



**中国农业银行**  
AGRICULTURAL BANK OF CHINA

# **中国农业银行网上支付平台**

## **商户接口编程指南**

### **ASP.NET2.0 Edition**

### **V1.0.7**

## 修订历史纪录

日期	版本	说明	作者
2009-06-20	V1.0	初版	网上支付平台开发小组
2010-08-23	V1.0.1	为防钓鱼支付在订单信息中添加客户 IP 地址 BuyIP，支持手机 WAP 支付和电话支付请求 网上支付平台开发小组	网上支付平台开发小组
2010-11-25	V1.0.2	补充浏览器提交模式和保险支付、网上付款	网上支付平台开发小组
2011-8-19	V1.0.3	修改委托扣款签约接口，增加农行卡类型支持	网上支付平台开发小组
2011-10-24	V1.0.4	增加委托扣款签约结果查询	网上支付平台开发小组
2011-11-03	V1.0.5	为支付请求、支付请求--页面提交、保险直销支付、 保险直销支付--页面提交、委托扣款单笔代扣、 委托扣款批量代扣添加订单有效期属性，支付请求的 remark 从 200 扩展到 500	网上支付平台开发小组
2012-08-06	V1.0.6	新增支付类型（基于第三方的跨行支付方式）	网上支付平台开发小组
2012-10-31	V1.0.7	新增接口包 64 位系统支持	网上支付平台开发小组

# 目录

<b>ASP.NET2.0 EDITION.....</b>	<b>1</b>
<b>V1.0.7 .....</b>	<b>1</b>
<b>1. 简介 .....</b>	<b>8</b>
1.1 目的.....	8
1.2 功能描述.....	8
1.3 总体架构图.....	8
<b>2. 接口开发软件包说明 .....</b>	<b>9</b>
<b>3. 安装步骤 .....</b>	<b>11</b>
3.1 安装前检查.....	11
3.2 接口开发软件包安装.....	11
3.3 接口开发软件包配置.....	11
3.4 配置测试应用.....	12
3.5 移除接口开发软件包.....	12
<b>4. 配置文件说明 .....</b>	<b>13</b>
4.1 单一商户配置说明.....	13
4.2 多商户配置说明.....	15
<b>5. 交易说明 .....</b>	<b>16</b>
5.1 交易流程.....	16
5.1.1 支付交易.....	16
5.1.2 确保支付结果正确送达商户网站的措施.....	19
5.1.3 其它交易.....	21

5.2	交易使用时机.....	22
5.3	支付请求.....	23
5.3.1	方式 1: 通过与农行服务器建立连接访问农行 b2c 支付平台服务.....	23
5.3.2	方式 2: 通过页面传参提交表单方式访问农行 b2c 支付平台服务.....	27
5.4	两种接收支付结果方式的区分.....	28
5.4.1	通过显示给消费者的支付结果接收页面通知商户.....	28
5.4.2	通过支付平台服务器通知商户.....	29
5.4.3	区别.....	31
5.5	支付结果接收页面.....	32
5.6	退货请求.....	34
5.7	订单查询.....	36
5.8	交易对账单下载.....	38
5.9	多商户交易方式.....	40
5.10	指定日期指定时间段交易对账单下载.....	40
5.11	农行卡身份验证交易请求.....	42
5.12	卡验证结果接收页面.....	42
5.13	身份验证交易请求.....	43
5.14	身份验证结果接收页面.....	44
5.15	批量退款发送请求.....	45
5.16	批量退款结果查询请求.....	46
5.17	委托扣款签约请求.....	47
5.18	委托扣款解约请求.....	49
<b>4、</b>	<b>使用交易结果对象的 ISSUCCESS()方法辨别交易是否成功 .....</b>	<b>49</b>
5.19	委托扣款单笔代扣请求.....	50

5.20	委托扣款批量请求.....	51
5.21	委托扣款批量结果查询.....	52
5.22	委托扣款签约结果查询.....	54
5.23	贷记卡交易对账单下载.....	56
5.24	保险直销支付请求.....	57
5.24.1	方式 1：通过与农行服务器建立连接访问农行 b2c 支付平台服务.....	57
5.24.2	方式 2：通过页面传参提交表单方式访问农行 b2c 支付平台服务.....	61
5.25	网上付款信息发送请求.....	62
5.26	网上付款交易结果查询请求.....	63
5.27	网上付款银行卡状态验证请求.....	65
6.	附录一、程序范例.....	67
A、	配置文件 WEB.CONFIG 范例 .....	67
B、	支付请求范例 .....	73
	<b>RESPONSE.WRITE (STRMESSAGE) ; .....</b>	<b>74</b>
C、	支付结果接收范例 .....	76
	<b>RESPONSE.WRITE (STRMESSAGE.TOSTRING()) ; .....</b>	<b>77</b>
D、	从服务器直接接收支付结果页面范例 .....	78
E、	退货交易范例.....	80
	<b>RESPONSE.WRITE (STRMESSAGE.TOSTRING()) ; .....</b>	<b>81</b>
F、	订单查询交易范例.....	82
	<b>RESPONSE.WRITE (STRMESSAGE.TOSTRING()) ; .....</b>	<b>84</b>
G、	交易对账单下载范例 .....	85
	<b>RESPONSE.WRITE (STRMESSAGE.TOSTRING()) ; .....</b>	<b>86</b>

H、指定时间段交易对账单下载 .....	86
<b>RESPONSE.WRITE (STRMESSAGE.TOSTRING()) ; .....</b>	<b>88</b>
I、身份验证请求范例.....	88
J、身份验证结果接收范例.....	89
<b>RESPONSE.WRITE (STRMESSAGE.TOSTRING()) ; .....</b>	<b>89</b>
K、批量退款发送请求范例 .....	89
<b>STRINGBUILDER STRMESSAGE = NEW STRINGBUILDER(""); ; .....</b>	<b>89</b>
<b>RESPONSE.WRITE (STRMESSAGE.TOSTRING()) ; .....</b>	<b>92</b>
L、批量退款交易结果查询请求范例 .....	92
M、委托扣款签约交易请求范例.....	94
<b>RESPONSE.WRITE (STRMESSAGE.TOSTRING()) ; .....</b>	<b>95</b>
N、委托扣款解约交易请求范例 .....	95
<b>RESPONSE.WRITE (STRMESSAGE.TOSTRING()) ; .....</b>	<b>96</b>
O、委托扣款单笔代扣交易请求范例 .....	96
<b>RESPONSE.WRITE (STRMESSAGE.TOSTRING()) ; .....</b>	<b>97</b>
P、委托扣款批量交易请求范例 .....	97
<b>STRINGBUILDER STRMESSAGE = NEW STRINGBUILDER(""); ; .....</b>	<b>97</b>
Q、委托扣款批量结果查询交易请求范例 .....	101
<b>RESPONSE.WRITE (STRMESSAGE.TOSTRING()) ; .....</b>	<b>103</b>
R、贷记卡对账单下载交易请求范例.....	103
<b>RESPONSE.WRITE (STRMESSAGE.TOSTRING()) ; .....</b>	<b>104</b>
S、商户通过页面传参数提交表单方式访问农行 B2C 服务范例程序 .....	104

