得到布尔类型的验证结果,完成 `call_back_url` 页面文件的验证动作。

步骤12: 买家对一笔交易付款完成的同时,除了支付宝会触发页面跳转至 `call_back_url` 文件的同步通知动作,还会自动触发服务器异步通知页面文件(商户设置参数 `notify_url` 的值)。此时,商户的 `notify_url` 文件也会收到支付宝异步通知回来的数据。

```
{service=alipay.wap.trade.create.direct,sign=V2fgcO2jwZ5G/HcDRS3N7m2TNqRSHFqvF7wQ6xD52NgLtNMhx2H+4H/yv5yKZxt1xY5K1aMfv6Ms00fpKAYZaZfG+WYs925IbAzpscbK72fzuhvfikjbEvuOLHY+LGNXwkn+VhQBbNkHs4GNJSM1iZ1jcXm8NU9oANviZDMAcg=,sec_id=0001,v=1.0,notify_data=0R5IeCoVWRerPIERgTFRQOUfIiw0FHmHGe6elY6VYp1WZU5bRsQkK5ekMcceD412gO+KgqPzvAKVQG1h3dwuhWgaHPdL5oNj1eIzf4PUuRYOb7qCwFmqpNDQ9Jznbm89kCqjenhWSIWQ/2Aad+4RPHOKy9AZAWut02MiW7g8gH49Nblrjki+u7QQN+xQ/Q/Fr4f2RlcoiutfsYB2mCy8smtjaH2LmE5wNz0I5yHMxnf9+NCKjm0JdZZrOC2wXly/tzLY6w+zef+m5s184D80HejiUPolS8dBtmEPs+SQzA5N7eu0BF1pPj9m7wyTINF6k4G1kQtvNM6UZd2cwuthwZeYcax5Db0rWYyDaQOCH/gEUSHzD6za84Gpj9/grKnvMy65C7ihmcb4wtA7CniQSnLiQZ3ksUGjb0WiWtpL9JeqQyVWJZiSpRxt73B9/qT1hzf5K1CWHTzD0zf0jifFbSrXQmAt5oIoMM49W7Ds95fTZNde3czf/hrcUQEi3LNW+6e975+Vl40+fmxxBeGaIlLs1EBTdZpDoNl35U7uhzdKbBzJR4AYY6H9xD2QmjtAS3GelIXpiqC29FbwwFLbEQM/a9xAS3KS8jiYDWEAK4dJi3U+F/QTismfjp7dfT2KqU89wsqJ+BtMwWLaOPvz1BSpj1TVYYmgK5MdLjh6S+GVH4ka/RcU2nGqp01cNHBHa0j2wY7Uu1p3SiSa62NTbu0wvLqiT3fs6K6PDwVxGDAPsQaXvCGrMRhhVLc7M5AcCVSBCJd4LAJU/BfTcYZjiFcKS1SSYKMFZRusNcFtzrbvilu4gwhRNRyrl1Z/pwGbQ5LtxV0831XtEkJ1fWFzt84hxfCOGH+o9EgJBKyIahQfyHfRir5quiixBqp5xWR8DU5nJNAW+/MLgPzNYt+uaIPadqrBt/j7+0gdgF
```

```
EdLzPLncI9h0VdlmdgO+CvxJlaKYi2pDY64oTzm87CN/Fiofln8SjZRgGh/ClxIJHunI9DCmV
szMtVkQhJwQzkIj9YnV+3n0Q3wj0s3iE69WJjHk+DpN8ARZiAJEFKtGT9lMnIf1JfbpXmdMla
rFVldoMEb8vKLN4kHioR4DhVyDvaLfo4rsgvGcLSB65R4Ya4tVdkle0+vnk+U/NMk9zrJf39h
InyBoazUpkPIqmmQIAvkpSbnbGuZUdqri9fame2e/gPkUzYqzo/gt6OCuzIdtVtVI0kQXGfCz
9XpazmYbTcKA9Ck6z6WF2lN7lIJH9DSNB7lCDM/cKD7xEZ5lFRucz1sOsvZNpNhH8R7EnyQYE
o7NM9ph8gDfgaVaaWh1+13TJfwvC5WBOi+5KyKh+6qb4Z98tFkxrqyU2Kn6hFAoQ==}
```

步骤13: 用 POST 方式获得这些返回数据。如果签名方式设置的是 0001 (RSA) 时, 需要先对这些数据中的参数 `notify_data` 做解密; 如果签名方式设置的是 MD5, 因为直接返回明文, 所以不需要解密。

```
{service=alipay.wap.trade.create.direct,
sign=V2fgc02jwZ5G/HcDRS3N7m2TNqRSHFqvF7wQ6xD52NgLtNMhx2H+4H/yv5yKZxt1xY5K
1aMfv6Ms00fpKAYZaZFG+Wys925IbAzpscbK72fzuhvikjbEvuOLHY+LGNXwn+VhQBBNkHs
4GNJSMliZljcCxm8NU9oANviZDMacg=, sec_id=0001, v=1.0,
notify_data=<notify><payment_type>1</payment_type><subject>测试
</subject><trade_no>2013061741934035</trade_no><buyer_email>xxxxxxx@xx.co
m</buyer_email><gmt_create>2013-06-17
19:15:58</gmt_create><notify_type>trade_status_sync</notify_type><quantit
y>1</quantity><out_trade_no>cs201306170001</out_trade_no><notify_time>201
3-06-17
19:16:09</notify_time><seller_id>2088501624560335</seller_id><trade_statu
s>TRADE_FINISHED</trade_status><is_total_fee_adjust>N</is_total_fee_adjus
t><total_fee>0.01</total_fee><gmt_payment>2013-06-17
19:16:08</gmt_payment><seller_email>alipayrisk10@alipay.com</seller_email
><gmt_close>2013-06-17
19:16:08</gmt_close><price>0.01</price><buyer_id>2088002396712354</buyer_
id><notify_id>78fca98191478a5e64459d8ef88601183y</notify_id><use_coupon>N
</use_coupon></notify>}
```

`notify_url` 异步通知的签名验证规律与 `call_back_url` 不同, 商户只需要顺序获取支付宝的返回数据, 不需要对这些参数排序。因此, 直接把数组 `params` 转变成待签名字符串。

```
service=alipay.wap.trade.create.direct&v=1.0&sec_id=0001&notify_data=<not
ify><payment_type>1</payment_type><subject>测试
</subject><trade_no>2013061741934035</trade_no><buyer_email>xxxxxxx@xx.co
m</buyer_email><gmt_create>2013-06-17
19:15:58</gmt_create><notify_type>trade_status_sync</notify_type><quantit
y>1</quantity><out_trade_no>cs201306170001</out_trade_no><notify_time>201
3-06-17
19:16:09</notify_time><seller_id>2088501624560335</seller_id><trade_statu
s>TRADE_FINISHED</trade_status><is_total_fee_adjust>N</is_total_fee_adjus
t><total_fee>0.01</total_fee><gmt_payment>2013-06-17
19:16:08</gmt_payment><seller_email>alipayrisk10@alipay.com</seller_email
```



```
<gmt_close>2013-06-17
19:16:08</gmt_close><price>0.01</price><buyer_id>2088002396712354</buyer_id>
<notify_id>78fca98191478a5e64459d8ef88601183y</notify_id><use_coupon>N
</use_coupon></notify>
```

接着对该字符串做签名验证（这一系列动作调用 `AlipayNotify` 类中的 `verifyNotify` 函数），得到布尔类型的验证结果，完成 `notify_url` 页面文件的验证动作。

步骤14: 在 `notify_url` 页面文件中，参数 `notify_data` 是 XML 结构的字符串，需要先 XML 解析，再获取需要的子节点数据。

2. PHP

步骤1: 商户为手机网页即时到账授权接口(`alipay.wap.trade.create.direct`)的请求参数配置好数据后,组装进数组`$para_token`中,再对该数组做空值过滤(调用 `AlipayCore` 类中的 `paraFilter` 函数)。

```
$para_token = array(
    "service" => "alipay.wap.trade.create.direct",
    "partner" => "2088501624560335",
    "sec_id" => "0001",
    "format" => "xml",
    "v" => "2.0",
    "req_id" => "201306140001",
    "req_data" =>
"<direct_trade_create_req><notify_url>http://www.xxx.com/WS_WAP_PAYWAP-PHP-UTF-8/notify_url.php</notify_url><call_back_url>http://127.0.0.1:8800/WS_WAP_PAYWAP-PHP-UTF-8/call_back_url.php</call_back_url><seller_account_name>alipayrisk10@alipay.com</seller_account_name><out_trade_no>cs201306140001</out_trade_no><subject>测试</subject><total_fee>0.01</total_fee></direct_trade_create_req>",
    "_input_charset" => "utf-8"
);
```

步骤2: 对该数组做参数名首字母排序的排序动作（调用`AlipayCore`类中的`argSort`函数）（排序规则请参见“3 技术接入规则”中的“请求时签名逻辑”），此时得到的数组为待签名数组。

```
$para_token = array(
    "_input_charset" => "utf-8",
    "format" => "xml",
    "partner" => "2088501624560335",
    "req_data" =>
"<direct_trade_create_req><notify_url>http://www.xxx.com/WS_WAP_PAYWAP-PHP-UTF-8/notify_url.php</notify_url><call_back_url>http://127.0.0.1:8800/W
```

```
S_WAP_PAYWAP-PHP-UTF-8/call_back_url.php</call_back_url><seller_account_name>alipayrisk10@alipay.com</seller_account_name><out_trade_no>cs201306140001</out_trade_no><subject>测试</subject><total_fee>0.01</total_fee></direct_trade_create_req>",
    "req_id" => "201306140001",
    "sec_id" => "0001",
    "service" => "alipay.wap.trade.create.direct",
    "v" => "2.0"
);
```

