91 移动开发平台服务端与应用服务端接口规范

网龙网络有限公司 无线事业部

版本: 1.00

时间	版本	更新说明
2010-10-27	1.00	创建
2010-11-01	1.00	添加发送用户动态接口
2010-11-04	1.00	添加应用服务直接扣费接口
2010-11-25	1.00	修改接收支付购买结果通知接口，加入 Act 参数，并调整 Sign 算法，添加接收应用自定义虚拟币充值结果通知接口。
2010-12-14	1.00	调整 MD5 算法，改用 UTF8 编码字符串
2011-01-11	1.00	添加接口 3,4,5,6
2011-01-13	1.00	修改接口 2
2011-01-18	1.00	修改接口 2
2011-02-18	1.00	修改部分用词，添加接口调用与返回值的示例
2011-07-22	1.00	添加接口接收平台事件通知
2012-03-23	1.00	查询支付购买结果(Act=1)接口的 PayStatus 参数添加返回值 2（订单处理中）
2012-06-15	1.00	添加软件互推接口（五.4） 添加支付验证（主动）模式的注意事项

目录

一、 概要

本文档是作为 91 移动开放平台服务器与各应用服务器相互交互的接口规范.. 服务器对接不是必须功能. 开发者可根据自己应用的需要,考虑是否需要两方服务器对接. 通常单机版的应用或者对安全性要求不高的应用, 直接使用客户端的 SDK 就可满足功能需要.

91 移动开发平台 SDK 分精简版 SDK(仅支持登录,注册,支付) 和 完全版 SDK(在精简版 SDK 基础上,增加好友,消息功能等等). 以下接口按各自应用所接入的 SDK 版本不同,使用不同的接口. 如无特别标注, 接口两个版本的 SDK 都适用.

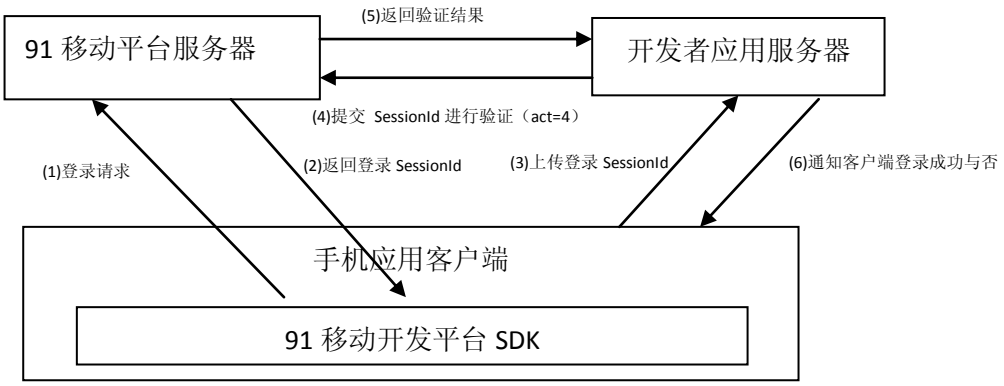
接口文档中涉及的 AppId, AppKey, 需要通过 dev.91.com 的开发者后台进行申请. 由于涉及到加密通信, 开发者必须严格对 AppKey 进行保密.

本文档涉及的技术均采用主流的 Web 开发技术, 相关技术(如 JSON)等若不清楚, 可通过互联网搜索即可了解. 本文档不做单独解释

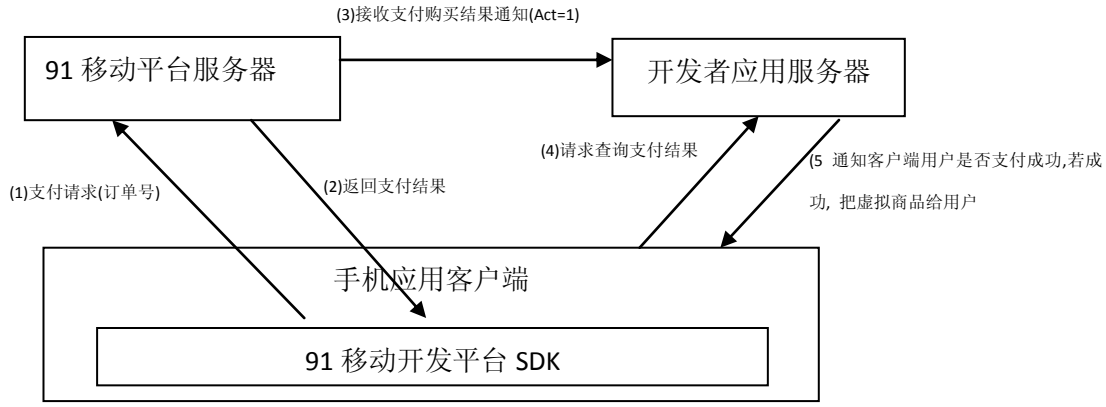
我们提供了基于 C#开发的以下接口的接入 Demo, 并提供源码. 开发者可参考该源码进行开发和测试.