T、商户支付请求验证失败结果接收范例 .....	107
U、商户理财验证失败结果接收范例 .....	108
V、保险直销支付请求范例 .....	109
W、页面提交保险支付请求范例 .....	113
X、网上付款信息发送请求范例 .....	118
Y、网上付款交易结果查询请求范例 .....	121
Z、网上付款银行卡状态验证请求范例 .....	124
AA、委托扣款签约结果查询请求范例 .....	125
<b>7. 附录二、响应码一览表 .....</b>	<b>127</b>
<b>8. 附录三、TRUSTPAY CLIENT API .....</b>	<b>130</b>
COM.HITRUST.TRUSTPAY.CLIENT.TrxRESPONSE .....	130
COM.HITRUST.TRUSTPAY.CLIENT.B2C.ORDER .....	131
COM.HITRUST.TRUSTPAY.CLIENT.B2C.ORDERITEM .....	134
COM.HITRUST.TRUSTPAY.CLIENT.B2C.PAYMENTREQUEST .....	135
COM.HITRUST.TRUSTPAY.CLIENT.B2C.PAYMENTRESULT .....	138
COM.HITRUST.TRUSTPAY.CLIENT.B2C.QUERYORDERREQUEST .....	140
COM.HITRUST.TRUSTPAY.CLIENT.B2C.REFUNDREQUEST .....	142
COM.HITRUST.TRUSTPAY.CLIENT.B2C.SETTLEREQUEST .....	144
COM.HITRUST.TRUSTPAY.CLIENT.B2C.SETTLEFILE .....	145
COM.HITRUST.TRUSTPAY.CLIENT.B2C.IDENTITYVERIFYREQUEST .....	146
COM.HITRUST.TRUSTPAY.CLIENT.B2C.BATCH .....	148
COM.HITRUST.TRUSTPAY.CLIENT.B2C.OVERDUEBATCH .....	149
COM.HITRUST.TRUSTPAY.CLIENT.B2C.BATCHSENDREQUEST .....	152
COM.HITRUST.TRUSTPAY.CLIENT.B2C.QUERYBATCHREQUEST .....	153
COM.HITRUST.TRUSTPAY.CLIENT.B2C.OVERDUEREFUNDREQUEST .....	153
COM.HITRUST.TRUSTPAY.CLIENT.B2C.QUERYOVERDUEREFUNDREQUEST .....	155
COM.HITRUST.TRUSTPAY.CLIENT.B2C.B2CAGENTSIGNRESULT .....	156
COM.HITRUST.TRUSTPAY.CLIENT.B2C.B2CAGENTSIGNCONTRACTREQUEST .....	157
COM.HITRUST.TRUSTPAY.CLIENT.B2C.B2CAGENTUNSIGNCONTRACTREQUEST .....	159
COM.HITRUST.TRUSTPAY.CLIENT.B2C.B2CAGENTPAYMENTREQUEST .....	161
COM.HITRUST.TRUSTPAY.CLIENT.B2C.AGENTBATCH .....	162
COM.HITRUST.TRUSTPAY.CLIENT.B2C.AGENTSIGN .....	164
COM.HITRUST.TRUSTPAY.CLIENT.B2C.QUERYAGENTSIGNREQUEST .....	165
COM.HITRUST.TRUSTPAY.CLIENT.B2C.AGENTBATCHDETAIL .....	166
COM.HITRUST.TRUSTPAY.CLIENT.B2C.B2CAGENTBATCHREQUEST .....	168
COM.HITRUST.TRUSTPAY.CLIENT.B2C.B2CAGENTBATCHQUERYREQUEST .....	169
COM.HITRUST.TRUSTPAY.CLIENT.B2C.INSURE .....	169
COM.HITRUST.TRUSTPAY.CLIENT.B2C.INSUREORDER .....	171

COM.HITRUST.TRUSTPAY.CLIENT.B2C.INSUREORDERITEM .....	171
COM.HITRUST.TRUSTPAY.CLIENT.B2C.INSUREUSER .....	172
COM.HITRUST.TRUSTPAY.CLIENT.B2C.ONLINEREMITREQUEST .....	173
COM.HITRUST.TRUSTPAY.CLIENT.B2C.ONLINERMTCARDVERIFYREQUEST .....	174
COM.HITRUST.TRUSTPAY.CLIENT.B2C.ONLINERMTQUERYRESULTREQUEST .....	175
COM.HITRUST.TRUSTPAY.CLIENT.B2C.ONLINERMTBATCH .....	176
COM.HITRUST.TRUSTPAY.CLIENT.B2C.QUERYRESULT .....	177
COM.HITRUST.TRUSTPAY.CLIENT.B2C.QUERYRESULTITEM .....	179
<b>QUERYRESULTITEM 默认构造函数.....</b>	<b>180</b>
COM.HITRUST.TRUSTPAY.CLIENT.B2C.AGENTSIGN .....	180
COM.HITRUST.TRUSTPAY.CLIENT.B2C.QUERYAGENTSIGNREQUEST.....	181
<b>9. 附录四、常见问题及解决办法 .....</b>	<b>183</b>
1. 无法连接.....	183
2. WSE3.0 安装不上 .....	183
3. 无法读取证书文档.....	183
4. 返回报文签名验证失败.....	184
5. 类型初始值设定引发异常 .....	184
6. 服务器换 IP 后的通知问题 .....	185



## 1. 简介

### 1.1 目的

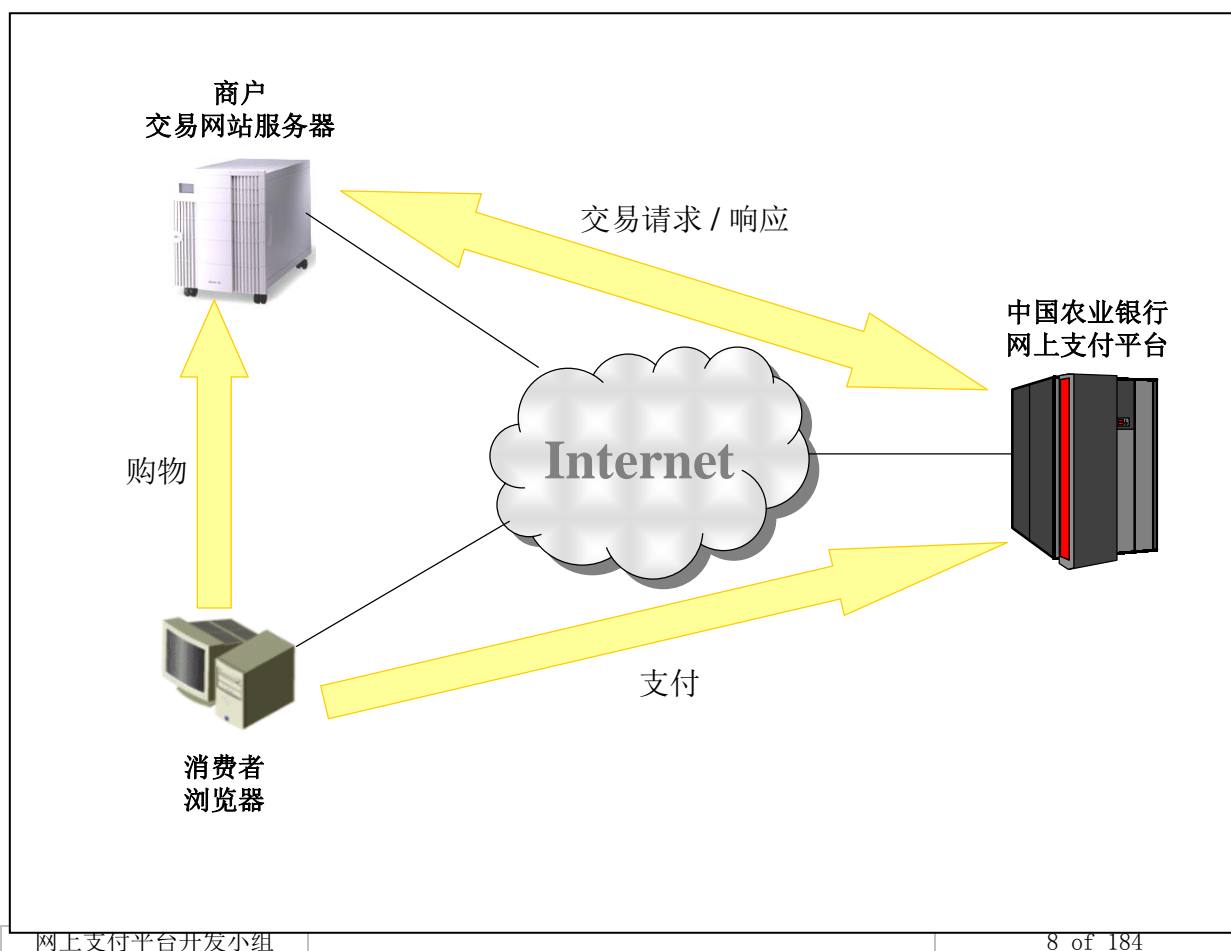
提供商户端交易网站通过中国农业银行网上支付平台提供的商户端开发软件包实现功能的编程指南。