步骤3: 把待签名数组转变成待签名字符串。

```
_input_charset=utf-8&format=xml&partner=2088501624560335&req_data=<direct_trade_create_req><notify_url>http://www.xxx.com/WS_WAP_PAYWAP-PHP-UTF-8/notify_url.php</notify_url><call_back_url>http://127.0.0.1:8800/WS_WAP_PAYWAP-PHP-UTF-8/call_back_url.php</call_back_url><seller_account_name>alipayrisk10@alipay.com</seller_account_name><out_trade_no>cs201306140001</out_trade_no><subject>测试</subject><total_fee>0.01</total_fee></direct_trade_create_req>&req_id=201306140001&sec_id=0001&service=alipay.wap.trade.create.direct&v=2.0
```

并对该字符串做签名（调用AlipaySubmit类中的buildRequestMysign函数），转变规则请参见“3 技术接入规则”中的“请求时签名逻辑”。

```
EGB8F+aWwBhotwNSQwF9Sik2OzQffV1Val1STvrRj4MzElW4pLM6Mfjk3/4Vj3ci4Pspcm1OLDHPZ2JQNsFruS6d4bFSQrtEF52irwSF+re42+TBTRe7w8TZAjBnBBLlknerq0DvN86SfA6QaXOl/nOfgcMfCx3YtJnX9u2Yi/k=
```

步骤4: 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。

```
$para_token = array(
    "_input_charset" => "utf-8",
    "format" => "xml",
    "partner" => "2088501624560335",
    "req_data" =>
    "<direct_trade_create_req><notify_url>http://www.xxx.com/WS_WAP_PAYWAP-PHP-UTF-8/notify_url.php</notify_url><call_back_url>http://127.0.0.1:8800/WS_WAP_PAYWAP-PHP-UTF-8/call_back_url.php</call_back_url><seller_account_name>alipayrisk10@alipay.com</seller_account_name><out_trade_no>cs201306140001</out_trade_no><subject>测试</subject><total_fee>0.01</total_fee></direct_trade_create_req>",
    "req_id" => "201306140001",
    "sec_id" => "0001",
    "service" => "alipay.wap.trade.create.direct",
    "v" => "2.0",
```



```
"sign"      =>
"EGB8F+aWwBhotwNSQwF9Sik2OzQffV1Val1STvrRj4MzE1W4pLM6Mfjk3/4Vj3ci4Pspcm1O
LDHPZ2JQNsFruS6d4bFSQrtEF52irwSF+re42+TBTRe7w8TZAjBnBBLknerq0DvN86SfA6Qa
XOl/nOfgcMfCx3YtJnX9u2Yi/k="
);
```

步骤5: 模拟远程 HTTP 协议请求支付宝（调用 `AlipaySubmit` 类中的 `getHttpResponsePOST` 函数），得到支付宝的处理结果，字符串文本格式。

```
res_data=wftFwWhcSfy%2Bjowl544yEnltaSiLPbtgKcLqaLYDeJ9DFMvQXPJGO3tE%2FJzs
CQsnSv5i%2FhJHTggXJpVBhmPlgwTQiBv32UsO5cFcFagQfdxet5axkmfsRmGwIaOD2eMJ4dX
K%2FyHXkEgElmxhjgBj5xtbZoZQJssSL%2F7ppYhIlGrU3Mabt0tdKoY6l3B2t0JUqHKYq4b8
xCFx9AEDjA2lSrJ%2BZYcVlVJWZBbQPJmWkB5RYHp9Shfj302qxPpwxSI%2FVX2jZq1AXyQMn
KBjj57xXtV5r4OZnH%2BxbXMBPnJsNnQ6NpTgjXdDU05eAo4NGiC7bHTcEZJRkJKE54oyn3jK
Nw%3D%3D&service=alipay.wap.trade.create.direct&sec_id=0001&partner=20885
01624560335&req_id=201306140001&sign=BdzmfbS2YONm7aO67zu%2FERNJalhm8zYd3T
VFCzXrsGrafc%2BS4svEk8TRFyOpLpF5SUI639j4uiOrObrSfYLR6EeVj8bMpDZhwdfqgr408
CFctZN6dvbbmBPY7LpXbP8mLplQNfJQTeagbGAA49X33cibM376yqLLsogAx%2FowTs4%3D&v
=2.0
```

步骤6: 对该结果做 URLDECODE 解码工作（调用 `urldecode` 函数）。

```
res_data=wftFwWhcSfy+jowl544yEnltaSiLPbtgKcLqaLYDeJ9DFMvQXPJGO3tE/JzsCQsn
Sv5i/hJHTggXJpVBhmPlgwTQiBv32UsO5cFcFagQfdxet5axkmfsRmGwIaOD2eMJ4dXK/yHXk
EgElmxhjgBj5xtbZoZQJssSL/7ppYhIlGrU3Mabt0tdKoY6l3B2t0JUqHKYq4b8xCFx9AEDjA
2lSrJ+ZYcVlVJWZBbQPJmWkB5RYHp9Shfj302qxPpwxSI/VX2jZq1AXyQMnKBjj57xXtV5r4O
ZnH+xbXMBPnJsNnQ6NpTgjXdDU05eAo4NGiC7bHTcEZJRkJKE54oyn3jKNw==&service=ali
pay.wap.trade.create.direct&sec_id=0001&partner=2088501624560335&req_id=2
01306140001&sign=BdzmfbS2YONm7aO67zu/ERNJalhm8zYd3TVFCzXrsGrafc+S4svEk8TR
FyOpLpF5SUI639j4uiOrObrSfYLR6EeVj8bMpDZhwdfqgr408CFctZN6dvbbmBPY7LpXbP8mL
plQNfJQTeagbGAA49X33cibM376yqLLsogAx/owTs4=&v=2.0
```

步骤7: 对这串结果字符串做字符串切割，获取返回参数的信息。其中，如果签名方式设置的是 0001（RSA），那么还需要对这结果中的返回参数 `res_data` 做解密工作；如果签名方式设置的是 MD5，则结果中的返回参数 `res_data` 直接显示明文。

```
res_data= <?xml version="1.0"
encoding="utf-8"?><direct_trade_create_res><request_token>20130614039a363
773d04690f4196e888bf7blff</request_token></direct_trade_create_res>&servi
ce=alipay.wap.trade.create.direct&sec_id=0001&partner=2088501624560335&re
q_id=201306140001&sign=BdzmfbS2YONm7aO67zu/ERNJalhm8zYd3TVFCzXrsGrafc+S4s
vEk8TRFyOpLpF5SUI639j4uiOrObrSfYLR6EeVj8bMpDZhwdfqgr408CFctZN6dvbbmBPY7Lp
XbP8mLplQNfJQTeagbGAA49X33cibM376yqLLsogAx/owTs4=&v=2.0
```

接着把 XML 结构的 `res_data` 的值做 XML 解析，从中获取子节点 `request_token` 的值（这一系列过程需调用 `AlipaySubmit` 类中的 `parseResponse` 函数），这个值就是授权令牌。

```
20130614039a363773d04690f4196e888bf7b1fff
```

步骤8: 商户为手机网页即时到账交易接口(`alipay.wap.auth.authAndExecute`)的请求参数配置好数据后，与步骤一、二、三、四同理，把数据组装进数组 `$parameter` 中。

```
$parameter = array(
    "service" => "alipay.wap.auth.authAndExecute",
    "partner" => "2088501624560335",
    "sec_id" => "0001",
    "format" => "xml",
    "v" => "2.0",
    "req_id" => "201306140001",
    "req_data" =>
"<auth_and_execute_req><request_token>20130614039a363773d04690f4196e888bf7b1fff</request_token></auth_and_execute_req>",
    "_input_charset" => "utf-8"
);
```

对该数组做空值过滤（调用 `AlipaySubmit` 类中的 `paraFilter` 函数），对该数组做参数名首字母升序的排序动作（调用 `AlipaySubmit` 类中的 `argSort` 函数），把待签名数组转变成待签名字符串，对该字符串做签名（调用 `AlipaySubmit` 类中的 `buildRequestMysign` 函数），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。

```
$parameter = array(
    "_input_charset" => "utf-8",
    "format" => "xml",
    "partner" => "2088501624560335",
    "req_data" =>
"<auth_and_execute_req><request_token>20130614039a363773d04690f4196e888bf7b1fff</request_token></auth_and_execute_req>",
    "req_id" => "201306140001",
    "sec_id" => "0001",
    "service" => "alipay.wap.auth.authAndExecute",
    "v" => "2.0",
    "sign" =>
"vtkY9FLWCzJSIpxiNicV85FAGSTacoV7X5k2zLLPAMfMJZPlk6FpJ2/s94g0R0AcEHZyvOp8D4qhR/p1TMRuqw5Hr86LeTeyR52d9VK1NDKrBRSzw6rXG8R6LqUJkqv1i095z1a5VeJW2G46MjQkDAY+AIDWbUx1XCSAaiyx4X4="
);
```

步骤9: 以 GET 方式, 请求支付宝(调用 AlipaySubmit 类中的 buildRequestForm 函数), 此时商户的页面会自动跳转至支付宝收银台。



