手机网站支付开发指南

版本 1.04

修订历史

版本号	作者	内容提要	核准人	发布日期
0.1	胡叶军(薛刚)	支付宝 wap 开发指南		
0.2	陈枫（炎汐）	新增接口描述、通知部分说明		2009-09-20
0.3	陈枫（炎汐）	修改线上服务器接入地址		2009-11-06
0.4	陈枫（炎汐）	交易创建接口新增 out_user 参数与 zero_pay 参数		2009-11-26
1.0	丁朗、曹腾	内容梳理修订		2010-09-27
1.01	曹腾	新增 pay_expire 参数		2010-10-28
1.02	曹腾	buyer_account_name 参数屏蔽		2010-12-27
1.03	洛心	交易创建接口增加 call_back_url 参数，授权并执行接口删除此参数，签名注意事项修改		2011-04-20
1.04	洛心	内容修订，增加 UC 浏览器		2011-06-29
1.05	洛心	修改 submit_img_url 参数		2011-07-25

目录

1. 前言

1. 目标

该文档目的是帮助商户 WAP 应用开发人员准确、快速完成支付宝接口集成。

2. 面向读者

本文档主要面向接入支付宝 WAP 支付的外部商户的开发人员。

3. 术语

名称	说明
外部商户、合作伙伴	和支付宝进行业务合作的商户
合作伙伴 ID	签约后，为商家自动分配的唯一编号。
密钥信息	主要用来对 request 和 response 中数据做签名、验签、解密之用，自助签约系统为每个商户都配置了三种类型密钥信息：MD5、RSA、DSA. 商户可根据本接口的支持种类择一用之。
无线商户签约平台	可供商户 web 页面访问，集签约、开通接口、获取资料信息等功能于一体的在线系统。URL: https://ms.alipay.com

4. 操作流程

- 从商户签约平台中获取： 合作伙伴 ID(partnerId) 和 密钥信息。
- 根据开发指南集成支付宝接口。（可参考已发布官方 Demo）

5. 商户交流平台

商户如果在接入过程中碰到疑问，可以通过以下形式进行咨询，支付宝工作人员会给予答复。

- 支付宝官方论坛：http://club.alipay.com/thread.php?fid=747

2. 接口介绍

2.1 Wap 支付接口

2.1.1 商户开发内容描述

步骤一 ： 调用接口 alipay.wap.trade.create.direct， 提交订单信息，获取 token 串。

步骤二 ： 调用接口 alipay.wap.auth.authAndExecute， 提交 token 串，跳转到支付宝收银台。
步骤三 ： 处理支付宝系统通知。详见

2.1.2 交互模式（调用形式）

基于 http/https 的 请求/响应模式。建议使用 http 请求已适配更多机型。
http 请求地址: http://wappaygw.alipay.com/service/rest.htm
https 请求地址: https://wappaygw.alipay.com:443/service/rest.htm

2.2 浏览器安全支付

仅需在你的 wap 页面的 html 代码中付款的位置增加标签。

3. 接口详细说明

3.1. wap 支付服务

3.1.1 交易创建（alipay.wap.trade.create.direct）

该接口由支付宝提供给商户调用。

3.1.1.1 请求样例（request）

为方便展示故写成 GET 形式，正式请求时必须使用 POST 形式，以避免请求内容过长而丢失信息。

以下样例分为三类参数(下划线参数为最外层参数，共 8 个。只有 req_data 参数值中包含内层 xml 标签参数。)：
蓝色参数---- 表示该参数为必传，值可自定义。（详见参数列表）
红色参数---- 表示该参数非必传。（详见参数列表）
绿色参数---- 表示该参数为必传，并且参数值须和示例一致。（详见参数列表）

样例

http://wappaygw.alipay.com/service/rest.htm?req_data=<direct_trade_create_req><subject>商户收银台测试</subject><out_trade_no>1282889603601</out_trade_no><total_fee>1</total_fee><seller_account_name>chenf003@yahoo.cn</seller_account_name><call_back_url>http://www.alipay.com/waptest0504/servlet/CallBack</call_back_url><notify_url>http://www.alipay.com/waptest0504/servlet/NotifyReceiver</notify_url><out_user>outID123</out_user><merchant_url>http://www.alipay.com</merchant_url><pay_expire>10</pay_expire></direct_trade_create_req>&service=alipay.wap.trade.create.direct&sec_id=0001&partner=2088101000137799&req_id=1282889689836&sign=VRVr7adPfsHblFjiBkGWryhKIKt+CaI4Cq2MA2wG1ENVuBAyFDlp3FbttndmID0USlfn22a9/6fQ+X+KPDE09hcTNz3gJ1edUiDWxHXY/ahTexCP79SDtoHx29uepXsHBe32DP0k9jZbfhpT8Ly0+ksuo5VJO0iymxQ87hQPjJw=&format=xml&v=2.0