二、 基本流程

1、登录验证



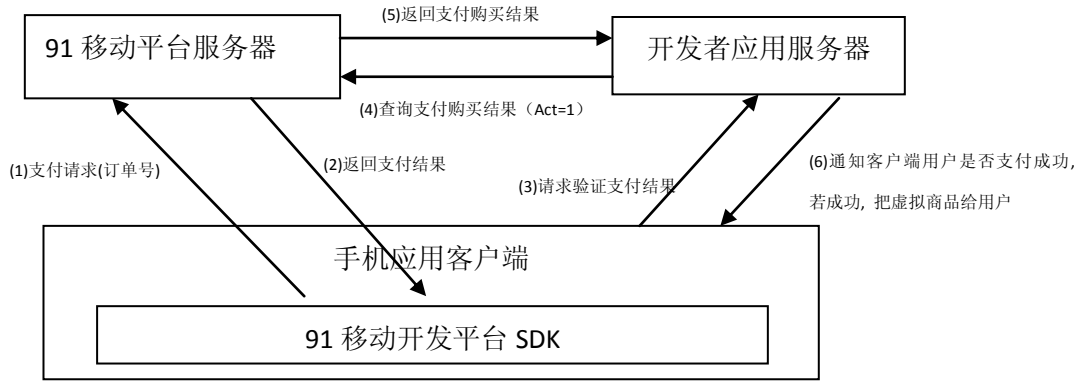
2、 支付验证(通知模式)



注: 由于是异步通知模型, (3), (4) 不一定是按序号产生. 因此(4), (5) 需要进行轮询处理,或者使用接口进行支付结果查询.

注：考虑性能和及时性的原因，推荐使用此模式

3、 支付验证(查询模式)



注：不推荐由手机应用客户端直接调用主动查询支付验证接口（正确方式应通过如上图所示的开发者应用服务器作中转），客户端直接调用可能导致由于接口参数变更时，旧手机应用客户端无法即时更新而调用失败。

注：91 服务器会对查询支付验证接口做相同参数内容调用的时间间隔限制，目前时间间隔为：2 秒

注意： 由于短信支付，充值渠道合作方支付，具有一定的延迟性，特殊情况下，甚至会有较长的延迟时间，我们强烈建议你使用支付验证通知模式，即 91 服务器通知支付结果的方式，支付一旦成功，立即通知到业务服务器。通常支付验证查询模式，即查询模式，建议在掉单，或者订单验证时，使用。

三、 接口格式

- ✓ **接口地址**: 具体见各个接口说明
- ✓ **调用方式:** 数据用 HTTP GET 方式上传到接口地址。
- ✓ **输入参数:** ?参数 1=值 1&参数 2=值 2&….&参数 n=值 n，如果遇到文本参数值，需要根据情况对参数值做 UrlEncode，避免因为文本参数值中包含&=等引起参数解析错误的问题
- ✓ **输出参数:** Json 格式表示，如： {"返回参数 1":"返回值 1","返回参数 2":"返回值 2",…. "返回参数 n":"返回值 n"}
- ✓ **MD5 算法: 标准 MD5 算法，结果为小写的 16 进制字符串**
  具体的示例代码请参见附录

四、 应用服务器向 91 服务器获取数据接口

正式外网：http://service.sj.91.com/usercenter/AP.aspx
验证接口地址，该地址是一个 Demo 页面，模拟了各个接口的调用的参数及返回值，应用服务商可以在该页面上验证接口：
http://mobileusercenter.sj.91.com/usercenter1/Default.aspx
注： 验证接口仅是验证参数，签名是否正确，不是测试地址。目前服务端接入都是接的正式外网

以下所有的 HashToMD5Hex() 签名算法实现，请参见附录

1、 查询支付购买结果(Act=1)