图5-3 跳转至支付宝收银台

后面的动作全由买家在支付宝收银台中操作完成。

步骤10: 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,并提示用户可返回商户网站。买家点击该功能按钮后,则当前界面会跳转回商户设置的 `call_back_url` 页面文件去,如果长时间没有点击,则当前支付宝收银台界面会自动跳转至商户 `call_back_url` 页面文件去。此时商户的 `call_back_url` 页面会收到一串带有处理结果数据的链接地址。

```
http://127.0.0.1:8800/WS_WAP_PAYWAP-PHP-UTF-8/call_back_url.php?out_trade_no=cs201306140001&request_token=requestToken&result=success&trade_no=2013061435574635&sign=UmQGu7GQ6lQuisaK9FhQj11UO3K%2BEt%2BXDsbdnuHuex38I1ssnsrTlADpCFHWfeezZs4%2FW5knSMo%2Fmg2DheSRSLj4%2FOTJRUTCnAJmiwmeK%2Bj9icN8vz5pcAXlXiFrwb9oTj8gLC0eY3GLXRmCVHTUW9UD89ZI9IraxsxCHgXQZaQ%3D&sign_type=0001
```

步骤11: 用 GET 方式获得这些返回数据,与步骤一、二、三、四同理,对数组 `$_GET` 做空值过滤、参数名首字母升序的排序、待签名数组转变成待签名字符串。

```
out_trade_no=cs201306140001&request_token=requestToken&result=success&trade_no=2013061435574635
```

对该字符串做签名验证(这一系列动作为调用 `AlipayNotify` 类中的 `verifyReturn` 函数完成),得到布尔类型的验证结果,完成 `call_back_url` 页面文件的验证动作。

步骤12: 买家对一笔交易付款完成的同时,除了支付宝会触发页面跳转至 `call_back_url` 文件的同步通知动作,还会自动触发服务器异步通知页面文件(商户设置参数 `notify_url` 的值)。此时,商户的 `notify_url` 文件也会收到支付宝异步通知回来的数据。

```
{service=alipay.wap.trade.create.direct,sign=MK1Q0vAATXDJTiBOxCh3XLulQDJgvYD7tU3doIpYuZef/JCmi6MubmanJbJ9TdSbGK3iM7anDxwdf1M9DVaPuc7ZdqS0Xp4N+YGUXz1cm3a121cAMjh7ttAsPy4cia4kKWJo0ATZZeIP8jkZG2L46nBeElqo3ZcbeMQklFhu6ak=,sec_id=0001,v=1.0,notify_data=wd2ALy4RTeR2xvoSUSphr98A2Ctm2rbOvhG/T8+U2vD0xTTWhf5sGsmRV+s33SLZIZQ/dJyUxIvmkrvpPDWsT1T+qRWc6MVT32Na/5JdQyyE89e6bhJHmQk+OjyCF68XdsCDA0cw8QtA5ScFfeIcV6ryXCw6luwtgBEebMDL0d8j9IWUYs1nnLdWlHNfRo8AUH6hEQpbz3pboMxVLZcXcFTKReblNuIfe5Df0x9tfeUlpnN+JewkUFSqQP9jMZD7PWjfx+Lm11st9ftHfhzZ+w/s116i1ow8MONEkwStQkT2yVBGMcpLw0IZeS/LZthleyM2oUfDMOAlaK7zeuP8ICdU2TjMNOahkBipBdgLpJQgaAW/xa05LKirg6Au7IRtHeKBf1haXcQ3LNkgc3HttbmATjJaMQM535CGaaXK7zFszHZCB3kluk3I1PJ2tVX2Uy+xkSZP/h20/IxH1Sdj8xXDe74g2XYtBbsVd8Qmd7gf8YNwQk+WYtMsAbHCzugE0MvxTwyyG1XoGyCRHYe7u2oudsQGzbrAyrGuMHpvn3Ku09up2sjcn0h5zAOUDCSlcJwCa2qYdly4KOZqtXqiPFUQ2yRuUMLuvWQC0r0ckAGlVhUV93oMu1CXU8PdQTPNGZPpb7AEDyCYWhpt/CWlz0def7o1GhWbC19Z3NMdVfDvoyaL//77cUjjqKVYQ9EeDdRnKujj0AzSV8gLG4rmmf1zHiYOpfXflyghxWc30/i8t1F7ruKbS7pSvcyBvzgmW0zojZnattdoKiIRNZAEZ+v4cqDKkBPmTlrKJc7yL7nYOEdXiTaVQ/3iFpZm0J+U3pOewsYbX6Vmkbv5wTYHJzNqFu6E1QyYjiibRSospNgRRlzhXXAUg7L7nK0JR8PKhen4ID1ZVRtL+r0RzGnPiMvSyt4fAbkP06nKikuZ
```

```
BeTD9isLY793XG6lj4PCTAHWZE4DzZAXoEppAdCqd810kpav9OrbShFpEhKvDcsDxTRixwjTx
saTv+N4DOBYuJThoR6SRlqmiQNArmGx3glNUbZdYQ4i5zdLU5B8v02idAwRFWEbH8oBzyKx1w
s1OeEwq0+WisDvxJCAtpPWeZvWL0v/u6MOPkAFoN9Ml4dLlOFcrSaifblfaCVXajS7QjcvS2Y
LuLzQUjjb2R0H0ZnSCVjS+FgiXhTWrEr/8VB8VsMcOZ6Vz+ZrcT5wjWpP/I119q8g0xB260B4
x50lxocMjTertociASRirLkg2jYH4EcUMN/fFZRLTVJuFZQ6rVLpsK+eHzgyIrs2Fz/YZORdS
et6zaFzERzEhrWSrL6mJ878v14b92CGoy6op7m7B/GBsrjEwizCPR039ZaDsMahg==}
```

步骤13: 用 POST 方式获得这些返回数据，如果签名方式设置的是 0001（RSA）时，需要先对这些数据中的参数 `notify_data` 做解密；如果签名方式设置的是 MD5，因为直接返回明文，所以不需要解密。

```
{service=alipay.wap.trade.create.direct,
sign=MK1Q0vAATXdjTiBOxCh3XLulQDJgvYD7tU3doIpYuZef/JCMi6MubmanJbJ9TdSbGK3i
M7anDxwdf1M9DVaPuc7ZdqS0Xp4N+YGUXz1cm3a12lcAMjh7ttAsPy4cia4kKWJo0ATZzeIP8
jkZG2L46nBeElqo3ZcbemQklFhu6ak=, sec_id=0001, v=1.0,
notify_data=<notify><payment_type>1</payment_type><subject>测试
</subject><trade_no>2013061435574635</trade_no><buyer_email>xxxxxxx@xx.co
m</buyer_email><gmt_create>2013-06-14
17:38:45</gmt_create><notify_type>trade_status_sync</notify_type><quantit
y>1</quantity><out_trade_no>cs201306140001</out_trade_no><notify_time>201
3-06-14
17:38:54</notify_time><seller_id>2088501624560335</seller_id><trade_statu
s>TRADE_FINISHED</trade_status><is_total_fee_adjust>N</is_total_fee_adjus
t><total_fee>0.01</total_fee><gmt_payment>2013-06-14
17:38:54</gmt_payment><seller_email>alipayrisk10@alipay.com</seller_email
><gmt_close>2013-06-14
17:38:54</gmt_close><price>0.01</price><buyer_id>2088002396712354</buyer_
id><notify_id>be3e931aa1cc2147d7fa5b6bf569b6153y</notify_id><use_coupon>N
</use_coupon></notify>}
```

`notify_url` 异步通知的签名验证规律与 `call_back_url` 不同，商户只需要顺序获取支付宝的返回数据，不需要对这些参数排序。因此，直接把数组 `$_POST` 转变成待签名字符串。

```
service=alipay.wap.trade.create.direct&v=1.0&sec_id=0001&notify_data=<not
ify><payment_type>1</payment_type><subject>测试
</subject><trade_no>2013061435574635</trade_no><buyer_email>xxxxxxx@xx.co
m</buyer_email><gmt_create>2013-06-14
17:38:45</gmt_create><notify_type>trade_status_sync</notify_type><quantit
y>1</quantity><out_trade_no>cs201306140001</out_trade_no><notify_time>201
3-06-14
17:38:54</notify_time><seller_id>2088501624560335</seller_id><trade_statu
s>TRADE_FINISHED</trade_status><is_total_fee_adjust>N</is_total_fee_adjus
t><total_fee>0.01</total_fee><gmt_payment>2013-06-14
17:38:54</gmt_payment><seller_email>alipayrisk10@alipay.com</seller_email
```

```
><gmt_close>2013-06-14
17:38:54</gmt_close><price>0.01</price><buyer_id>2088002396712354</buyer_
id><notify_id>be3e931aalcc2147d7fa5b6bf569b6153y</notify_id><use_coupon>N
</use_coupon></notify>
```

接着对该字符串做签名验证（这一系列动作调用 `AlipayNotify` 类中的 `verifyNotify` 函数），得到布尔类型的验证结果，完成 `notify_url` 页面文件的验证动作。

步骤14: 在 `notify_url` 页面文件中，参数 `notify_data` 是 XML 结构的字符串，需要先 XML 解析，再获取需要的子节点数据。

3. ASP.NET(C#)

步骤1: 商户为手机网页即时到账授权接口(`alipay.wap.trade.create.direct`)的请求参数配置好数据后，组装进数组 `sParaTempToken` 中，再对该数组做空值过滤（调用 `AlipayCore` 类中的 `FilterPara` 函数）。

```
{[partner, 2088101568358171]}
{[_input_charset, utf-8]}
{[sec_id, 0001]}
{[service, alipay.wap.trade.create.direct]}
{[format, xml]}
{[v, 2.0]}
{[req_id, 201306170002]}
{[req_data,
<direct_trade_create_req><notify_url>http://www.xxx.com/WS_WAP_PAYWAP-CSH
ARP/notify_url.aspx</notify_url><call_back_url>http://127.0.0.1:48034/WS_
WAP_PAYWAP-CSHARP/call_back_url.aspx</call_back_url><seller_account_name>
alipayrisk10@alipay.com</seller_account_name><out_trade_no>cs201306170002
</out_trade_no><subject>测试
</subject><total_fee>0.01</total_fee></direct_trade_create_req>]}
}
```