3.1.1.2 成功返回样例（response）

返回 response 参数中只要包含 res_data 参数即可认为成功返回。

当商户使用 RSA 签名方式时，实际返回的内容如下（其中 res_data 参数值为加密内容，商户需用商户 RSA 私钥先进行解密后再验签）：

样例

res_data=Cl2Mm1Z2YILG8oYe8%2FngEAvYSM9YYmcqUqLtUCZ10habqYb6poowofjzVG3nsUJY6qlgnRrq%2FxFTtdLdwBDGltV8rwpf1AFB01ydCanpQoFgQg%2Brt79JRQ%2B9CC3E%2Fg148C4F95eJ1FNf0L6taXaMFwxaxrvTAdDHzzvSigy3%2BaKdFh8z2K1Zs4gm2bD39lR1CRXSipOyVFhCZZR9L9N8tQNZbDqnyBu%2FjLdLbvXvEuE4fImZPPbsALecVCvsHL4iKFrquPnhA4Zz%2FZEM%2FoJghXA6xlAO0a1d0h6Os%2Fd83mvDPfmhs3oVjPX3FsXCL18Dg4mdzj3gWllbqLnwamM94g%3D%3D&service=alipay.wap.trade.create.direct&sec_id=0001&partner=2088201747196380&req_id=1288337908547&sign=RiyyndPEei2QQc%2FHt1%2FIrmYyW6%2FFKNZFxPUiOcXndAOo3OifNRshRjaLlwEs3d2pBpbmyclfooF7tctFdXcrSM584wgsY%2Bj2o0Z6dXst9lmz%2F4OD%2BL2ubk1DXoLWau0f5NiteIuGqGDWUdXMKRLx1FJ0f%2FMN8GOCUZYN15%2FUE%2FE%3D&v=2.0

当商户使用MD5签名方式时，实际返回的内容如下（其中res_data参数值为明文内容，无需解密）

样例

partner=2088101000137799&req_id=1283133204160&res_data=<?xml version="1.0" encoding="utf-8"?><direct_trade_create_res><request_token>20100830e8085e3e0868a466b822350ede5886e8</request_token></direct_trade_create_res>&sec_id=MD5&service=alipay.wap.trade.create.direct&v=2.0&sign=72a64fb63f0b54f96b10cefb69319e8a

3.1.1.3 失败返回样例（response）

失败返回无论哪种签名方式，内容都是明文无需解密。

样例

partner=2088101000137799&req_id=1283133132946&res_error=<?xml version="1.0" encoding="utf-8"?><err><code>0005</code><sub_code>0005</sub_code><msg>partner illegal</msg><detail>合作伙伴没有开通接口访问权限</detail></err>&sec_id=0001&service=alipay.wap.trade.create.direct&v=2.0&sign=72a64fb63f0b54f96b10cefb69319e8a

3.1.2 授权并执行(alipay.wap.auth.authAndExecute)

该接口由支付宝提供给商户调用。

3.1.2.1 请求样例（request）

Get/Post 形式都可以。

以下样例分为三类参数(下划线参数为最外层参数，共 8 个。只有 req_data 参数值中包含内层 xml 标签参数。)：

蓝色参数---- 表示该参数为必传，值可自定义。（详见参数列表）

红色参数---- 表示该参数非必传。（详见参数列表）

绿色参数---- 表示该参数为必传，并且参数值须和示例一致。（详见参数列表）

样例（如用 GET 方式请求需要 URL Encode）：

http://wappaygw.alipay.com/service/rest.htm?req_data=<auth_and_execute_req><request_token>201008309e298cf01c58146274208eda1e4cdf2b</request_token></auth_and_execute_req>&service=alipay.wap.auth.authAndExecute&sec_id=0001&partner=2088101000137799&sign=LdXbwMLug8E4UjfJMuYv2KoD5X5F3vHGQsQbZ/rdEQ3eaN4FPal7rhsbZZ/+ZUL1kAKzTQSDdMk87MEWtWO1Yq6rhnt2Tv8Hh6Hb16211VXKgbBCpq861+LopRwegPbGStcwBuAyE4pi6fYlJ6gxzL4tMyeLe+T5XZ0RKRUk00U=&format=xml&v=2.0