功能描述	应用服务商向 91 移动开发平台服务器查询支付购买的订单结果	
应用服务器传给 91 服务器参数	**参数名**	**说明(**没有说明可选的参数，为必传参数**)**
	AppId	应用 ID
	Act	1
	CooOrderSerial	商户订单号
	Sign	参数值与 AppKey 的 MD5 值 String.Format("{0}{1}{2}{3}", AppId, Act, CooOrderSerial, AppKey).HashToMD5Hex();
	例如： AppId=100010&Act=1&CooOrderSerial=4764f9ff47174e2287ded31673a73a50&Sign=b4a67099d354dce29ab8a9b526344cb6	
91 服务端处理	1. 验证 Sign 是否有效 2. 读取订单数据 3. 返回结果给应用服务器	

91 服务器返回应用服务器参数	ConsumeStreamId	消费流水号，平台流水号
	CooOrderSerial	商户订单号，购买时应用传入，原样返回给应用
	MerchantId	商户 ID
	AppId	应用 ID
	ProductName	应用名称
	Uin	91 账号 ID，购买时应用传入，原样返回给应用
	GoodsId	商品 ID，购买时应用传入，原样返回给应用
	GoodsInfo	商品名称，购买时应用传入，原样返回给应用
	GoodsCount	商品数量，购买时应用传入，原样返回给应用
	OriginalMoney	原价(格式：0.00)，购买时应用传入的单价*总数，总原价
	OrderMoney	实际价格(格式：0.00)，购买时应用传入的单价*总数，总实际价格
	Note	即支付描述（客户端 API 参数中的 payDescription 字段） 购买时客户端应用通过 API 传入，原样返回给应用服务器 开发者可以利用该字段，定义自己的扩展数据。例如区分游戏服务器
	PayStatus	支付状态：0=失败，1=成功，2=正在处理中 (仅当 ErrorCode=1，表示接口调用成功时，才需要检查此字段状态，开发商需要根据此参数状态发放物品)
	CreateTime	创建时间(yyyy-MM-dd HH:mm:ss)
	Sign	以上参数的 MD5 值，其中 AppKey 为 91SNS 平台分配的应用密钥 String.Format("{0}{1}{2}{3}{4}{5}{6}{7}{8}{9:0.00}{10:0.00}{11}{12}{13:yyyy-MM-dd HH:mm:ss}{14}", ConsumeStreamId, CooOrderSerial, MerchantId, AppId, ProductName, Uin, GoodsId, GoodsInfo, GoodsCount, OriginalMoney, OrderMoney, Note, PayStatus, CreateTime, AppKey).HashToMD5Hex()
	ErrorCode	错误码(0=失败，1=成功，2= AppId 无效，3= Act 无效，4=参数无效，5= Sign 无效, 11=没有该订单)
	ErrorDesc	错误描述
	{"ConsumeStreamId":"1-10001-20110218131629-2-1238","CooOrderSerial":"4764f9ff47174e2287ded31673a73a50","MerchantId":"10001","AppId":"100010","ProductId":"100010","ProductName":"测试应用1xx","Uin":"704458214","GoodsId":"100010","GoodsInfo":"超级化肥","GoodsCount":"2","OriginalMoney":"0.02","OrderMoney":"0.02","Note":"购买2 袋超级化肥","PayStatus":"1","CreateTime":"2011-02-18 13:16:29","Sign":"dfa36dfb18d6a43984e3ab01884f1f4e","ErrorCode":"1","ErrorDesc":""}	
备注	例子中的 ProductId 即 AppId，兼容旧版所用，两者传一个即可	

2、发送用户动态(Act=2)(完全版 SDK 适用)

功能描述	应用服务商向 91 移动开发平台服务器发送用户动态		
应用服务器传给 91 服务器参数	**参数名**	**说明(**没有说明可选的参数，为必传参数**)**	
	AppId	应用 ID	
	Act	2	
	Uin	用户的 91Uin	
	TemplateId	模板 ID	
	ParamList …	参数列表 ,参数名称同模板里的参数名称 ,通过 URL 参数传递时，先对下列参数进行 Json 编码，示例如下	
		Uin	11323356
		ResourceInfo	1000035
		例如：{"ResourceInfo":"1000035","Uin":"11323356"}	
	Sign	参数值与 AppKey 的 MD5 值 String.Format("{0}{1}{2}{3}{4}{5}", AppId, Act, Uin, TemplateId, ParamList, AppKey).HashToMD5Hex();	
	例如： AppId=100010&Act=2&Uin=11323356&TemplateID=11&ParamList=%7b%22ResourceInfo%22%3a%221000035%22%2c%22Uin%22%3a%2211323356%22%7d&Sign=1641058ce70232ec205e639972bea8e5		
91 服务端处理	1. 验证 Sign 是否有效 2. 验证模板是否有效 3. 将动态发送到 UAP 系统 4. 返回结果给应用服务器		
91 服务器返回应用服务器参数	ErrorCode	错误码(0=失败，1=成功，2= AppId 无效，3= Act 无效，4=参数无效，5= Sign 无效, 11=模板 ID 无效, 12=参数错误)	
	ErrorDesc	错误描述	
	例如：{"ErrorCode":"1","ErrorDesc":"成功"}		
备注			