步骤2: 对该数组做参数名首字母升序的排序动作（调用 `AlipayCore` 类中的 `SortPara` 函数），排序规则请参见“3 技术接入规则”中的“请求时签名逻辑”，此时得到的数组为待签名数组。

```
{[_input_charset, utf-8]}
{[format, xml]}
{[partner, 2088101568358171]}
{[req_data,
<direct_trade_create_req><notify_url>http://www.xxx.com/WS_WAP_PAYWAP-CSH
ARP/notify_url.aspx</notify_url><call_back_url>http://127.0.0.1:48034/WS_
WAP_PAYWAP-CSHARP/call_back_url.aspx</call_back_url><seller_account_name>
alipayrisk10@alipay.com</seller_account_name><out_trade_no>cs201306170002
</out_trade_no><subject>测试
}
```

```
</subject><total_fee>0.01</total_fee></direct_trade_create_req>]]}
{[req_id, 201306170002]}
{[sec_id, 0001]}
{[service, alipay.wap.trade.create.direct]}
{[v, 2.0]}
```

步骤3: 把待签名数组转变成待签名字符串。

```
_input_charset=utf-8&format=xml&partner=2088101568358171&req_data=<direct
_trade_create_req><notify_url>http://www.xxx.com/WS_WAP_PAYWAP-CSHARP/not
ify_url.aspx</notify_url><call_back_url>http://127.0.0.1:48034/WS_WAP_PAY
WAP-CSHARP/call_back_url.aspx</call_back_url><seller_account_name>alipayr
isk10@alipay.com</seller_account_name><out_trade_no>cs201306170002</out_t
rade_no><subject>测试
</subject><total_fee>0.01</total_fee></direct_trade_create_req>&req_id=20
1306170002&sec_id=0001&service=alipay.wap.trade.create.direct&v=2.0
```

并对该字符串做签名（调用AlipaySubmit类中的BuildRequestMysign函数），转变规则请参见“3 技术接入规则”中的“请求时签名逻辑”。

```
xO/eAP4pFusEveZmaw2mgCEoQGxDqkFZ+AF8wI1U52OGASS480/efTZLn9Cr0tF9tDJQ+hwlc
Nj+F4/Hx+w6v35c0oE3F1Eab/Z2wKAirIYVh/HagumS34aa6t7lJb21YFIshulYgoqWfwakQ/
fzdcwVncW7jJQIvMsIgAhEJ8I=
```

步骤4: 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。

```
{[_input_charset, utf-8]}
{[format, xml]}
{[partner, 2088101568358171]}
{[req_data,
<direct_trade_create_req><notify_url>http://www.xxx.com/WS_WAP_PAYWAP-CSH
ARP/notify_url.aspx</notify_url><call_back_url>http://127.0.0.1:48034/WS_
WAP_PAYWAP-CSHARP/call_back_url.aspx</call_back_url><seller_account_name>
alipayrisk10@alipay.com</seller_account_name><out_trade_no>cs201306170002
</out_trade_no><subject>测试
</subject><total_fee>0.01</total_fee></direct_trade_create_req>]]}
{[req_id, 201306170002]}
{[sec_id, 0001]}
{[service, alipay.wap.trade.create.direct]}
{[v, 2.0]}
{[sign,
xO/eAP4pFusEveZmaw2mgCEoQGxDqkFZ+AF8wI1U52OGASS480/efTZLn9Cr0tF9tDJQ+hwlc
Nj+F4/Hx+w6v35c0oE3F1Eab/Z2wKAirIYVh/HagumS34aa6t7lJb21YFIshulYgoqWfwakQ/
fzdcwVncW7jJQIvMsIgAhEJ8I=]}
```


步骤5: 模拟远程 HTTP 协议请求支付宝（调用 AlipaySubmit 类中的 BuildRequest(string GATEWAY_NEW, Dictionary<string, string> sParaTemp)函数），得到支付宝的处理结果，字符串文本格式。

```
res_data=KzTV7n3JHKMmaxc9jObz8FqT9PO8MQw5kqX8uqO4fGwvNIDHNngLdEig%2FaN4A97i0nRviLzoutVSojBIM%2BRECEv5UR6aqhME%2Bei%2BX8LYf9sZaE1V3o2oMz3BnubFOcnozWBgXyDaA2fzeU1YxT9TL4BLADEJ0gZB5upjpMeiyO%2Bil9068g111ojNbJE3jeGtgqp%2FXTSXmFZ0x7DYefB1XYWmzETLywMkKdwlhnYIDrTLHLj4wdMTmsQS%2BxbTD2egERafRFD3DtPDK06j7pRfHcJBtoEtC%2Fg4Zj2FD58ADEGLKb7oAkMU6ado0911a%2FvAzH3gZtElHAS%2BdYglI%2FUIBw%3D%3D&service=alipay.wap.trade.create.direct&sec_id=0001&partner=2088101568358171&req_id=201306170002&sign=oo4ShGclfndnYiI20JzclNSQcfspEzoQchrYS8fhiaAzddE3lra%2Ffq7xOm9RCzDTfgyO3Sl8pUcAWY%2B4BYPrxMF5GPVv2IZDY%2FN%2FqjTwEPrkQYazOtimzB1qRTA%2FUrOrEejEyMUqN2XXrMw14%2FavdFjPpBAsCsF7KFJOL%2FB7H4%3D&v=2.0
```

步骤6: 对该结果做 URLDECODE 解码工作（调用 urldecode 函数）。

```
res_data=KzTV7n3JHKMmaxc9jObz8FqT9PO8MQw5kqX8uqO4fGwvNIDHNngLdEig/aN4A97i0nRviLzoutVSojBIM+RECEv5UR6aqhME+ei+X8LYf9sZaE1V3o2oMz3BnubFOcnozWBgXyDaA2fzeU1YxT9TL4BLADEJ0gZB5upjpMeiyO+il9068g111ojNbJE3jeGtgqp/XTSXmFZ0x7DYefB1XYWmzETLywMkKdwlhnYIDrTLHLj4wdMTmsQS+xbTD2egERafRFD3DtPDK06j7pRfHcJBtoEtC/g4Zj2FD58ADEGLKb7oAkMU6ado0911a/vAzH3gZtElHAS+dYglI/UIBw==&service=alipay.wap.trade.create.direct&sec_id=0001&partner=2088101568358171&req_id=201306170002&sign=oo4ShGclfndnYiI20JzclNSQcfspEzoQchrYS8fhiaAzddE3lra/fq7xOm9RCzDTfgyO3Sl8pUcAWY+4BYPrxMF5GPVv2IZDY/N/xQjTwEPrkQYazOtimzB1qRTA/UrOrEejEyMUqN2XXrMw14/avdFjPpBAsCsF7KFJOL/B7H4=&v=2.0
```

步骤7: 对这串结果字符串做字符串切割，获取返回参数的信息。其中，如果签名方式设置的是 0001（RSA），那么还需要对这结果中的返回参数 res_data 做解密工作；如果签名方式设置的 MD5，则结果中的返回参数 res_data 直接显示明文。

```
res_data=<?xml version="1.0"
encoding="utf-8"?><direct_trade_create_res><request_token>201306179d4c458a00e26aca863b4dac3cc2bfd3</request_token></direct_trade_create_res>&service=alipay.wap.trade.create.direct&sec_id=0001&partner=2088101568358171&req_id=201306170002&sign=oo4ShGclfndnYiI20JzclNSQcfspEzoQchrYS8fhiaAzddE3lra/fq7xOm9RCzDTfgyO3Sl8pUcAWY+4BYPrxMF5GPVv2IZDY/N/xQjTwEPrkQYazOtimzB1qRTA/UrOrEejEyMUqN2XXrMw14/avdFjPpBAsCsF7KFJOL/B7H4=&v=2.0
```

接着把 XML 结构的 res_data 的值做 XML 解析，从中获取子节点 request_token 的值（这一系列过程需调用 AlipaySubmit 类中的 ParseResponse 函数），这个值就是授权令牌。

```
201306179d4c458a00e26aca863b4dac3cc2bfd3
```


步骤8: 商户为手机网页即时到账交易接口(alipay.wap.auth.authAndExecute)的请求参数配置好数据后, 与步骤一、二、三、四同理, 把数据组装进数组 sParaTemp 中。

```
{[partner, 2088101568358171]}
{[_input_charset, utf-8]}
{[sec_id, 0001]}
{[service, alipay.wap.auth.authAndExecute]}
{[format, xml]}
{[v, 2.0]}
{[req_data,
<auth_and_execute_req><request_token>201306179d4c458a00e26aca863b4dac3cc2
bfd3</request_token></auth_and_execute_req>]}
```

对该数组做空值过滤(调用 AlipaySubmit 类中的 FilterPara 函数), 对该数组做参数名首字母升序的排序动作(调用 AlipaySubmit 类中的 SortPara 函数), 把待签名数组转变成待签名字符串, 对该字符串做签名(调用 AlipaySubmit 类中的 BuildRequestMysign 函数), 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。

```
{[_input_charset, utf-8]}
{[format, xml]}
{[partner, 2088101568358171]}
{[req_data,
<auth_and_execute_req><request_token>201306179d4c458a00e26aca863b4dac3cc2
bfd3</request_token></auth_and_execute_req>]}
{[sec_id, 0001]}
{[service, alipay.wap.auth.authAndExecute]}
{[v, 2.0]}
{[sign,
yB+DrSNb0zvnpgzgizd7h/J45V0xbNsOSr7q0/qpX4jElmSCDxUl6ouwc3T39XwCylnwil/SGx
0PiwjN1AKHy+4zADjQVmDMCR28b9skLrVM/Gw5cU35FEgv1TIK+UZ85t/5m4KVby9CIqeI+B6
ziXM4zQUxsNTU7AjRGi/kU3Es=]}
```