3.1.2.2 成功返回页面（response）

3.1.2.3 失败返回页面（response）

3.1.3 处理支付宝系统通知(notify_url)

商户需要提供一个 http 的 URL(例:http://www.partnertest.com/servlet/NotifyReceiver)来接受支付宝后台请求。当交易状态发生变化，支付宝后台都会用 POST 方式主动调用该接口。商户必须在收到 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），并返回 success（注意：支付宝需精确匹配该字样，不能包含任何其他的 HTML 脚本语言）表示通知处理完毕，反之支付宝系统会

对同一笔订单进行周期性的通知重发(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)。

3.1.3.1 请求样例（request）

当商户使用 RSA 签名方式时，商户实际收到支付宝通知请求如下（其中 notify_data 参数值为加密内容，商户需用商户 RSA 私钥先进行解密后再验签）：

样例：

http://www.partnertest.com/servlet/NotifyReceiver?service=alipay.wap.trade.create.direct&sign=Rw/y4ROnNicXhaj287Fiw5pvP6viSyg53H3iNiJ61D3YVi7zGniG2680pZv6rakMCeXX++q9XRLw8Rj6I1//qHrwMAHS1hViNW6hQYsh2TqemuL/xjXRCY3vjm1HCoZOUa5zF2jU09yG23MsMIUx2FAWCL/rgbcQcOjLe5FugTc=&sec_id=0001&v=1.0&notify_data=g3ivqicRwI9rI5jgmSHSU2osBXV1jcxohapSAPjx4f6qiqsoAzstaRWuPuutE0gxQwzMOtwL3npZqWO3Z89J4w4dXIY/fvOLoTNn8FjExAf7OozoptUS6suBhdMyo/YJyS3lVALfCeT3s27pYWihHgQgna6cTfgi67H2MbX40xtexIpUnjgxBkmOLai8DPOUI58y4UrVwoXQgdcwnXsfn2OthhUFiFPfpINgEphUAq1nC/EPymP6ciHdTCWRI6l1BgWuCzdFy0MxJLliPSnuLyZTou7f+Z5Mw24FgOacaISB+1/G+c4XIJVKJwshCDw9Emz+NAWsPvq34FEEQXVAeQRDOphJx8bDqLK75CGZX+6fx88m5ztq4ykuRUcrmoxZLJ+PiABvYFzi5Yx2uBMP/PmknRmj1HUKEhuVWsXR0t6EWpJFXlyQA4uxbShzncWDigndD7wbfNtkNLg5xMSFFIKay+4YzJK68H9deW4xqk4JYTKsv8eom9Eg9MrJZiIrFkFpVYPuaw0y/n61UEFYdzEQZz+garCmMYehEAQCGibYUQXBlf1iwTOZdqJIxdgCpSX21MIa9N9jicmFu8OXWZJkdN+UrSyvIcpzRori+U6522ovMz5Z8EzVTfcUENu+dWJRnhFlo6pvm0a3Fq2wBEyUV1/YYS3LaZiPj+wig5BCyJ92QXZnEUEtn87oX5kuzSRuLcVVi8OJIgyQWaWT9N0YFyH5AfV+VDNxu4UYy6KkGtcaVjSvbzDuzThMXs2HDwX3qujq25A/hzJKlgR9EjcumJeF/TM6eS7JS+FKXE1kUXnMnGbokaNemZn2yKlPC1VO4LU77G5v1nUs6MfYFq9HC4FYiQ6Y+hL8RgAMorty/RYT3cZ8SQCTO0bQ+qJuOnx79YEEmCUQc3iJBp0zFKYXIU6viqJYghEs6F3LiK8TvJRo8+ST5hKtnuU5b8R6f9yD8Uek1BruWvlYaA7I3Cc90CDhTyOghL2oCMOoKlxqgXdh3MGm128FOVyCjDLRw04b+kK83JGFMcjyVuhfhoVeETQicUCtFQ9ItlH3uFkB5su+r3399WGSXyGflrTbFhMq7mRztWotL2ATvf/enMBcGSCSCb47OzGxXhMDGZZu4Sq4pdF9fsZVBHgWsB/KS8bwxyvM068NoqnRmI72zgL7WFWumlm88j3K6KPxbB6soDSXRv6drbSv2t93llE5q4SP6GLztAw7UPWGTJLXOFyhyaszvhyZWxsX+C5PbXoCta1/cxt4Sp4WXDjaHn6qHI/Vea28xx8fYV/xK5WTmvFwb0k9eRGCgB6/nzmGV1+lPJuK3pKy3L5LbUP0zJFh5gdPG7DecH+F0uBUC0QNMQ==