### 1.2 功能描述

按照本编程指南所描述的标准，商户交易网站可以呼叫调用，支持功能包括支付请求、退货、订单查询、交易对账单下载，并且具备接收网上支付平台支付结果响应的功能。

接口采用电子证书的方式来保证商户与网上支付平台间的身份验证、中间信息传递的完整性，以便进行电子商务安全当中非常重要的交易身份辨识、不可抵赖、防止篡改等功能。

### 1.3 总体架构图



## 2. 接口开发软件包说明

银行提供的接口开发软件包 TrustPayClient-ASP.NET2.0-Vx.x.x.zip (x.x.x 为接口开发软件包的版本号) 包含下列文档, 此接口包同时支持 32 位及 64 位操作系统。

文件名称	说明
/docs/ 商 户 接 口 编 程 指 南 -ASP.NET2.0-Edition-Vx.x.x.pdf	本文件
/docs/农行网上支付平台-商户使用手册.pdf	商户使用手册
/docs/农行网上支付平台-USB Key 安装及使用手册.pdf	USB Key 安装及使用手册
/docs/农行网上支付平台-商户端软件包-FAQ.pdf	商户开发中常见问题解答
/demo/Bin/TrustPayClient.dll	商户开发接口包
/demo/Web.Config	接口配置文件
/demo/Merchant.html	接口范例首页
/demo/B2CAgentBatch.aspx	委托扣款批量范例程序
/demo/B2CAgentPayment.aspx	委托扣款单笔代扣范例程序
/demo/B2CAgentSignContract.aspx	委托扣款签约范例程序
/demo/B2CAgentSignResult.aspx	委托扣款签约解约结果接收范例程序
/demo/B2CAgentUnsignContract.aspx	委托扣款解约范例程序
/demo/B2CQueryAgentBatch.aspx	委托扣款批量处理结果结果查询范例程序
/demo/CustomerPage.aspx	服务器支付结果接收范例程序
/demo/IdentityVerify.aspx	身份验证范例程序
/demo/IdentityVerifyResult.aspx	身份验证结果接收范例程序
/demo/MerchantBatchSend.aspx	退款批量文件发送范例程序
/demo/MerchantCreditTrxSettle.aspx	贷记卡交易对账单下载范例程序
/demo/MerchantFailure.aspx	服务器支付结果接收失败范例程序
/demo/MerchantOverdueRefund.aspx	订单批量退款范例程序
/demo/MerchantPayment.aspx	支付请求交易范例程序
/demo/MerchantQueryBatch.aspx	查询批量处理结果范例程序

/demo/MerchantQueryOrder.aspx	订单查询交易范例程序
/demo/MerchantQueryOverdueRefund.aspx	查询批量退款处理结果范例程序
/demo/MerchantRefund.aspx	退货支付交易范例程序
/demo/MerchantResult.aspx	支付结果接收范例程序
/demo/MerchantTrxSettle.aspx	交易对账单下载范例程序
/demo/MerchantTrxSettleByHour.aspx	指定时间段交易对账单下载范例程序
/demo/ReciveServerPage.aspx	直接接收服务器支付结果范例程序
/demo/MerchantPaymentInsure.aspx	保险支付请求范例程序
/demo/IE/MerchantIEFailure.aspx	浏览器模式支付验证失败处理范例程序
/demo/IE/MerchantIssueFailure.aspx	浏览器模式理财身份验证失败处理范例程序
/demo/IE/MerchantPaymentIE.aspx	浏览器模式提交支付请求范例程序
/demo/IE/MerchantPaymentInsureIE.aspx	浏览器模式提交保险支付请求范例程序
/demo/SendOnlineRemitInfo.aspx	网上付款信息发送请求范例程序
/demo/OnlineRmtQueryResult.aspx	网上付款交易结果查询请求范例程序
/demo/OnlineRmtCardStatusQuery.aspx	网上付款银行卡状态查询请求范例程序
/cert/abc.truststore	农行根证书
/cert/trustpay.cer	网上支付平台证书
/ABCIcon/*.jpg	用于商户在自行开发的页面上，如果有指向农行的图片链接，请使用这些图片做为农行标识。注意：图片的整体尺寸可以根据需要进行缩放，但是图片的内容和比例大小不能修改。

### 3. 安装步骤

#### 3.1 安装前检查

1、本接口软件包采用 Microsoft .NET Framework 2.0 标准。

2、请确定服务器已经安装了下列软件：

Microsoft .NET Framework 2.0

Microsoft Web Services Enhancements 3.0 for Microsoft .NET（下载网址

<http://www.microsoft.com/en-us/download/details.aspx?id=14089>）

#### 3.2 接口开发软件包安装

1、如果服务器上已经安装过前一个版本的接口开发软件包，请参考【3.5 移除接口开发软件包】的说明，移除前一个版本的接口开发软件包。

2、将银行提供的接口开发软件包 TrustPayClient-ASP.NET2.0-Vx.x.x.zip 解压缩到商户自定的安装目录中。

3、请参考《农行网上支付平台 -商户使用手册 V1.0》登录网上支付平台下载商户交易证书，并将商户交易证书保存到服务器的硬盘或签名服务器中。

#### 3.3 接口开发软件包配置

1、打开 Demo 中的 Web.Config 文件，依照银行提供的信息设定文件中相对应的参数。详细配置文件的说明请参考下一章的说明。

### 3.4 配置测试应用

- 1、打开 IIS，建立虚拟目录，指向 *接口软件包安装目录\demo*。
- 2、开启浏览器进入 <http://your.server.name/your.virtual.directory/Merchant.html>，确定接口软件包是否已正确安装及配置。

### 3.5 移除接口开发软件包

如果想移除服务器上的接口开发软件包安装，或者想升级最新版本的接口开发软件包，请按照下列步骤进行接口开发软件包的移除。

- 1、删除接口软件包安装目录。

## 4. 配置文件说明

本接口软件包支持多商户的配置，通常一般商户只会使用到单一商户的功能。如果商户向农行申请了多个商户号，同时这些商户的交易都是部署在同一部应用服务器上，这时你才需要使用到多商户配置功能。