步骤9: 以 GET 方式, 请求支付宝(调用 AlipaySubmit 类中的 BuildRequest(string GATEWAY_NEW, Dictionary<string, string> sParaTemp, string strMethod, string strButtonValue)函数), 此时商户的页面会自动跳转至支付宝收银台。



图5-4 跳转至支付宝收银台

后面的动作全由买家在支付宝收银台中操作完成。

步骤10: 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,并提示用户可返回商户网站。买家点击该功能按钮后,则当前界面会跳转回商户设置的 `call_back_url` 页面文件去,如果长时间没有点击,则当前支付宝收银台界面会自动跳转至商户 `call_back_url` 页面文件去。此时商户的 `call_back_url` 页面会收到一串带有处理结果数据的链接地址。

```
http://127.0.0.1:48034/WS_WAP_PAYWAP-CSHARP/call_back_url.aspx?out_trade_no=cs201306170002&request_token=requestToken&result=success&trade_no=2013061742035035&sign=vnXVWq91w87tmKdrtePYKnEJZmVhuUMeVCVv18HwU8ABmvKH48CLXRQjjAdU4y75WtR62HhEoNCLAnTkx1hxsVGOZwpSRx9DxdXwz1m1NozIOV%2BTN8V91%2FQXEJj0dmlgd3UjHdxAbMkFKFDznIfZwYSbhCAL2H4KwXdefLAGiw%3D&sign_type=0001
```

步骤11: 用 GET 方式获得这些返回数据，与步骤一、二、三、四同理，对数组 sPara（调用 call_back_url 页面中的 GetRequestGet()函数）做空值过滤、参数名首字母升序的排序、待签名数组转变成待签名字符串。

```
out_trade_no=cs201306170002&request_token=requestToken&result=success&trade_no=2013061742035035
```

对该字符串做签名验证（这一系列动作作为调用 AlipayNotify 类中的 VerifyReturn 函数完成），得到布尔类型的验证结果，完成 call_back_url 页面文件的验证动作。

步骤12: 买家对一笔交易付款完成的同时，除了支付宝会触发页面跳转至 call_back_url 文件的同步通知动作，还会自动触发服务器异步通知页面文件（商户设置参数 notify_url 的值）。此时，商户的 notify_url 文件也会收到支付宝异步通知回来的数据。

```
{service=alipay.wap.trade.create.direct,
sign=eUHFymt6/MSGNDX8XcnD44IRKEXkx3mG5dk9hp39LYqmwO/WWf9+mldyljSkEXT09wA2p8Tv9CR1NCTy9nMjHg19KWX8Rtrz2CifoOvTsE8ttxaRzkQ8TBo/K3sHPERJmQZE2uW8RyXywMaRbRMC+90YATDqex6e7FCR0HQ4KI=, sec_id=0001, v=1.0,
notify_data=iUtYE4ipOb+hpmw+1UllDaLvTxusiNwV6LKDn6vSJLY5Pl2HaE909sqWfYXeh+oos6cOS3bB5fkmDqJeDUXHH5oJFttkIosZAJ1W0QjROcpF3/j4fx+15DsbGMwXfF4qeBAGA2PdAlwpdZK+krJeAyNn7jNUc2STWVS7ZBRBs0SV8meE1TrjQf5zcQMmoBhLRot6/9iizyoeBYh2s4JA2u+1HakCLEFproN288teBwA6X/apKe+AauHCYnrUMSO8Wjhz7vaQlkWkQ7tq5rcT6kPGkmwhVvhVylDYXB51SD0KI/17qHm4z8hWiXzArl22Plr6TwP87WBmnAudPfAM5Jj0Uq1UeCE2Lpg+8Z2IzKPe10TLJhtcrjza3q/vFj/J1lTlpq2PM9eDs9p0aV3KuCH3+1Ms7oMqQ8IjeJdEXYFIa002NxCTkc7VWyMnwHWYCSGC+vPUR9KeZKtWvMpHpJRIJegXiF2LWRU0XrXfwnguiOpWVMSW3XGemedxRsIG0AOyNSLafq+9Wqvcbz6lczwskDcLwwOho2Qo9sRWB7wEytCjWgstXHbXLXR3GXvIkogD7slYd3mvqsczeTPCJ7KeftQP0yGi6gb3wM7igDkIc2tvvdrwQqg7SBj0OU1mUwZBMXYb4E9rlFNELHw+5/Tu/qlQcaSma/WWiOL1z0yjlHnfF/PhdWJ/ohlDCMg5iwPW4XalgJx+SEGz/lr9ZFbSvCIzzVFgXN+v+26kNb/VvM4dqfSUoj722AOv0fvkNA7qS0EZ7j0gv2L5IZMjibK7DwQ/SLgiQf9sALPWkSBoJfBdZ4k5vJXry4p3Uh2BHUEdckQzNaB3oQJVT8X+6h+8hpbvZS5KpDRrq57odJEWqcWQ3tEozyANqL2XTea3tH3h9Fi3ZqmhNTazvdRMGXHLAOhMig6kmo84aYbPMapv/5fyQdXrELZLiuZh9oxLTECqrHN1kx8BLSLp4ubpkent7NV2LgJbA+jZhE7pUH2vnND4b6XNpOOSha+3y6/kLCjj6cnxUhnwjuUdCCfzXTK1j1+RCS1+G89SR+f7q1NQy/LPK4qD/FVAmquCFMSDzwKNPfnJdxadiElAezivEP4q7ihjH2iNkm/7WJouYfWfYTYXAepa5OIzodJP1otQuluheDmm1COKnUw9PW6rOQkNK41Weiej0fWqTLi19Fe9YbWagVXf4qJ74RhXLhoWso2JdQV3OPNjCXa5EXn8mRGsB8yQKtwZvZdTgtjMB+BMcEa6Qqb74G3mFifyB1+ZCdGIBiJk0eMPzyskyqdQG15wC152F5A2BnImIObQ4MgFFqY0gbjM1xzUcb/mhQ196sPP+Mv+25uw4CcJE+5FQW6ONx343MEcs8P+esaRly2CH4uXiUBtjltzlcD3kpC4RobNzuCmb/NLROkx+mLKZJeID1SgBa1Shj7OzYqVY
```

```
yyUEpdPfgLEgwFsCtBl3Tsf05yWl9acFX2+fYQXk+/k5gs/pkCt2ol27Z4bcGDEOP4zPCd5Kr
HyvguyUDXWepc48}
```

步骤13: 用 POST 方式获得这些返回数据, 如果签名方式设置的是 0001 (RSA) 时, 需要先对这些数据中的参数 `notify_data` 做解密; 如果签名方式设置的是 MD5, 因为直接返回明文, 所以不需要解密。

```
{service=alipay.wap.trade.create.direct,
sign=eUHFymt6/MSGNDX8XcnD44IRKEXkx3mG5dk9hp39LYqmwO/WWf9+mldyljSkEXT09wA2
p8Tv9CR1NCTy9nMjHg19KWX8Rtrz2CifoOvTsE8ttxaRzkQ8TBo/K3SHPERJmQZE2uW8RyXyw
MaRbRMC+90YATDqex6e7FCR0HQ4KI=, sec_id=0001, v=1.0,
notify_data=<notify><partner>2088101568358171</partner><payment_type>8</p
ayment_type><subject>测试
</subject><trade_no>2013061742035035</trade_no><buyer_email>xxxxxxx@xx.co
m</buyer_email><gmt_create>2013-06-17
20:12:35</gmt_create><notify_type>trade_status_sync</notify_type><quantit
y>1</quantity><out_trade_no>cs201306170002</out_trade_no><notify_time>201
3-06-17
20:12:40</notify_time><seller_id>2088501624560335</seller_id><out_channel
_type>BALANCE</out_channel_type><trade_status>TRADE_SUCCESS</trade_status
><is_total_fee_adjust>N</is_total_fee_adjust><total_fee>0.01</total_fee><
gmt_payment>2013-06-17
20:12:40</gmt_payment><seller_email>alipayrisk10@alipay.com</seller_email
><price>0.01</price><buyer_id>2088002396712354</buyer_id><out_channel_amo
unt>0.01</out_channel_amount><notify_id>c6936b49aa4422d63bf14bc507c535983
y</notify_id><use_coupon>N</use_coupon></notify>}
```

`notify_url` 异步通知的签名验证规律与 `call_back_url` 不同, 商户只需要顺序获取支付宝的返回数据, 不需要对这些参数排序。因此, 直接把数组 `sPara` (调用 `notify_url` 页面中的 `GetRequestPost()` 函数) 转变成待签名字符串。

```
service=alipay.wap.trade.create.direct&v=1.0&sec_id=0001&notify_data=<not
ify><partner>2088101568358171</partner><payment_type>8</payment_type><sub
ject>测试
</subject><trade_no>2013061742035035</trade_no><buyer_email>xxxxxxx@xx.co
m</buyer_email><gmt_create>2013-06-17
20:12:35</gmt_create><notify_type>trade_status_sync</notify_type><quantit
y>1</quantity><out_trade_no>cs201306170002</out_trade_no><notify_time>201
3-06-17
20:12:40</notify_time><seller_id>2088501624560335</seller_id><out_channel
_type>BALANCE</out_channel_type><trade_status>TRADE_SUCCESS</trade_status
><is_total_fee_adjust>N</is_total_fee_adjust><total_fee>0.01</total_fee><
gmt_payment>2013-06-17
20:12:40</gmt_payment><seller_email>alipayrisk10@alipay.com</seller_email
><price>0.01</price><buyer_id>2088002396712354</buyer_id><out_channel_amo
```

```
unt>0.01</out_channel_amount><notify_id>c6936b49aa4422d63bf14bc507c535983
y</notify_id><use_coupon>N</use_coupon></notify>
```