当商户使用 MD5 签名方式时，商户实际收到支付宝通知请求如下（其中 notify_data 参数值为明文内容，无需解密。）

样例：

http://www.partnertest.com/servlet/NotifyReceiver?service=alipay.wap.trade.create.direct&sign=Rw/y4ROnNicXhaj287Fiw5pvP6viSyg53H3iNiJ61D3YVi7zGniG2680pZv6rakMCeXX++q9XRLw8Rj6I1//qHrwMAHS1hViNW6hQYsh2TqemuL/xjXRCY3vjm1HCoZOUa5zF2jU09yG23MsMIUx2FAWCL/rgbcQcOjLe5FugTc=&v=1.0&sec_id=MD5&notify_data=<notify><payment_type>1</payment_type><subject>收银台【1283134629741】</subject><trade_no>2010083000136835</trade_no><buyer_email>dinglang@a.com</buyer_email><gmt_create>2010-08-30 10:17:24</gmt_create><notify_type>trade_status_sync</notify_type><quantity>1</quantity><out_trade_no>1283134629741</out_trade_no><notify_time>2010-08-30 10:18:15</notify_time><seller_id>2088101000137799</seller_id><trade_status>TRADE_FINISHED</trade_status><is_total_fee_adjust>N</is_total_fee_adjus

t><total_fee>1.00</total_fee><gmt_payment>2010-08-30 10:18:26</gmt_payment><seller_email>chenf003@yahoo.cn</seller_email><gmt_close>2010-08-30 10:18:26</gmt_close><price>1.00</price><buyer_id>2088102001172352</buyer_id><notify_id>509ad84678759176212c247c46bec05303</notify_id><use_coupon>N</use_coupon></notify>

3.2. 浏览器安全支付服务

3.2.1 实现效果

往右边流程是用户手机没有安装安全支付的情况，需要下载安装，往下是手机已经安装安全支付，直接调起，进入收银台。

3.2.2 如何让 UC 浏览器识别出“安全支付”logo

仅需在你的 wap 页面的 html 代码中付款的位置增加标签

标签接口定义：

为兼容 Wml 和 xhtml 网页，支付标签以注释实现。

第一部分：**ALIPAYMSP** 工作显示模式

格式定义 :<!--ALIPAYMSP[固定参数][定单参数名1=”参数值1”&参数名2=”参数值2”&…参数名 n=”参数值 n”]-->

格式说明：

1) <!--ALIPAYMSP [][]--> 为[固定格式][定单参数和值采用 K=V 的方式。]

2) 多个参数之间以“&”号连接

3) 参数名和参数值都全部需要进行 Html 编码

4) 参数名大小写敏感，建议定义时全部使用小字

样例（红色为固定参数）：

<!--ALIPAYMSP[v=”1.0”&submit_img_url=”http://wappaygw.alipay.com/images/safebutton.png”][ordertoken=”4579807260245”&timestamp=”20100112208095423”]-->

固定参数说明：

1) v：版本号。本期全部写为 1.0，本期 UC 浏览器 hardcode 只支持 1.0，以备后面的协议升级兼容。

2) submit_img_url：安全支付显示的图标的 url

订单参数说明：

1） ordertoken：提交订单信息后返回的 token 值。

2） timestamp：时间戳。

UC 浏览器解析上述标签，会转化为：

如果使用了安全支付按钮后想屏蔽原有的 wap 支付按钮：

第二部分：ALIPAYMSP 工作时隐藏域内容标签模式

<!--ALIPAYMSP-ENABLE-BEGIN-->

xxx-这部分如果安全支付打开将自动隐藏!