### 4.1 单一商户配置说明

配置段	参数名称	数值类型	说明
网上支付平台 系统配置段  *请依照银行的 指示设定	TrustPayConnectMethod	字符串	网上支付平台通讯方式 <b>http:</b> 使用 HTTP 通讯方式 <b>https:</b> 使用 HTTPS 通讯方式
	TrustPayServerName	字符串	网上支付平台服务器名 可以使用服务器的域名或服务器的 IP 地址
	TrustPayServerPort	数字	网上支付平台交易端口
	TrustPayNewLine	数字	网上支付平台接口特性 <b>1 或 2</b>
	TrustPayTrxURL	字符串	网上支付平台交易网址
	TrustPayCertFile	字符串	网上支付平台证书
	TrustStoreFile	字符串	农行根证书文件
	TrustStorePassword	字符串	农行根证书文件密码
	TrustPayIETrxURL	字符串	商户通过浏览器提交网上支付平台交易网址
商户资料段	MerchantID	字符串	商户编号
商户系统配置段	MerchantErrorURL	字符串	商户通过浏览器提交网上支付平台交易失败网址
	EnableLog	字符串	日志开关。 <b>true:</b> 打开日志功能。 <b>false:</b> 关闭日志功能
	log4net/file	字符串	日志文件存放绝对位置。
	MerchantKeyStoreType	数字	证书储存媒体 <b>0: File</b> <b>1: 硬件签名服务器</b>

	MerchantCertFile	字符串	商户证书储存目录档名 当 KeyStoreType=0 时，必须设定。 必须为 PKCS#12 的文件格式。
	MerchantCertPassword	字符串	商户私钥加密密码 当 KeyStoreType=0 时，必须设定。
	SignServerIP	字符串	签名服务器 IP 地址 当 KeyStoreType=1 时，必须设定。
	SignServerPort	数字	签名服务器端口 当 KeyStoreType=1 时，必须设定。
	SignServerPassword	字符串	签名服务器密码 当 KeyStoreType=1 时，必须设定。

具体可以参考[附录：配置文件 web.config 范例](#)

## 4.2 多商户配置说明

多商户的配置除下列参数外，其它的参数配置与单一商户配置相同。

参数名称	数值 类型	说明及范例
MerchantID	字符串	此参数可以配置多个商户号，各个商户号间以逗号分隔开。 范例： MerchantID=103100070113810,123456789012345,103456464564564
MerchantCertFile	字符串	此参数可以配置多个商户证书储存目录档名，各个档名间以逗号分隔开。 范例： MerchantCertFile =C:\cert\3810.pfx,C:\cert\2345.pfx,C:\cert\4564.pfx
MerchantCertPassword	字符串	此参数可以配置多个商户私钥加密密码，各个密码间以逗号分隔开。 范例： MerchantCertPassword =11111111,22222222,12345678

请特别注意此三个参数各个参数值的顺序，相同顺序号的参数值代表一个商户的配置。以上述的范例来说：商户号 103100070113810 的证书文件为 C:\cert\3810.pfx，而要打开此证书所需要的密码为 11111111。



## 5. 交易说明

农行网上支付平台商户接口采用面向对象的方式设计，商户在交易的过程中会需要使用到各个不同的类来完成所需要的交易。本章的说明着重在流程的说明，类的详细说明请参考《附录三、TrustPay Client API》。