接着对该字符串做签名验证（这一系列动作调用 AlipayNotify 类中的 VerifyNotify 函数），得到布尔类型的验证结果，完成 notify_url 页面文件的验证动作。

步骤14：在 notify_url 页面文件中，参数 notify_data 是 XML 结构的字符串，需要先 XML 解析，再获取需要的子节点数据。

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	<ul style="list-style-type: none"> 正常获取授权令牌 模拟手机浏览器进行付款 页面跳转同步返回 	仅仅把接口配置好，不要放在商户的正式网站项目中。
第二步： 在服务器上单独对这个接口进行调试	<ul style="list-style-type: none"> 正常获取授权令牌 模拟手机浏览器进行付款 页面跳转同步返回 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： 接口融合到网站项目中	无	把调试好的接口与商户网站项目的业务流程进行衔接和融合
第四步： 在本机对融合后的网站项目进行调试	<ul style="list-style-type: none"> 整个业务操作流程 模拟手机浏览器进行付款 页面跳转同步返回 业务后续的执行 	在本机调试衔接到网站项目后的接口
第五步： 在服务器对融合后的网站项目进行调试	<ul style="list-style-type: none"> 整个业务操作流程 使用手机浏览器进行付款 页面跳转同步返回 服务器异步通知返回 业务后续的执行 	本机调试没有问题后，再放入服务器中调试。

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: 在地址栏中输入https://mobiless.alipay.com/myproduct/my_order.htm，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 查看签约支付宝账号等信息

步骤3: 点击<密钥管理>按钮，输入“支付密码”，点击“确认”，即可查看到各类型的私钥。



图7-3 查看私钥

步骤4: 上传 RSA 类型的商户公钥，获得支付宝公钥。

[首页](#) [产品介绍](#) [成功案例](#) [我的商家服务](#)

密钥管理

密钥类型	创建时间	最后修改时间
交易安全检验码 (MD5)		
MD5密钥	2009-11-18 13:15:56	2013-03-19 14:38:26
<div>um[redacted]o</div>		
交易安全检验码 (DSA)		
上传商户公钥		
<div><input type="text"/> <input type="button" value="浏览"/> <input type="button" value="上传"/></div> <div>请上传文本格式的文件。</div>		
交易安全检验码 (RSA)		
支付宝公钥	2011-03-22 13:49:15	2013-04-16 14:41:25
<div>M[redacted]t+FXGJ7kRIKXN/DHbnU j[redacted]DlkHgCYuXEck7N t[redacted]IDAQAB</div>		
商户公钥		
<div>MIGf[redacted]6Dd/5me6ESVEKj1Soz8 GpuU[redacted]2mZo8Cgbr6sIzapGYrk1R r2A3[redacted]IDAQAB</div>		
更改商户公钥		
<div><input type="text"/> <input type="button" value="浏览"/> <input type="button" value="上传"/></div> <div>请上传文本格式的文件。</div>		

图7-4 获取支付宝公钥

步骤5: 保存支付宝账号、合作者身份 ID (PID)、交易安全校验码 (MD5)、交易安全校验码 (RSA) 中的支付宝公钥。

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：

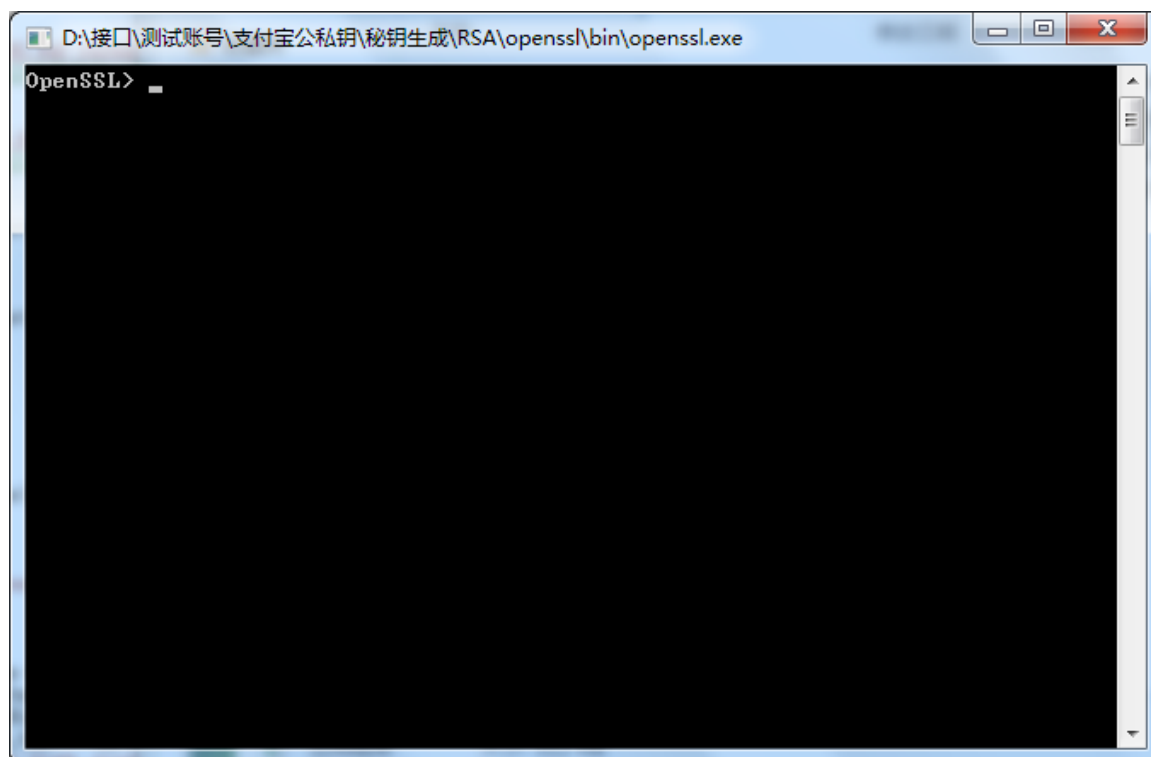


图7-5 执行 openssl.exe 文件

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：

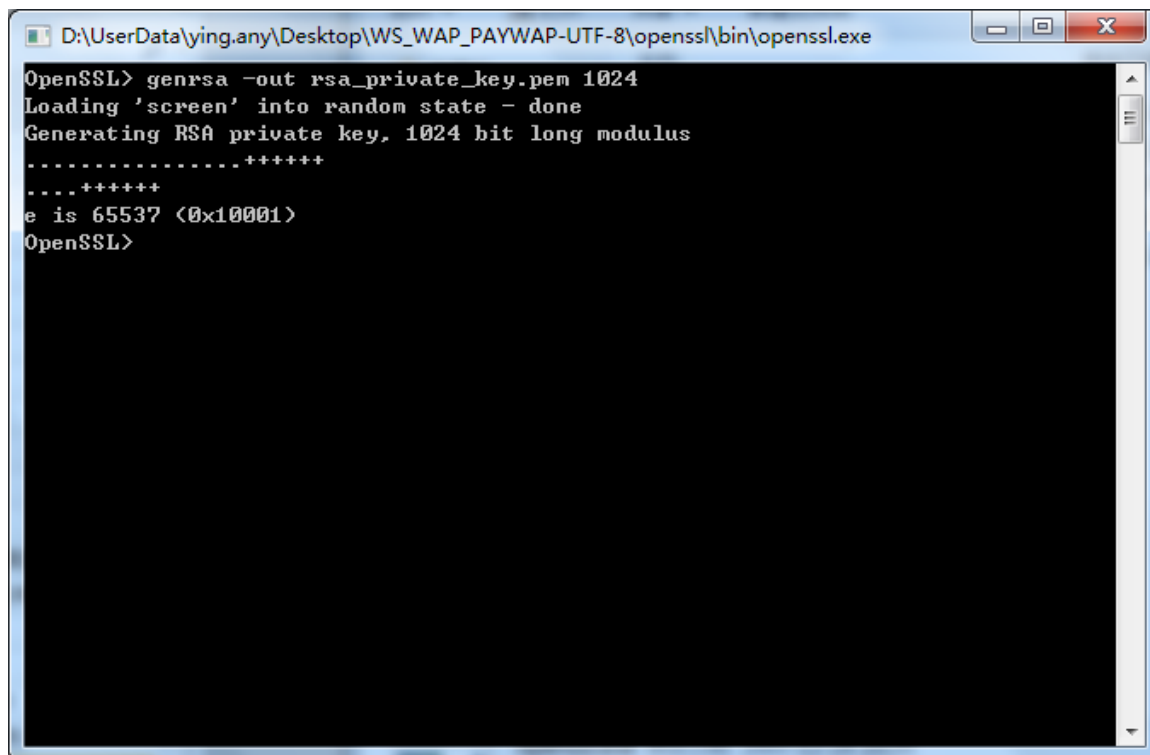
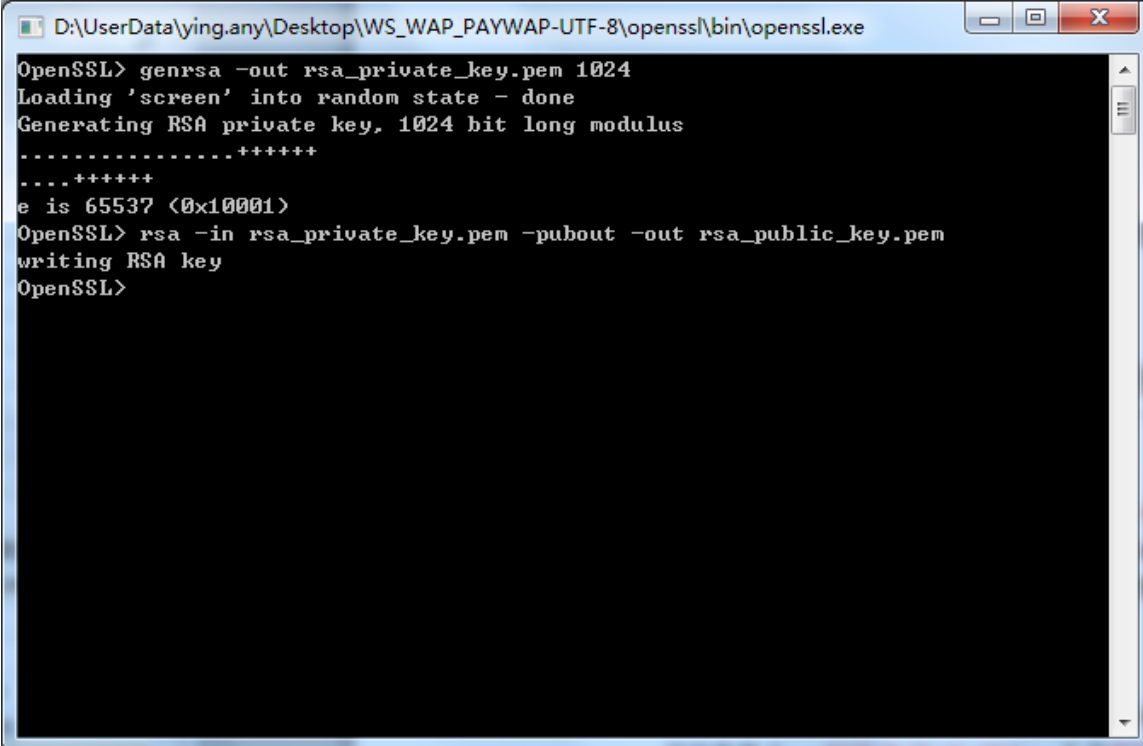