<!--ALIPAYMSP-ENABLE-END-->

格式说明：当浏览器正常工作能发现有 ALIPAYMSP 标签时将原页面上的 <!--ALIPAYMSP-ENABLE-BEGIN--> 至 <!--ALIPAYMSP-ENABLE-END--> 部分（不能分析 ALIPAYMSP 标签浏览器的 WAP 定单内容）自动隐藏。

3.2.3 上述例子中的 token 如何生成

如果您之前有接入过支付宝的 wap 支付产品，那么很简单，你继续调用 wap 的 alipay.wap.trade.create.direct 接口就可以获得 token 了，更加详细的接口规范和签名规范请参考 3.1.1 节交易创建（alipay.wap.trade.create.direct）

文档可以在自助签约平台(http://ms.alipay.com) 产品服务频道的手机网站支付服务下下载到.

3.2.4 处理支付宝系统通知(notify_url)

详见 3.1.3

4. 附录

4.1. 所有参数查询列表

参数名	中文描述	类型(精度)	说　明	商户必传	参数值样例
service	接口名称	String	注意：交易创建、授权并执行两次请求的值不同。	Y	alipay.wap.trade.create.direct/alipay.wap.auth.authAndExecute
partner	合作伙伴 id	String(16)	合作伙伴在支付宝的用户 ID，与支付宝签约后自动生成	Y	2088002007015955
sec_id	签名算法	String(4)	签名算法。目前只支持 MD5 和 RSA(用 0001 表示)	Y	0001　或　MD5
req_id	请求号	String(32)	请求号用于关联请求与响应，并且防止请求重播。支付宝 wap 限制来自一个 partner 的请求号必须唯一。	Y	1e925b9b4b115961660130f9281e3898

sign	签名	String	签名，对 request/response 中参数签名后的值，详见	Y	72020eb70e0fdcfbf404edcbb83bfd81
format	请求参数格式	String	参数值必须和样例保持一致	Y	xml
v	接口版本号	String	参数值必须和样例保持一致	Y	2.0
req_data	请求业务参数	String	参数值内容为 xml 格式,包含内层标签参数	Y	<?xml version="1.0" encoding="UTF-8"?><direct_trade_create_req> <subject>彩票</subject><out_trade_no>20080801-1</out_trade_no><total_fee>50</total_fee><seller_account_name>tbbusi003@126.com</seller_account_name><out_user>xxxxx</out_user><notify_url>http://www.yoursite.com/notifyurl.htm</notify_url></direct_trade_create_req>
direct_trade_create_req	固定标签	String	req_data 参数值 xml 内容中必须包含的固定标签。	Y	<subject>彩票</subject><out_trade_no>20080801-1</out_trade_no><total_fee>50</total_fee><seller_account_name>tbbusi003@126.com</seller_account_name><out_user>xxxxx</out_user><notify_url>http://www.yoursite.com/notifyurl.htm</notify_url>
subject	商品名称	String(256)	订单商品名称	Y	彩票
out_trade_no	外部交易号	String(64)	合作伙伴系统的交易号,传递给支付宝系统做外部交易号（不能重复）	Y	2008080101
total_fee	订单价格	String(15)	用户购买的商品或服务的价格（必须是金额的格式,单位：元）	Y	1.01
pay_expire	交易自动关闭时间	Int	买家如未能在该设定值范围内支付成功，交易将被关闭。单位：“分钟”，值区间 0<pay_expire，默认值 21600（15 天）。最终关闭时间点误差 1-2 分钟。	N	10

seller_account_name	卖家帐号	String(100)	交易卖方的支付宝帐号，交易成功后该笔交易的资金会转入这个支付宝帐号中	Y	tbbusi003@126.com
out_user	商户系统用户唯一标识	String(32)	买家在商户系统的唯一标识，当该 out_user 支付成功一次后再来支付时，30 元内无需密码。	N	21321211111
notify_url	商户接受通知的 url	String(200)	商户接受通知的 url，详见 。	Y	http://www.yoursite.com/notifyurl.htm
merchant_url	返回商户链接	String	用户在支付宝页面可返回商户的链接	N	http://www.yoursite.com/partnerurl.htm
call_back_url	支付成功跳转链接	String(200)	由商户提供，只有当交易支付成功之后，才会跳转到该 url。	N	http://www.yoursite.com/callbackurl.htm
request_token	token	String(40)	前面调用交易创建接口成功返回后获得的**(注当此参数为页面返回时，为固定值)**		20081113f9d49c20e8e5c8e40b6107ec42259e41
trade_no	交易号	String(64)	交易号，该笔交易在支付宝系统的交易号		2009092904171521
notify_data	通知业务参数	String	通知的业务参数，包含交易号、外部交易号、交易状态等信息。		见例子
payment_type	支付方式	String	用户的支付方式(商户可不关心该参数)		1
buyer_email	买家账号	String(100)	买家的支付宝账号		chenf002@yahoo.cn
gmt_create	创建时间	String	交易创建时间		2009-09-29 19:59:24
notify_type	通知类型	String	该通知的类型，暂时只有交易状态同步(商户可不关心该参数)		trade_status_sync
quantity	数量	String	购买商品数量		1
notify_time	通知时间	String	发送通知的时间		2009-09-29 19:59:25
seller_id	卖家 id	String	卖家的支付宝账号 id		2088102001058148
trade_status	交易状态	String	交易的状态。TRADE_FINISHED（支付成功），WAIT_BUYER_PAY(等待买家付款)		TRADE_FINISHED/ WAIT_BUYER_PAY
is_total_fee_adjust	总价是否被修改	String	交易价格是否被修改，Y 或 N(本接口创建的交易不会被		N