3、获取当前应用的用户列表(Act=3) (完全版 SDK 适用)

功能描述	获取当前应用的用户列表	
应用服务器传给 91 服务器参数	**参数名**	**说明(**没有说明可选的参数，为必传参数**)**
	AppId	应用 ID
	Act	3

	PageNo	第几页页号(从 1 开始)
	PageSize	每页数量(5 的倍数)
	Sign	参数值与 AppKey 的 MD5 值 String.Format("{0}{1}{2}{3}{4}", AppId, Act, PageNo, PageSize, AppKey).HashToMD5Hex();
	例如： AppId=100010&Act=3&PageNo=1&PageSize=5&Sign=b046f564e558915159edb1cfbcea6742	
91 服务端处理	1. 验证 Sign 是否有效 2. 获取当前应用的用户列表 3. 返回结果给应用服务器	
91 服务器返回应用服务器参数	ErrorCode	错误码(0=失败，1=成功，2= AppId 无效，3= Act 无效，4=参数无效，5= Sign 无效)
	ErrorDesc	错误描述
	UserList	符合条件的用户列表 单个用户数据项格式为：Uin,昵称,性别,省份,地区,年龄,CheckSum,在线状态 用户项之间以分号做分隔符，最后一个用户不用加分隔符 在线状态：0=未知，1=在线，2=离线，目前在线状态统一为 0
	{"UserList":"11270233,风云之雄霸天下 kkkkkkkkkkk,0,,,0,ffffffffffffffffffffffffffffffff,0;704458214,wyl1234567,0,,,0,ffffffffffffffffffffffffffffffff,0;2828833,Qq,0,,,0,ffffffffffffffffffffffffffffffff,0;11323356,神吗,1,福建省,泉州市,0,d2c89e5006fc75959bf8fa650e620e57,0;11195353,雄霸 m1044026,0,,,0,ffffffffffffffffffffffffffffffff,0","TotalCount":"62","ErrorCode":"1","ErrorDesc":"成功"}	
备注		

4、 检查用户登录 SessionId 是否有效(Act=4)

功能描述	检查用户登录 SessionId 是否有效	
应用服务器传给 91 服务器参数	**参数名**	**说明(**没有说明可选的参数，为必传参数**)**
	AppId	应用 ID
	Act	4
	Uin	用户的 91Uin
	SessionId	用户登录 SessionId
	Sign	参数值与 AppKey 的 MD5 值 String.Format("{0}{1}{2}{3}{4}", AppId, Act, Uin, SessionId ,AppKey).HashToMD5Hex();
	例如：	

	AppId=100010&Act=4&Uin=11323356&Sign=76bd057af04298da8379359810b8c9f6&SessionID=5e9c3844563640daa9b9d4846031bbd4	
91 服务端处理	1. 验证 Sign 是否有效 2. 验证 SessionId 是否有效 3. 返回结果给应用服务器	
91 服务器返回应用服务器参数	ErrorCode	错误码(0=失败，1=成功(SessionId 有效)，2= AppId 无效，3= Act 无效，4=参数无效，5= Sign 无效，11=SessionId 无效)
	ErrorDesc	错误描述
	例如：{"ErrorCode":"1","ErrorDesc":"有效"}	
备注		

5、 获取用户的好友列表(Act=5) (完全版 SDK 适用)

功能描述	获取用户的好友列表	
应用服务器传给 91 服务器参数	**参数名**	**说明(**没有说明可选的参数，为必传参数**)**
	AppId	应用 ID
	Act	5
	Uin	用户的 91Uin
	SessionId	用户的客户端 SessionId
	PageNo	第几页页号(从 1 开始)
	PageSize	每页数量(5 的倍数)
	Sign	参数值与 AppKey 的 MD5 值 String.Format("{0}{1}{2}{3}{4}{5}{6}", AppId, Act, Uin, SessionId, PageNo, PageSize, AppKey).HashToMD5Hex();
	例如： AppId=100010&Act=5&Uin=11323356&Sign=7634109403de4fee688a22fcf7da242a&PageNo=1&PageSize=5&SessionID=5e9c3844563640daa9b9d4846031bbd4	
91 服务端处理	1. 验证 Sign 是否有效 2. 验证 SessionId 是否有效 3. 获取用户的好友列表 4. 返回结果给应用服务器	
91 服务器返回应用服务器参数	ErrorCode	错误码(0=失败，1=成功，2= AppId 无效，3= Act 无效，4=参数无效，5= Sign 无效)
	ErrorDesc	错误描述
	FriendList	好友列表 每个好友项格式为：Uin,昵称,积分,心情,CheckSum,在线状态 好友项之间以分号做分隔符，最后一个好友不用加分隔符 在线状态：0=未知，1=在线，2=离线，目前在线状态统一为 0