图7-6 生成 RSA 私钥

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：

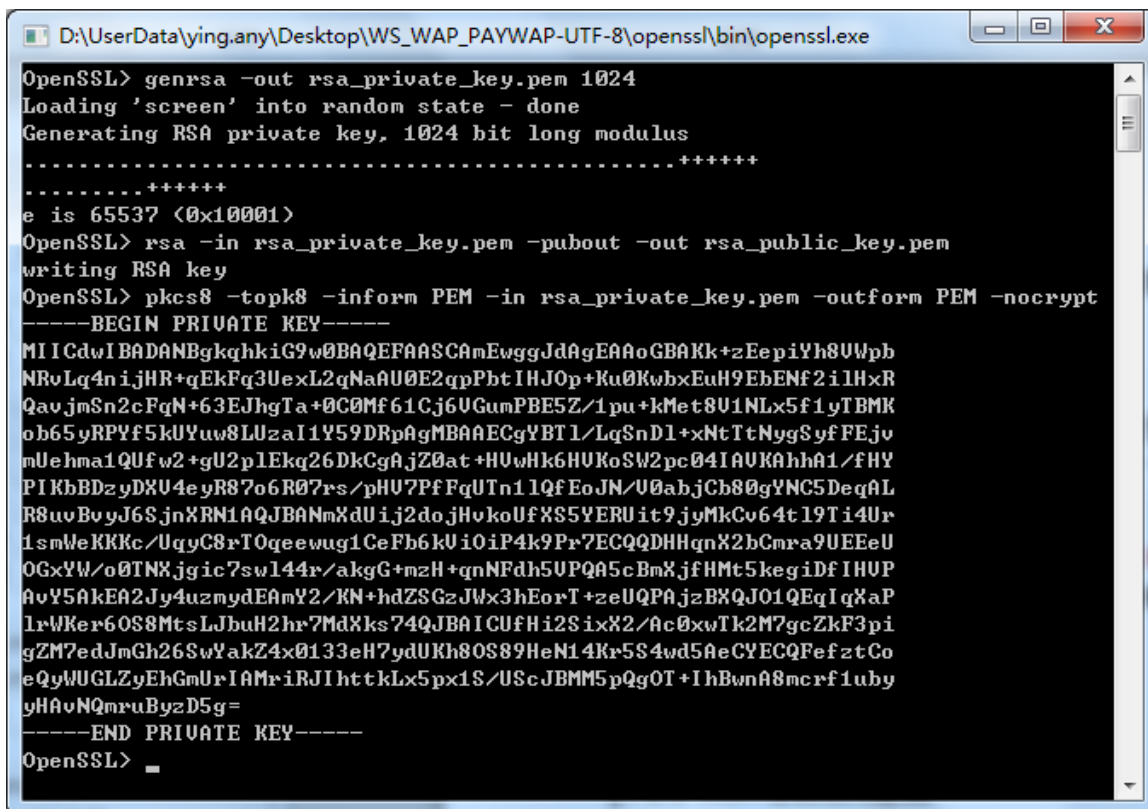


```
D:\UserData\ying.any\Desktop\WS_WAP_PAYWAP-UTF-8\openssl\bin\openssl.exe
OpenSSL> genrsa -out rsa_private_key.pem 1024
Loading 'screen' into random state - done
Generating RSA private key, 1024 bit long modulus
.....+++++
....+++++
e is 65537 (0x10001)
OpenSSL> rsa -in rsa_private_key.pem -pubout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

图7-7 生成 RSA 公钥

4. 生成PKCS8 编码的私钥

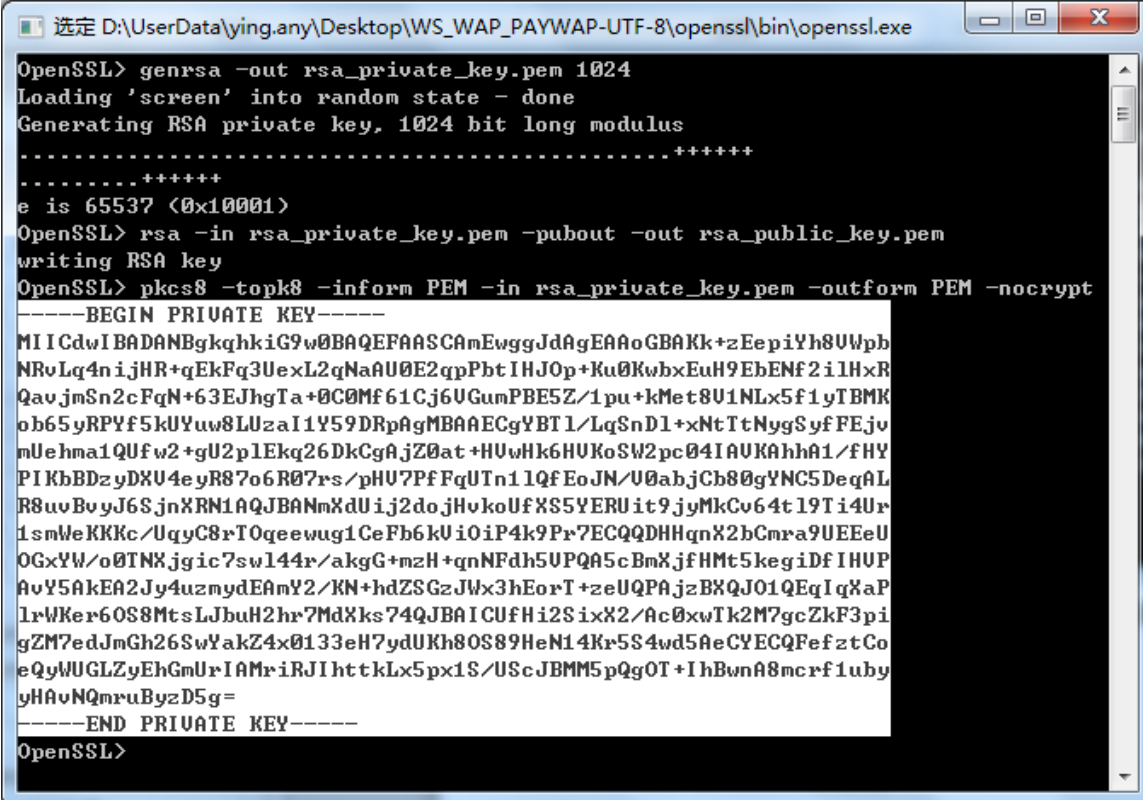
输入命令 “*pkcs8 -topk8 -inform PEM -in rsa_private_key.pem -outform PEM -nocrypt*” 并回车，当前界面中会直接显示出生成结果：



```
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
ob65yRPyf5kUYuw8LUzaI1Y59DRpAgMBAAGCgYBT1/LqSnD1+xNtTtNygsyfFEju
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
yHAvNQmruByzD5g=
-----END PRIVATE KEY-----
OpenSSL>
```

图7-8 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：