			修改)		
total_fee	交易总价	String	即订单金额。单位：元		2.21
gmt_payment	付款时间	String	交易的付款时间,如果交易未付款，没有该属性		2009-09-29 19:59:25
seller_email	卖家账号	String(100)	卖家的支付宝账号		youngbeckham@gmail.com
gmt_close	交易结束时间	String	交易结束的时间		2009-09-29 19:59:25
price	单个商品价格	String	目前和 total_fee 值相同。单位：元		2.21
buyer_id	买家 id	String	买家的支付宝账号 id		2088101000137393
notify_id	通知 id	String	唯一识别通知内容,重发相同内容的通知 notify_id 值不变。		2311b764be6fba98f593ba98f7eb7470
use_coupon	是否使用红包	String	交易时是否使用红包，Y 或 N		N

4.2. 错误代码列表

错误代码	说明
0000	系统异常
0001	缺少必要的参数，检查非空参数是否已经传递
0002	签名错误，检查签名的参数是否符合支付宝签名规范
0003	服务接口错误，检查 service 是否传递正确
0004	req_data 格式不正确
0005	合作伙伴没有开通接口访问权限，合同是否有效
0006	sec_id 不正确，支持 0001，MD5
0007	缺少了非空的业务参数

4.3. 签名规范

为了确保数据传输过程中的数据真实性和完整性，支付宝和商户都需要对 request/response 数据进行签名验证。目前本接口支持的签名算法为 MD5、RSA。

4.3.1. 签名注意事项

a．没有值的参数无需传递，也无需包含到待签名数据中。

b. 签名时将字符转变成字节流时指定的字符集要与_input_charset保持一致。

c. 如果传递了_input_charset参数，那么这个参数也应该包含在待签名数据中。

d. 根据HTTP协议要求，传递参数的值中如果存在特殊字符（如：&、@等），那么该值需要做URL Encoding，这样请求接受方才能接受到正确的参数值。这种情况下，做签名时使用的应该是原生值而不是encoding后的值。例如：待签名数据是

email=test@msn.com&partner=2088006300000000&service=test，而不是

email=test%40msn.com&partner=2088006300000000&service=test

4.3.2. 用于生成 sign 的待签名数据构造规则

HTTP 传递的所有参数(**除 sign 以外**)按照参数名称字符升序的顺序串联起来（如：p1=v1&p2=v2&p3=v3），构成待签名数据。按照 sec_id 指定的算法对待签名数据进行签名。

例如：调用某接口需要以下参数：

service=cae_charge_agent

partner=2088006300000000

email=test@msn.com

那么待签名数据就是：email=test@msn.com&partner=2088006300000000&service=cae_charge_agent。

4.3.3. 支付宝系统通知待签名数据构造规则

支付宝系统通知待签名数据构造规则比较特殊，为固定顺序。

商户收到如下请求:

```
http://www.partnertest.com/servlet/NotifyReceiver?service=alipay.wap.trade.create.direct&sign=Rw/y4ROnNicXhaj287Fiw5pvP6viSyg53H3iNiJ61D3YVi7zGniG2680pZv6rakMCeXX++q9XRLw8Rj6I1//qHrwMAHS1hViNW6hQYsh2TqemuL/xjXRCY3vjm1HCoZOUa5zF2jU09yG23MsMIUx2FAWCL/rgbcQcOjLe5FugTc=&v=1.0&sec_id=0001&notify_data=<notify><payment_type>1</payment_type></notify>
```