	{"FriendList":"11673967,机友 11673967201409,0,,ffffffffffffffffffffffffffffffff,0;11995720,雄霸 100,0,,ffffffffffffffffffffffffffffffff,0;704458214,wyl1234567,0,,ffffffffffffffffffffffffffffffff,0;11749323,机友 11749323175635,0,,ffffffffffffffffffffffffffffffff,0;11996097,机友 11996097185817,0,,ffffffffffffffffffffffffffffffff,0","TotalCount":"20","ErrorCode":"1","ErrorDesc":"成功"}
备注	

6、 获取用户的当前应用的好友列表(Act=6) (完全版 SDK 适用)

功能描述	获取用户的当前应用的好友列表	
应用服务器传给 91 服务器参数	**参数名**	**说明(**没有说明可选的参数，为必传参数**)**
	AppId	应用 ID
	Act	6
	Uin	用户的 91Uin
	SessionId	用户的客户端 SessionId
	PageNo	第几页页号(从 1 开始)
	PageSize	每页数量(5 的倍数)
	Sign	参数值与 AppKey 的 MD5 值 String.Format("{0}{1}{2}{3}{4}{5}{6}", AppId, Act, Uin, SessionId, PageNo, PageSize, AppKey).HashToMD5Hex();
	AppId=100010&Act=6&Uin=11323356&PageNo=1&PageSize=5&SessionID=5e9c3844563640daa9b9d4846031bbd4&Sign=e22aa45b90ae4aaf5d63e1d0799d10a2	
91 服务端处理	1. 验证 Sign 是否有效 2. 验证 SessionId 是否有效 3. 获取用户的当前应用的好友列表 4. 返回结果给应用服务器	
91 服务器返回应用服务器参数	ErrorCode	错误码(0=失败，1=成功，2= AppId 无效，3= Act 无效，4=参数无效，5= Sign 无效)
	ErrorDesc	错误描述
	FriendList	好友列表 每个好友项格式为：Uin,昵称,积分,心情,CheckSum,在线状态 好友项之间以分号做分隔符，最后一个好友不用加分隔符 在线状态：0=未知，1=在线，2=离线，目前在线状态统一为 0
	{"FriendList":"11270233,风云之雄霸天下 kkkkkkkkkkk,0,,ffffffffffffffffffffffffffffffff,0;704458214,wyl1234567,0,,ffffffffffffffffffffffffffffffff,0;11697680,lbj05,0,,795e9fc4c0cd7ba82f161db62d8e8179,0;11673967,机友 11673967201409,0,,ffffffffffffffffffffffffffffffff,0;11268566,雄霸	

	letgo,0,,ffffffffffffffffffffffffffffffff,0","TotalCount":"10","ErrorCode":"1","ErrorDesc":"成功"}
备注	

五、 91 服务器通知给应用服务器接口

正式外网：由各应用服务商需要在开发者后台配置应用服务器的回传地址

1、 接收支付购买结果(Act=1)

开发者需要在后台配置支付回传地址。

功能描述	接收由 91 移动开发平台服务器发送给各个应用服务商的支付购买结果	
91 服务器通知给应用服务器参数	**参数名**	**说明(**没有说明可选的参数，为必传参数**)**
	AppId	应用 ID
	Act	1
	ProductName	应用名称
	ConsumeStreamId	消费流水号
	CooOrderSerial	商户订单号
	Uin	91 账号 ID
	GoodsId	商品 ID
	GoodsInfo	商品名称
	GoodsCount	商品数量
	OriginalMoney	原始总价(格式：0.00)
	OrderMoney	实际总价(格式：0.00)
	Note	即支付描述（客户端 API 参数中的 payDescription 字段） 购买时客户端应用通过 API 传入，原样返回给应用服务器 开发者可以利用该字段，定义自己的扩展数据。例如区分游戏服务器
	PayStatus	支付状态：0=失败，1=成功
	CreateTime	创建时间(yyyy-MM-dd HH:mm:ss)
	Sign	以上参数的 MD5 值，其中 AppKey 为 91SNS 平台分配的应用密钥 String.Format("{0}{1}{2}{3}{4}{5}{6}{7}{8}{9:0.00}{10:0.00}{11}{12}{13:yyyy-MM-dd HH:mm:ss}{14}", AppId,Act, ProductName,ConsumeStreamId, CooOrderSerial, Uin, GoodsId, GoodsInfo, GoodsCount, OriginalMoney, OrderMoney, Note, PayStatus, CreateTime, AppKey).HashToMD5Hex()