本章绝大部分的说明适用于单一商户配置状况，多商户配置需要特别注意的地方请参考《5.10 多商户交易方式》。

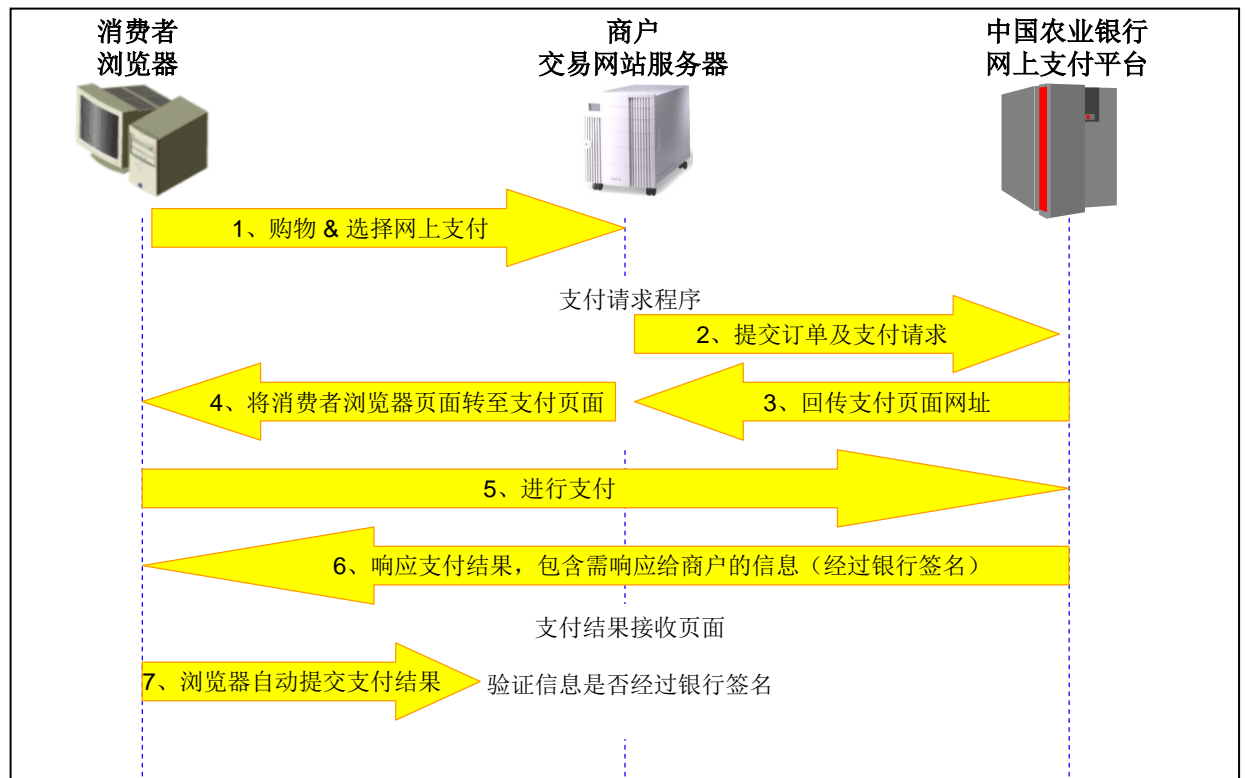
### 5.1 交易流程

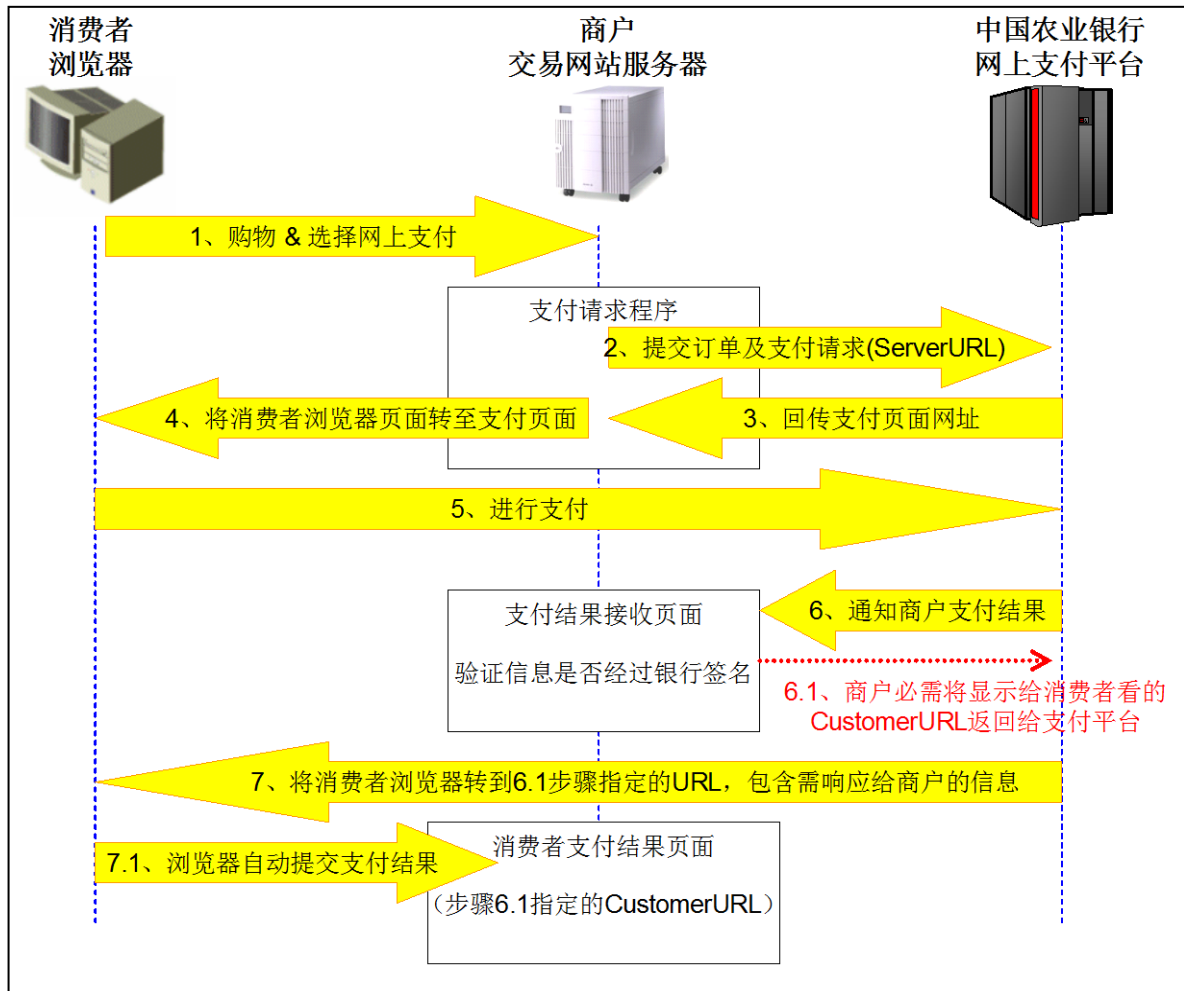
本节将说明商户交易平台如何与网上支付平台通信，来完成交易的过程。

#### 5.1.1 支付交易

支付交易因为需要三方的配合（消费者、商户交易网站、网上支付平台），且交易流程是分两阶段进行，所以商户交易平台需要开发两个主要的程序才能完成整个支付的流程，此两支程序为“支付请求程序”及“支付结果接收页面”。交易的过程根据支付结果的接收方式的不同而不同，两种交易流程分别如下图所示：

页面通知支付结果方式：





服务器通知支付结果方式:

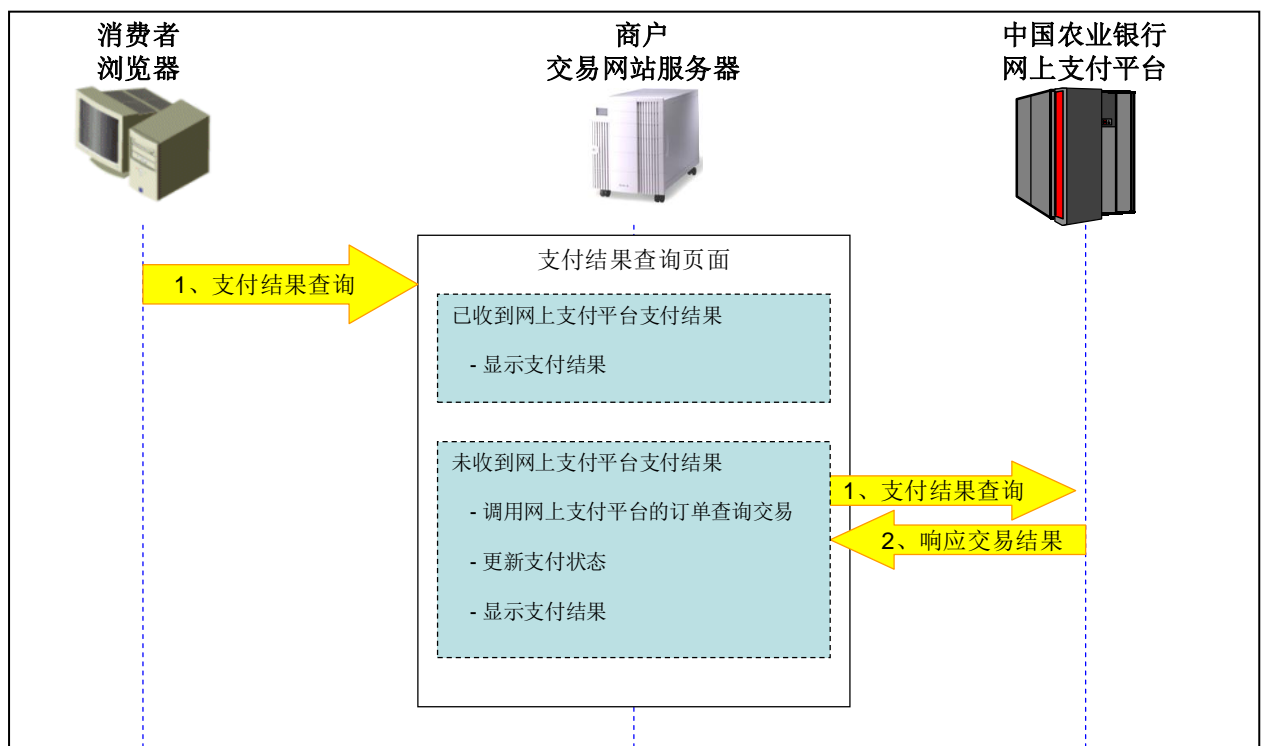
### 5.1.2 确保支付结果正确送达商户网站的措施

网上支付平台为了防止网络异常中断所造成的支付结果丢失，建议商户实现下列网上支付平台所提供的机制。

#### 订单查询交易

针对未收到银行交易结果回复的订单，或银行响应交易状态未明的订单，商户可以在任何时刻主动发起订单查询请求（详细交易说明请参考 **5.7 订单查询**），查询订单（支付）的状态。