则只需对以下数据验签:

```
service=alipay.wap.trade.create.direct&v=1.0&sec_id=0001&notify_data=<notify><payment_type>1</payment_type></notify>
```

4.3.4. 签名算法对比

特性 \ 算法	MD5	RSA
实现简单	√	×
防篡改	√	√
防抵赖	×	√
加密	×	√
电子签名法是否承认	×	√

4.3.5. MD5 算法签名

定义：MD5 是一种摘要生成算法，本来是不能用于签名的。但是，通过在待签名数据之后加上一串私密内容（指令发送、接收双方事先规定好的，这里我们称其为签名密钥），就可以用于签名了。使用这种算法签名只能起到防数据篡改的功能，不能起到签名防抵赖的功能，因为双方都知道签名密钥

- 登录 **无线商户签约平台** 获取 我的产品>>密钥管理 >>**安全校验码（MD5）**。
- 生成 sign 值 JAVA 示例：content =待签名数据+安全校验码（MD5）；String sign = DigestUtils.md5Hex(content.getBytes("utf-8"));
- 验签时商户只需用 response 中的参数签名后和 sign 值比较，一致说明验签成功。

4.3.6. RSA 算法签名

定义：RSA 是一种非对称的签名算法，即签名密钥（私钥）与验签名密钥（公钥）是不一样的，私钥用于签名，公钥用于验签名。使用这种算法签名在起到防数据篡改功能的同时，还可以起到防抵赖的作用，因为私钥只有签名者知道。

- 用集成文档压缩包中 openssl 工具生成一套 RSA 公私钥。登录 **无线商户签约平台**，把公钥用 txt 文件上传至我的产品>>密钥管理>>**安全校验码（RSA）>>商户公钥** 。并获取 **安全校验码（RSA）>>支付宝公钥**

RSA 密钥生成命令:

生成 RSA 私钥

openssl genrsa -out rsa_private_key.pem 1024

生成 RSA 公钥

openssl rsa -in rsa_private_key.pem -pubout -out rsa_public_key.pem

将 RSA 私钥转换成 PKCS8 格式

openssl pkcs8 -topk8 -inform PEM -in rsa_private_key.pem -outform PEM -nocrypt

- 生成 sign 值的示例代码(非真实代码仅供参考)：

content: 按 4.3.2 生成。 privateKey: openssl 生成的私钥。

```
public String sign(String content, String privateKey) throws Exception {
   java.security.KeyFactory keyFactory =
   java.security.KeyFactory.getInstance("RSA");
   byte[] encodedKey = StreamUtil.readText(new ByteArrayInputStream(
  privateKey.getBytes())).getBytes();
```

```
    // 先base64解码
    encodedKey = Base64.decodeBase64(encodedKey);
    java.security.PrivateKey prikey = keyFactory.generatePrivate(new
    java.security.spec.PKCS8EncodedKeySpec(encodedKey));
    java.security.Signature signature =
    java.security.Signature.getInstance("SHA1WithRSA");
    signature.initSign(prikey);
    signature.update(content.getBytes("utf-8"));
    byte[] signBytes = signature.sign();
    String sign = new String(Base64.encodeBase64(signBytes));
    return sign;
}
```

- 验签 JAVA 示例代码(非真实代码仅供参考):

content：按4.3.2 或 4.3.3 生成。
sign ：支付宝返回给商户的sign值。
publicKey：登录无线商户签约平台获取 安全校验码（RSA）>>支付宝公钥

```
public boolean verify(String content, String sign, String publicKey) throws
Exception {
    java.security.KeyFactory keyFactory =
    java.security.KeyFactory.getInstance("RSA");
    StringWriter writer = new StringWriter();
    StreamUtil.io(new InputStreamReader(new ByteArrayInputStream(publicKey
    .getBytes())), writer);
    byte[] encodedKey = writer.toString().getBytes();
    // 先base64解码
    encodedKey = Base64.decodeBase64(encodedKey);
    java.security.PublicKey pubKey = keyFactory.generatePublic(new
    java.security.spec.X509EncodedKeySpec(encodedKey));
    byte[] signed = Base64.decodeBase64(sign.getBytes());
    java.security.Signature signature =
    java.security.Signature.getInstance("SHA1WithRSA");
    signature.initVerify(pubKey);
    signature.update(content.getBytes("utf-8"));
    boolean verify = signature.verify(signed);
    return verify;
}
```