	AppId=100010&Act=1&ProductName=%e6%98%9f%e9%99%85%e8%bf%b7%e8%88%aaDemo&ConsumeStreamId=1-10001-20101214233421-1-6422&CooOrderSerial=a258337465ff4e85b78b2c23d7046098&Uin=155451276&GoodsId=80370&GoodsInfo=X1000%e6%88%98%e6%96%97%e6%9c%ba&GoodsCount=1&OriginalMoney=0.01&OrderMoney=0.01&Note=%e6%88%98%e6%96%97%e6%9c%ba&PayStatus=1&CreateTime=2010-12-14 23:34:21&Sign=2667575c8582b52a5fe5aa09141b076b	
应用服务端处理	1. 验证 Sign 是否有效 2. 读取订单数据 3. 返回结果给 91 移动开发平台服务器 注：验证 Sign 时，所拼接的字符串，需要用 91 服务器传回的原样数据，否则 Sign 会不一致。OriginalMoney 和 OrderMoney 拼接时需要保留两位小数，例如 15 个豆用 15.00	
应用服务端返回给 91 服务器参数	ErrorCode	错误码(0=失败，1=成功（已处理过的，也当作成功返回），2= AppId 无效，3= Act 无效，4=参数无效，5= Sign 无效，其他的应用自己定义，并在错误描述中体现)
	ErrorDesc	错误描述
	例如 {"ErrorCode":"0","ErrorDesc":"接收失败"} {"ErrorCode":"1","ErrorDesc":"接收成功"}	
备注	对于用户完成正常支付后的通知，移动开发平台服务器会间隔时间，向应用服务器发送支付结果通知，直到应用服务器确认接收成功。 由于网络等原因，可能存在同一笔订单，重复通知，业务方服务器若收到重复通知，需要回复成功（ErrorCode=1），避免下次再次通知。	

2、 接收平台事件通知

平台事件通知，是以 91 服务器发生的相关事件通知到开发者应用服务器，开发者可以根据需要进行相关的处理。

例如： 小明成功邀请了一个小花来玩当前应用，就会通知到开发者应用服务器，开发者可对小明进行虚拟道具的奖励。

开发者需要在后台配置事件的通知地址，以及选择需要接收的通知类型。系统默认为不通知。

功能描述	接收由 91 移动开发平台服务器发送给各个应用服务商的平台事件通知	
91 服务器通知给应用服务器	用 HTTP POST 方式向接收地址发送数据	
Url 参数	**参数名**	**说明(**没有说明可选的参数，为必传参数**)**
	AppId	应用 Id

<table>
<tr><td></td><td>CreateTime</td><td>创建时间(yyyy-MM-dd HH:mm:ss)</td></tr>
<tr><td></td><td>Sign</td><td>以上参数的 MD5 值，其中 AppKey 为 91SNS 平台分配的应用密钥
String.Format("{0}{1:yyyy-MM-dd HH:mm:ss}{2}", AppId, CreateTime, AppKey).HashToMD5Hex()</td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td></td><td colspan="2">示例:
ReceiveNotify.ashx?AppId=100010& CreateTime=2010-12-14 23:34:21&Sign=2667575c8582b52a5fe5aa09141b076b</td></tr>
<tr><td>POST 参数</td><td>参数名</td><td>说明(没有说明可选的参数，为必传参数)</td></tr>
<tr><td></td><td>NotifyList</td><td>通知记录集合,请参阅下方<事件通知类型说明></td></tr>
<tr><td></td><td colspan="2">示例:
[
{
"Id": 12,
"Type": 6,
"AppId": 100011,
"Uin": 11657374,
"NickName": "aabbcc",
"CreateTime": "2011-07-12 14:38:14"
} ,
{
"Id": 623,
"Type": 6,
"AppId": 100011,
"Uin": 11657374,
"NickName": "aabbcc",
"LoginAppId": 100010,
"CreateTime": "2011-07-12 14:38:14"
} ,
…
]</td></tr>
<tr><td>应用服务端处理</td><td colspan="2">1. 验证 Sign 是否有效
2. 读取事件通知数据
3. 返回结果给 91 移动开发平台服务器
注：验证 Sign 时，所拼接的字符串，需要用 91 服务器传回的原样数据，否则 Sign 会不一致。</td></tr>
<tr><td>应用服务端返回给 91 服务器参数</td><td>ErrorCode</td><td>错误码(0=失败，1=成功（已处理过的，也当作成功返回），2= AppId 无效，3= Act 无效，4=参数无效，5= Sign 无效，其他的应用自己定义，并在错误描述中体现)</td></tr>
<tr><td></td><td>ErrorDesc</td><td>错误描述</td></tr>
<tr><td></td><td colspan="2">例如
{"ErrorCode":"0","ErrorDesc":"接收失败"}
{"ErrorCode":"1","ErrorDesc":"接收成功"}</td></tr>
<tr><td>备注</td><td colspan="2">开发者服务端无响应或返回 ErrorCode=0 时，服务端会尝试重新发送，最多尝试发送</td></tr>
</table>