例如在商户网站提供消费者支付结果查询的功能，如该订单未收到网上支付平台交易结果，则调用网上支付平台的订单查询交易取得交易结果（订单状态），然后以取得



的交易结果更新商户网站的支付状态。



中国农业银行  
AGRICULTURAL BANK OF CHINA

农行网上支付平台

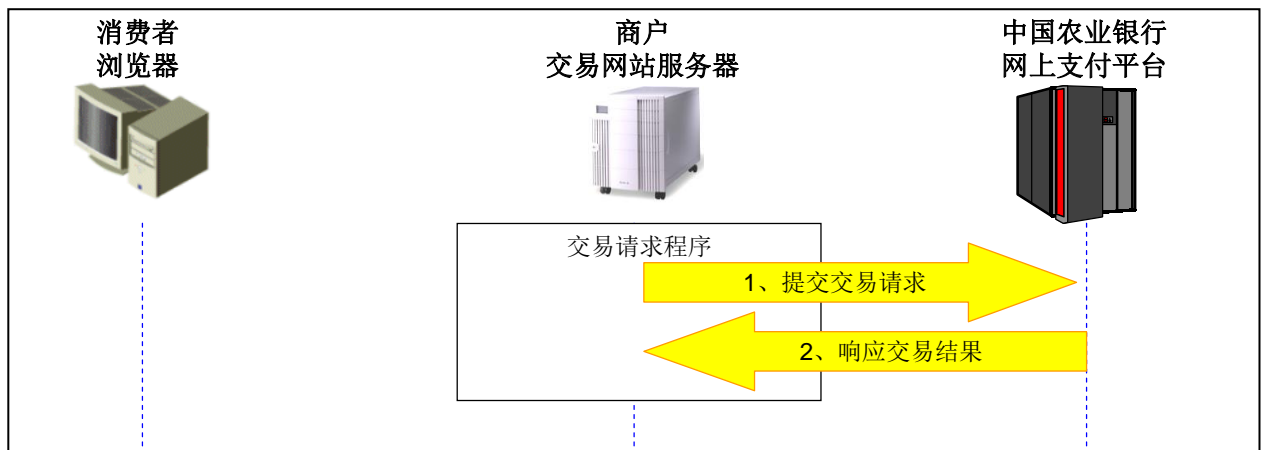
商户接口编程指南– ASP.NET2.0 Edition – V1.0.7

### 通知商户支付成功

支付成功后，如果消费者浏览器安装了某些拦截弹出窗口软件（例如 **3721**），那么支付结果接收页面有可能不会正常弹出，此时消费者可以点击【通知商户支付成功】按钮，重新发送支付结果到商户交易平台，确保商户能够收到网上支付平台的交易结果通知。

### 5.1.3 其它交易

其它的交易（退货、订单状态查询、交易对账单下载）只需要商户及网上支付平台的参与，交易的过程是实时响应，商户只需要简单的开发交易程序即可完成交易的过程。交易过



程如下图所示：

## 5.2 交易使用时机

### 支付请求交易

消费者在商户网站上购买商品，并选择网上支付时。

### 支付结果接收

消费者在网上支付平台上进行在线支付的操作，支付成功后，网上支付平台会将支付的结果通知到商户指定的支付结果通知页面。商户必须开发此页面，否则无法收到支付结果的通知。

### 订单查询

针对未收到银行交易结果回复的订单，或银行响应交易状态未明的订单，商户可以发起订单查询请求，查询订单的状态。


网上支付平台的支付结果页面也会提供消费者通知商户支付成功的链接按钮，用来确定商户是否已经收到网上支付平台的通知。商户必须开发此页面。

### 退货

针对已经结帐的订单，商户可以使用退货的交易来退还交易金额给消费者。退货的交易由商户自行发起，不需要消费者的参与。

### 交易对账单下载

网上支付平台每日根据联机交易后台返回的会计日期来生成对账单。商户可在每日早上08:00 后进行前一日的交易对账单的下载，确定是否有未回传的成功交易。

 <b>中国农业银行</b> <small>AGRICULTURAL BANK OF CHINA</small>	农行网上支付平台
	商户接口编程指南– ASP.NET2.0 Edition – V1.0.7

## 付款信息发送

商户需要对待付款客户进行付款时，可使用该接口生产付款批量信息，编辑付款批次流水号、批次总笔数、批次总金额、备注等批次信息，以及批次内各个付款明细信息，包括：序号、收款方账号、收款方户名、付款金额、用途等，以报文形式发送至网上付款平台。

## 付款交易结果查询

商户操作员在商户服务系统对付款信息进行确认后，可调用付款交易结果查询接口对该批次的付款结果进行查询，接口提供了按照批次号查询和按照收款账号查询两种方式进行查询。

## 银行卡状态验证查询

商户针对待付款的收款方进行维护时，可先行调用该接口验证账户与户名一致性、账户状态是否可以收款等信息，以免发生收款方填写错误或收款方账户冻结等情况。

# 5.3 支付请求

## 5.3.1 方式 1：通过与农行服务器建立连接访问农行 b2c 支付平台服务

当消费者在商户的网站购物确定订单后，选择使用网上支付付款。商户首先提交支付请求给网上支付平台，接着将消费者的浏览器导到农行网上支付平台的支付页面。消费者在网上支付平台上进行在线支付的操作，支付成功后，网上支付平台会将支付结果通知给商户，目前通知方式有两种：

- ◆ 页面通知：网上支付平台会将支付的结果通知到商户指定的支付结果通知页面。
- ◆ 服务器通知：网上支付平台会将支付的结果通过支付平台的服务器直接发送到商户知道的 URL 连接



提示：开发过程中，当商户将消费者的浏览器导到农行网上支付平台的支付页面时，有可能会受到消费者安装了某些拦截弹出窗口软件（例如 **3721**）的影响，所以提醒商户请选择正确的页面跳转方式（例如通过按钮点击或者重新刷新页面等方式）。

#### 1、生成订单对象 `com.hitrust.trustpay.client.b2c.Order`

##### 2、设定订单对象的属性

OrderNo          订单编号（必要信息）

ExpiredDate      订单有效期(必填信息)

**注意：订单有效期是该订单在农行数据库中保留的天数，超过有效期的订单将会被清理。**

OrderDesc        订单说明

OrderDate        订单日期（必要信息 – YYYY/MM/DD）

OrderTime        订单时间（必要信息 – HH:MM:SS）

OrderAmount      订单金额（必要信息）

OrderURL         订单查询网址（必要信息）

BuyIP             买方 IP 地址信息

#### 3、生成订单明细对象 `com.hitrust.trustpay.client.b2c.OrderItem`，并将订单明细加入订单中（可选信息）

#### 4、生成支付请求对象 `com.hitrust.trustpay.client.b2c.PaymentRequest`

##### 5、设定支付请求对象的属性

Order             订单对象（必要信息）

ProductType                      设定商品种类（必要信息）

PaymentType                      设定支付类型（必要信息）

注意：目前支付类型分为农行卡支付、国际卡支付、贷记卡支付以及农行卡和贷记卡支付合并四种，另外为了满足商户要求，我们还增加了一种基于第三方的跨行支付的方式。

如果商户设定了农行卡支付（PaymentType 为 1），成功提交支付请求后就会给消费者直接导向到农行卡支付页面；

如果商户设定了国际卡支付（PaymentType 为 2），成功提交支付请求后就会给消费者直接导向到国际卡支付页面。

如果商户设定了贷记卡支付（PaymentType 为 3），成功提交支付请求后就会给消费者直接导向到贷记卡支付页面。

如果商户设定了农行卡和贷记卡支付合并（PaymentType 为 A），成功提交支付请求后就会给消费者直接导向到农行卡和贷记卡支付方式合并页面，用户在该页面可以选择农行卡支付或者贷记卡支付。