```
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-----
MIICdwIBADANBgkqhkiG9w0BAQEFAASCAmEwggJdAgEAAoGBAKk+zEepiYh8UWpb
NRvLq4nijHR+qEkFq3UexL2qNaAU0E2qpPbtIHJO+Ku0KwbxEuH9EbENf2i1HxR
QavjmSn2cFqN+63EJhgTa+0C0Mf61Cj6UGumPBE5Z/1pu+kMet8U1NLx5f1yTBMK
ob65yRPyf5kUYuw8LUza11Y59DRpAgMBAECgYBT1/LqSnD1+xNtTtNygSyfFEjv
mUehma1QUfw2+gU2p1Ekq26DkCgA jZ0at+HUwHk6HUkoSW2pc04IAUkAhhA1/fHY
PIKbBDzYdXU4eyR87o6R07rs/pHv7PfFqUTn1lQfEoJN/U0abjCb80gYNC5DeqAL
R8uvBvyJ6SjnXRN1AQJBANmXduij2dojHukoUfXSSYERUit9jyMkCv64t19Ti4Ur
1smWeKKKc/UqyC8rT0qeeuwg1CeFb6kUiOiP4k9Pr7ECQQDHHqnX2bCnra9UEEeU
OGxYW/oTNXjgic7sw144r/akg+mzH+qnNFDh5UPQA5cBmXjfhMt5kegidfIHUP
AvY5AkEA2Jy4uzmydEAmY2/KN+hdZSGzJWx3hEorT+zeUQPAjzBXQJO1QEgIqXaP
lrWker60S8MtsLJbuH2hr7MdXks74QJBAICUfHi2S ixX2/Ac0xwTk2M7gcZkF3pi
gZM7edJmGh26SwYakZ4x0133eH7ydUKh80S89HeN14Kr5S4wd5AeCYECQFefztCo
eQyWUGLZyEhGmUrIAMriRJIhttkLx5px1S/UScJBMM5pQgOT+IhBwnA8mcrf1ubY
yHAuNQmruByzD5g=
-----END PRIVATE KEY-----
OpenSSL>
```

图7-9 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
 - (1) 先用签约支付宝账号登录支付宝网站，再浏览器访问“<https://ms.alipay.com/index.htm>”，点击菜单栏“我的商家服务”，右侧点击“密钥管理”进入密钥管理页面，复制“交易安全校验码（RSA）”→“支付宝公钥”栏目下的整串信息到一个新的记事本中，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，切割后的格式与商户刚生成的公钥格式一致即可，如下图：

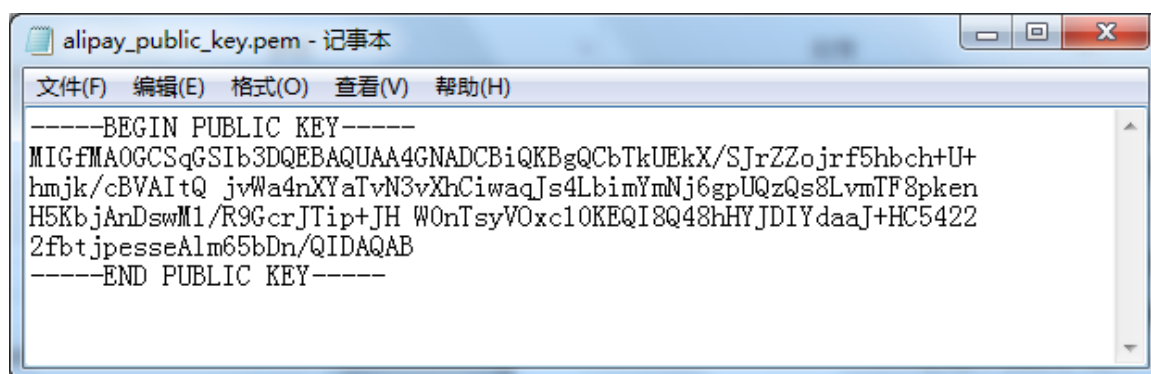


图7-10 支付宝公钥示意图

- (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

- (1) 先用签约支付宝账号登录支付宝网站，再浏览器访问
“<https://ms.alipay.com/index.htm>”，点击菜单栏“我的商家服务”，右侧
点击“密钥管理”进入密钥管理页面，复制“交易安全校验码（RSA）”→
“支付宝公钥”栏目下的整串信息；
- (2) 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。

7.4 如何增加扩展业务参数



注意：

本章节以代码示例为例，来说明如何增加扩展业务参数，仅供参考。商户需要根据自身情况来编写扩展业务参数的代码。

打开配置业务参数的页面文件，找到代码修改位置。

1. JAVA修改流程

alipayapi.jsp 文件，在参数 req_dataToken 中的根节点<direct_trade_create_req>下新增子节点。举例，新增一个参数 merchant_url，那么修改前后比较如下：

- 修改前：

```
String req_dataToken = "<direct_trade_create_req><notify_url>" + notify_url  
+ "</notify_url><call_back_url>" + call_back_url +  
"</call_back_url><seller_account_name>" + seller_email +  
"</seller_account_name><out_trade_no>" + out_trade_no +  
"</out_trade_no><subject>" + subject + "</subject><total_fee>" + total_fee  
+ "</total_fee></direct_trade_create_req>";
```

- 修改后：

```
String req_dataToken = "<direct_trade_create_req><notify_url>" + notify_url  
+ "</notify_url><call_back_url>" + call_back_url +
```



```
"</call_back_url><seller_account_name>" + seller_email +  
"</seller_account_name><out_trade_no>" + out_trade_no +  
"</out_trade_no><subject>" + subject + "</subject><total_fee>" + total_fee  
+ "</total_fee><merchant_url>" + merchant_url +  
"</merchant_url></direct_trade_create_req>";
```

merchant_url 为新增的业务参数变量，需商户为其赋值。

2. PHP修改流程

alipayapi.php 文件，在参数\$req_data 中的根节点<direct_trade_create_req>下新增子节点。举例，新增一个参数 merchant_url，那么修改前后比较如下：

- 修改前：

```
$req_data = '<direct_trade_create_req><notify_url>' . $notify_url .  
'</notify_url><call_back_url>' . $call_back_url .  
'</call_back_url><seller_account_name>' . $seller_email .  
'</seller_account_name><out_trade_no>' . $out_trade_no .  
'</out_trade_no><subject>' . $subject . '</subject><total_fee>' . $total_fee .  
'</total_fee></direct_trade_create_req>';
```

- 修改后：

```
$req_data = '<direct_trade_create_req><notify_url>' . $notify_url .  
'</notify_url><call_back_url>' . $call_back_url .  
'</call_back_url><seller_account_name>' . $seller_email .  
'</seller_account_name><out_trade_no>' . $out_trade_no .  
'</out_trade_no><subject>' . $subject . '</subject><total_fee>' . $total_fee .  
'</total_fee><merchant_url>' . $merchant_url . '</merchant_url></direct_trade_create_req>';
```

\$merchant_url 为新增的业务参数变量，需商户为其赋值。

3. ASP.NET(C#)修改流程

default.aspx.cs 文件，在参数 req_dataToken 中的根节点<direct_trade_create_req>下新增子节点。举例，新增一个参数 merchant_url，那么修改前后比较如下：

- 修改前：

```
string req_dataToken = "<direct_trade_create_req><notify_url>" + notify_url  
+ "</notify_url><call_back_url>" + call_back_url +  
"</call_back_url><seller_account_name>" + seller_email +  
"</seller_account_name><out_trade_no>" + out_trade_no +  
"</out_trade_no><subject>" + subject + "</subject><total_fee>" + total_fee  
+ "</total_fee></direct_trade_create_req>";
```

- 修改后：

```
string req_dataToken = "<direct_trade_create_req><notify_url>" + notify_url  
+ "</notify_url><call_back_url>" + call_back_url +
```

```
"</call_back_url><seller_account_name>" + seller_email +  
"</seller_account_name><out_trade_no>" + out_trade_no +  
"</out_trade_no><subject>" + subject + "</subject><total_fee>" + total_fee  
+ "</total_fee><merchant_url>" + merchant_url +  
"</merchant_url></direct_trade_create_req>";
```

merchant_url 为新增的业务参数变量，需商户为其赋值。

7.5 如何更新订单

在交易过程中，不仅需要实现能够让买家成功付款，而且还需要商户网站的订单数据与支付宝的交易管理中的交易信息保持一致。

要使信息保持一致，就需要商户网站的程序开发、维护或管理的技术人员根据商户网站的业务流程做订单更新的程序开发。

- 订单更新的业务程序代码放置位置

参数 notify_url 对应的页面文件、参数 return_url 对应的页面文件，注释指定的位置。

- 订单更新的页面文件如何被调用

前提：参数 notify_url、参数 return_url 有被正确设置。

- 当买家付款完成时，当前界面会自动跳转到参数 return_url 对应的页面文件，此时 return_url 对应的页面文件则被启动；
- 当该笔交易在支付宝交易管理中存在且交易状态改变时，支付宝会主动发送通知到参数 notify_url 对应的页面文件，此时参数 notify_url 对应的页面文件则被启动。

- 如何能获得支付宝处理完成后的通知返回数据

通知返回数据的参数详见技术文档通知返回参数列表。

- 参数 return_url 对应的页面文件中获取参数方式是 GET 方式获取，如：
request.querrystring("参数名")、\$_GET['参数名']
- 参数 notify_url 对应的页面文件中获取参数方式是 POST 方式获取，如：
request.form("参数名")、\$_POST['参数名']

- 订单更新基本思想

通过代码示例中的验证之后，获取支付宝反馈数据，先根据这笔交易找到商户网站订单系统中对应的订单，再判断该笔订单是否已经做过处理，如果未做处理，那么按照商户网站的业务流程更新订单数据。