	5 次。

1) 事件通知类型说明

通知类型	参数名	说明
邀请 91 好友玩当前应用(Type=1)	Id	编号
	Type	通知类型
	AppId	应用 ID
	Uin	邀请者的用户 ID
	NickName	邀请者的昵称
	CreateTime	创建时间,格式“yyyy-MM-dd HH:mm:ss”
成功邀请 91 好友玩当前应用(Type=2)	Id	编号
	Type	通知类型
	AppId	应用 ID
	Uin	用户 ID
	NickName	昵称
	InviteUin	被邀请的用户 ID
	InviteNickName	被邀请的
	CreateTime	创建时间,格式“yyyy-MM-dd HH:mm:ss”
邀请第三方平台好友玩当前应用 (Type=3)	Id	编号
	Type	通知类型
	AppId	应用 ID
	Uin	用户 ID
	NickName	昵称
	CreateTime	创建时间,格式“yyyy-MM-dd HH:mm:ss”
成功邀请第三方平台好友玩当前应用(Type=4)	Id	编号
	Type	通知类型
	AppId	应用 ID
	Uin	用户 ID
	NickName	昵称
	InviteUin	被邀请的用户 ID
	CreateTime	创建时间,格式“yyyy-MM-dd HH:mm:ss”
分享到第三方平台通知 (Type=5)	Id	编号
	Type	通知类型
	AppId	应用 ID
	Uin	用户 ID
	SharePlatformType	分享平台类型。1：新浪微博；3：腾讯微博；4：人人微博；若同时分享至多平台，以逗号分隔。

		示例：1，4 表示同时分享至新浪微博和人人微博。
	NickName	昵称
	CreateTime	创建时间,格式“yyyy-MM-dd HH:mm:ss”
用户首次玩其它款游戏（当前应用除外）通知(Type=6)	Id	编号
	Type	通知类型
	AppId	应用 ID
	Uin	用户 ID
	NickName	昵称
	LoginAppId	*首次登录的其它应用 ID*
	CreateTime	创建时间,格式“yyyy-MM-dd HH:mm:ss”

3、 接收兑换码兑换物品结果(Act=3)

应用开发者的接口地址同 Act=1。

功能描述	接收由 91 移动开发平台服务器发送给各个应用服务商的兑换码兑换物品结果	
91 服务器通知给应用服务器参数	**参数名**	**说明(**没有说明可选的参数，为必传参数**)**
	AppId	应用 ID
	Act	3
	ProductName	应用名称
	Code	兑换码
	Is91	是否 91 帐号:　0=否；1=是
	UserId	用户 Id(91 帐号的 Uin 或其他系统的用户 Id ,长度限制 40 个字符)
	GoodsId	兑换物品 ID(此参数可选，默认为空，根据开发者后台配置，有即返回，无则不返回)
	GoodsCount	兑换物品数量(此参数可选，默认为 0，如果没有 GoodId 参数返回，则此参数也不返回)
	Sign	以上参数的 MD5 值,其中 AppKey 为 91SNS 平台分配的应用密钥 String.Format("{0}{1}{2}{3}{4}{5}{6}{7}{8}",AppId,Act,ProductName, Code, Is91, UserId, GoodsId, GoodsCount, AppKey).HashToMD5Hex()
应用服务端处理	1. 验证 Sign 是否有效 2. 读取相关数据，给用户发放添加虚拟物品 3. 返回结果给通用平台服务器	
应用服务端返回给 91 服务器参数	ErrorCode	错误码(0=失败，1=成功（已处理过的，也当作成功返回），2= AppId 无效，3= Act 无效，4=参数无效，5= Sign 无效，其他的应用自己定义，并在错误描述中体现)