如果商户设定了基于第三方的跨行支付（PaymentType 为 5），成功提交支付请求后就会给消费者直接导向到基于第三方的跨行支付页面。

PaymentLinkType    设定支付接入方式（必要信息）

注意：目前支持三种接入方式，Internet 网络接入，Mobile 网络接入，数字电视网络接入，不同的支付方式会返回不同的支付处理页面。

NotifyType                      设定支付结果通知方式（必要信息）

ResultNotifyURL              支付结果地址（必要信息）

注意：

如果支付结果通知方式选择了页面通知，此处填写就是支付结果回传网址；

如果支付结果通知方式选择了服务器通知，此处填写的就是接收支付平台服务器发送响应信息的地址。

MerchantRemarks 商户备注信息

BuyIP 商户填写请求支付的客户端的 IP 地址信息

6、使用支付请求对象的 `postRequest()` 方法传送支付请求并取得交易结果对象

7、使用交易结果对象的 `isSuccess()` 方法辨别支付请求是否成功

8、若请求成功，可以使用交易结果对象的 `getValue("PaymentURL")` 方法取得支付页面网址，并将消费者的浏览器导向到此支付页面网址进行支付。

支付平台页面支持多种语言，如果商户需要将支付页面导向其他语种界面，支付页面 URL 网址需设定为：`getValue("PaymentURL") + &language=语种代码`

语种代码如下：中文-----ZH

英文-----EN

法语-----FR

德语-----DE

韩语-----KO

语种代码不区分大小写，目前支付平台只支持中文和英文两种语言系统，如果不传入语种代码参数，则默认为中文。

9、若请求失败，可以使用交易结果对象的 `getReturnCode ()` 及 `getErrorMessage()` 方法取得交易失败原因。

交易结果对象的 `getReturnCode ()` 所回传的响应码请参考《附录二、响应码一览表》的说明。

### 5.3.2 方式 2：通过页面传参提交表单方式访问农行 b2c 支付平台服务

该种访问方式用于某些商户服务器不能通过方式 1 访问农行 b2c 支付平台的情况；通过这种方式访问农行 b2c 支付平台服务时，接口程序的处理流程步骤 1-5 与方式 1 一样（参见方式 1 的步骤 1-5）。

6. 使用支付请求对象的 `genSignature ()` 方法产生经过商户服务器证书签名的交易报文。在此过程中如果发生错误，会将错误结果和错误码返回到 `MerchantPaymentIE.aspx` 页面，该页面需要商户开发。

7. 如果没有错误，则将该参数通过表单提交方式转到农行支付平台进行处理；商户还需要开发一个错误页面 `MerchantIEFailure.aspx`，将其访问地址作为另外一个参数传给农行支付平台（对应配置文件中的 `MerchantErrorURL` 项）；当支付平台处理该商户请求报文如果有错误发生时，会将错误结果和错误码返回到商户的 `MerchantIEFailure.aspx` 页面，以便商户进行后续处理。农行支付平台的入口地址是：

<https://easyabc.95599.cn/b2c/trustpay/ReceiveMerchantIERequestServlet>（对应配置文件中的 `TrustPayIETrxURL` 项，请在具体配置时与农行人员联系）；

8. 如果支付平台处理商户的请求报文成功，则支付平台自动将消费者的浏览器导向到支付页面网址进行支付。

## 5.4 两种接收支付结果方式的区别

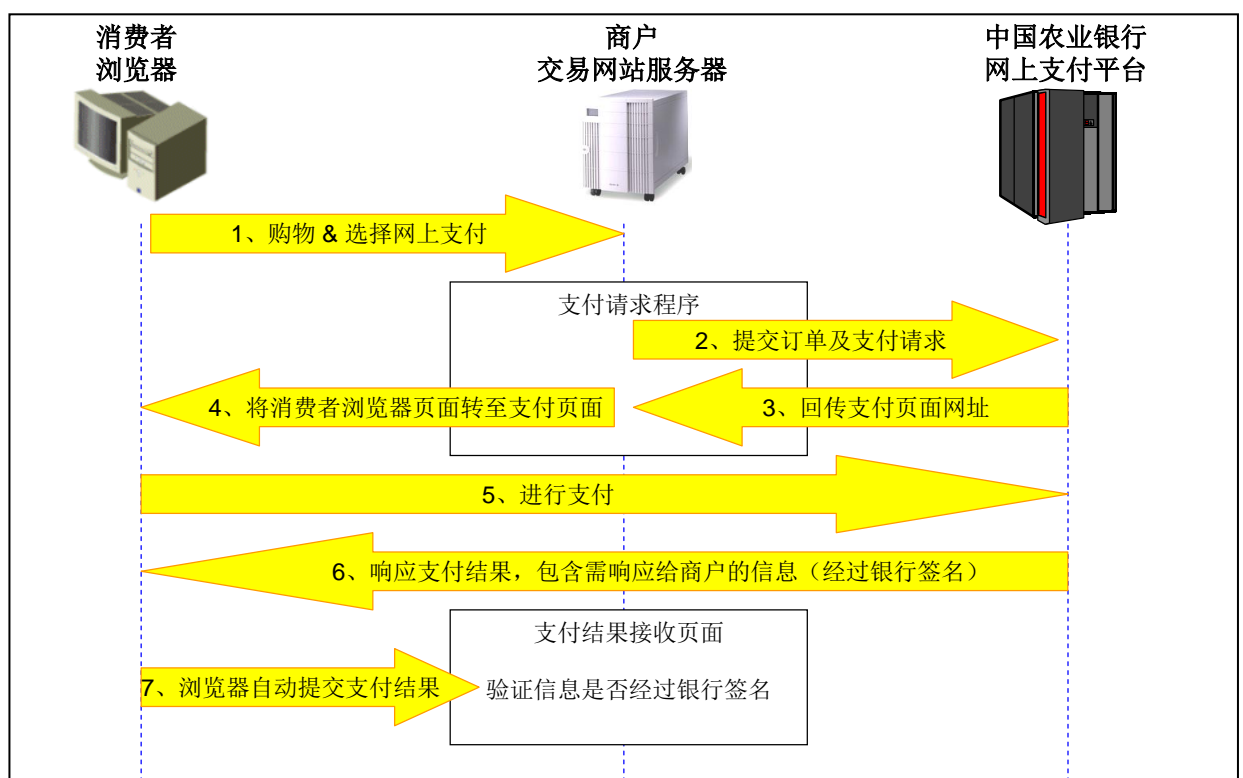
消费者在网上支付平台上进行在线支付的操作，支付成功后，网上支付平台会将支付结果通知给商户，目前通知方式有两种：**通过显示给消费者的支付结果接收页面通知商户**和**通过支付平台服务器通知商户**

### 5.4.1 通过显示给消费者的支付结果接收页面通知商户

商户选择此种接收支付结果通知的方式，需要开发一个接收支付结果通知的页面。

商户在向网上支付平台发送交易请求的时候选择通过**页面通知**方式接收支付结果，传送给支付平台一个支付结果通知的页面地址；然后消费者在网上进行在线支付，如果支付成功后，网上支付平台会将支付结果信息通过显示给消费者的支付结果通知页面通知给商户。