	ErrorDesc	错误描述
	例如 {"ErrorCode":"0","ErrorDesc":"接收失败"} {"ErrorCode":"1","ErrorDesc":"接收成功"}	
备注		

4、 接收软件互推物品奖励通知(Act=4)

应用开发者的接口地址同 Act=4。

功能描述	接收由 91 移动开发平台服务器发送给各个应用服务商的软件互推功能中用户得到的奖励物品通知	
91 服务器通知给应用服务器参数	**参数名**	**说明(**没有说明可选的参数，为必传参数**)**
	AppId	应用 ID
	Act	4
	PopAppId	被推广的应用 ID
	PopAppName	被推广的应用名称
	Uin	获得奖励的用户的 91 帐号 Id
	GoodsId	奖励物品 ID(根据开发者后台配置)
	GoodsName	奖励物品名称(根据开发者后台配置)
	AwardCount	奖励物品数量(根据开发者后台配置)
	ActiveTime	用户成功激活时间,格式“yyyy-MM-dd HH:mm:ss”
	Sign	以上参数的 MD5 值，其中 AppKey 为 91SNS 平台分配的应用密钥 String.Format("{0}{1}{2}{3}{4}{5}{6}{7}{8}{9}{10}",AppId,Act, PopAppId, PopAppName, Uin, GoodsId, GoodsName, AwardCount, ActiveTime, AppKey).HashToMD5Hex()
应用服务端处理	1. 验证 Sign 是否有效 2. 读取相关数据，给用户发放奖励物品 3. 返回结果给通用平台服务器	
应用服务端返回给 91 服务器参数	ErrorCode	错误码(0=失败，1=成功（已处理过的，也当作成功返回），2= AppId 无效，3= Act 无效，4=参数无效，5= Sign 无效，其他的应用自己定义，并在错误描述中体现)
	ErrorDesc	错误描述
	例如 {"ErrorCode":"0","ErrorDesc":"接收失败"} {"ErrorCode":"1","ErrorDesc":"接收成功"}	
备注		

六、 附录

1、 HashToMD5Hex ()算法实现

测试字符串：最新的测试结果表明，IE9 的预览版本已经完全支持 W3C Web Standards HTML5 和 CSS3。

签名结果： HashToMD5Hex("最新的测试结果表明，IE9 的预览版本已经完全支持 W3C Web Standards HTML5 和 CSS3。") = "fc17dd9ac671a43f880dae37ecfc2c78"

如果大家在各平台实现签名算法时，可使用上面的字符串进行签名验证，如果签名结果为"fc17dd9ac671a43f880dae37ecfc2c78" 说明签名算法是正确的。

在签名算法一致的情况下，需要确认，您用于拼接的相关参数是否一致。
特别注意以下类型与格式：

originalmoney	精度：0.01
ordermoney	精度：0.01
createtime	格式 ：yyyy-MM-dd HH:mm:ss

1）C#实现

```
sourceStr ：为各接口要求的参数拼接成的字符串
byte[] Bytes = Encoding.UTF8.GetBytes(sourceStr);
using (MD5CryptoServiceProvider md5 = new MD5CryptoServiceProvider())
{
  byte[] result = md5.ComputeHash(Bytes);

  StringBuilder sBuilder = new StringBuilder();
  for (int i = 0; i < result.Length; i++)
    sBuilder.Append(result[i].ToString("x2"));
  return sBuilder.ToString();
}
```

2）Python 实现

```
sourceStr ：为各接口要求的参数拼接成的字符串
#coding=utf-8
import hashlib

signStr = hashlib.md5()
signStr.update(sourceStr.encode('utf-8'))
return signStr.hexdigest()
```

3）Java 实现

```
sourceStr ：为各接口要求的参数拼接成的字符串
import java.io.UnsupportedEncodingException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import com.sun.org.apache.xerces.internal.impl.dv.util.HexBin;

try {
	byte[] bytes = sourceStr.getBytes("utf-8");
	MessageDigest md5 = MessageDigest.getInstance("MD5");
	md5.update(bytes);
	byte[] md5Byte = md5.digest();
	if(md5Byte != null){
		signStr = HexBin.encode(md5Byte);
	}
} catch (NoSuchAlgorithmException e) {
	e.printStackTrace();
} catch (UnsupportedEncodingException e) {
	e.printStackTrace();
}
return signStr;
```