交易流程如下：



#### 5.4.2 通过支付平台服务器通知商户

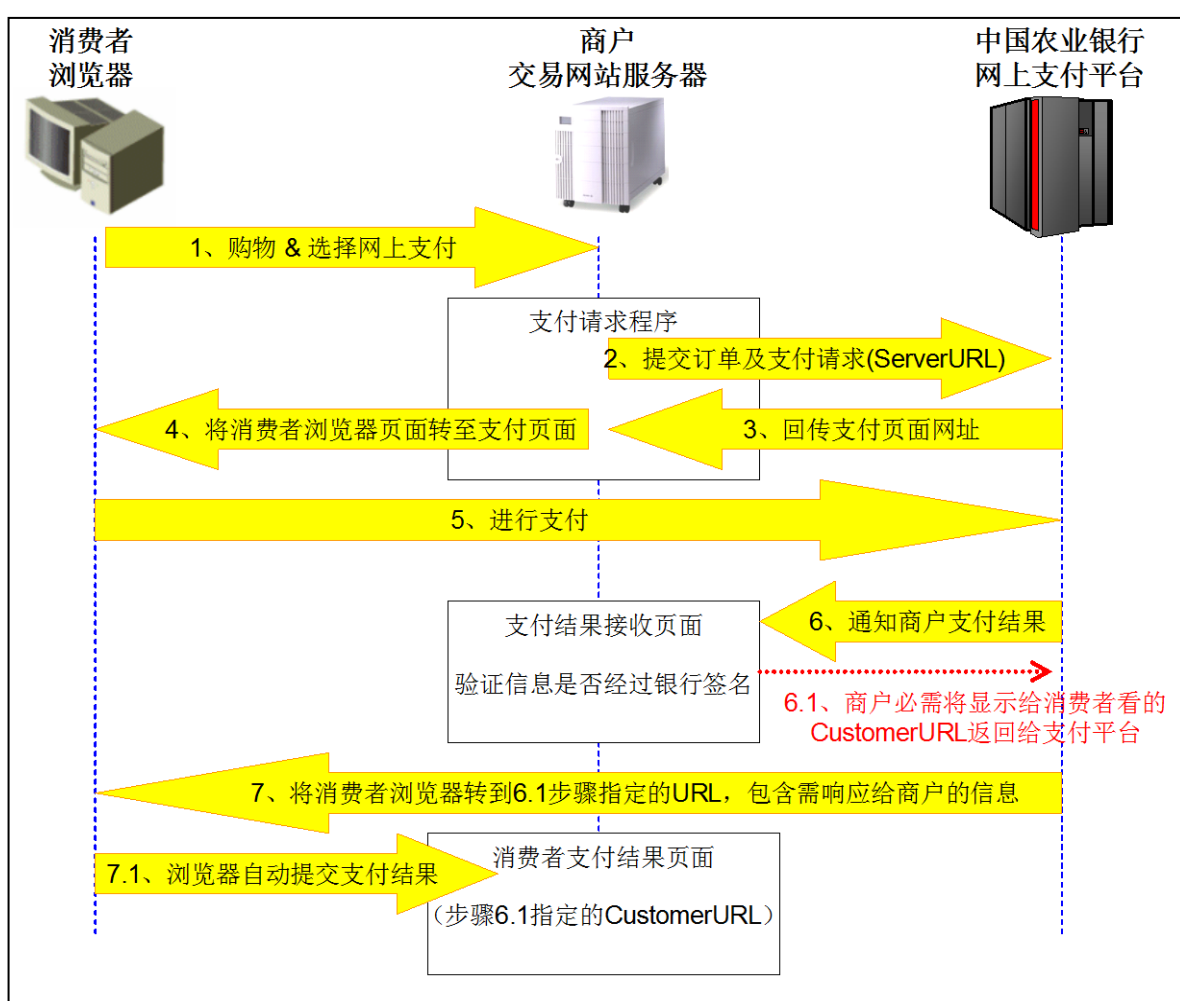
商户选择此种接收支付结果通知的方式，需要开发两个页面：


接收服务器通知的页面（ServerURL）

展示给消费者支付结果信息的页面（CustomerURL）

注意：

这两支页面的 URL 应该是在公网能访问的网址，而且端口号必须为 80 或者 443（http



 <b>中国农业银行</b> AGRICULTURAL BANK OF CHINA	农行网上支付平台
	商户接口编程指南– ASP.NET2.0 Edition – V1.0.7


默认端口为 80；https 默认端口为 443）

商户在向网上支付平台发送交易请求的时候选择通过**服务器通知**的方式接收支付结果，传送给支付平台一个接收服务器通知的页面（**ServerURL**），此页面的 HTML 代码里应该包含一个准备展示给消费者支付结果的 URL 链接（**CustomerURL**）（**注意：链接之间需要用 <URL></URL> 包含**，具体代码参见附录一、C 页面），然后消费者在网上进行在线支付，如果支付成功后，网上支付平台会将支付结果通知给商户，商户接收到支付结果信息后，必需将显示给消费者的页面 URL（**CustomerURL**）链接返回给支付平台服务器，然后支付平台服务器把接收到的这个展示给消费者支付结果信息的页面弹出给消费者显示。

如果第一次通知向商户发送通知时发生下列情况时：

- 1、无法连接到指定的商户交易结果接收页面；
- 2、商户交易结果接收页面没有正确响应消费者支付结果 URL。

系统将会在消费者的浏览器弹出一个新的窗口，并以此新窗口打开商户支付结果接收页面（**ServerURL**）。为了保证在此状况下消费者还是可以看到正常的商户交易结果页面（**CustomerURL**），建议在 **ServerURL** 页面加上自动转向 **CustomerURL** 的脚本，此脚本范例请参考附录一、C 页面。

 <b>中国农业银行</b> <small>AGRICULTURAL BANK OF CHINA</small>	农行网上支付平台
	商户接口编程指南– ASP.NET2.0 Edition – V1.0.7

### 5.4.3 区别

采取通过页面通知的方式将支付结果通知给商户，如果消费者的浏览器里安装了一些弹出窗口拦截软件（例如：3721），就会导致页面无法弹出，商户也就无法接收到通知消息；采用服务器通知的方法，网上支付平台会将支付结果消息通过服务器直接发送给商户指定的URL，而且发送失败以后可以重复发送，这样就保证了商户可以不受消费者本地设置的影响，正确的接收到支付结果通知。




## 5.5 支付结果接收页面

消费者在网上支付平台上进行在线支付的操作，如果商户的支付结果通知方式选择了“页面通知”，那么当支付成功后，网上支付平台会将支付的结果发送到商户指定的支付结果通知页面；如果商户的支付结果通知方式选择了“服务器通知”，那么此页面就是网上支付平台服务器弹出显示给消费者的支付结果页面。商户取得银行通知信息的步骤说明如下：

注意：对于同一笔订单的支付结果信息，网上支付平台有可能会多次提交到商户指定的支付结果页面，所以提醒商户在开发过程中注意处理。

- 1、取得网上支付平台 **post** 的 **MSG** 参数
- 2、利用此参数生成支付结果对象 **com.hitrust.trustpay.client.b2c.PaymentResult**
- 3、使用支付结果对象的 **isSuccess()** 方法辨别支付是否成功
- 4、若支付成功，则商户可以使用支付结果对象的 **getValue()** 方法取得下列回传值来进行后续的作业：

OrderNo	订单号
Amount	订单金额
BatchNo	交易批次号
VoucherNo	交易凭证号（用于交易对账时使用）
HostDate	银行交易日期（YYYY/MM/DD）
HostTime	银行交易时间（HH:MM:SS）
MerchantRemarks	商户备注信息（商户在支付请求时所提交的信息）
PayType	消费者支付方式

 <b>中国农业银行</b> <small>AGRICULTURAL BANK OF CHINA</small>	农行网上支付平台
	商户接口编程指南– ASP.NET2.0 Edition – V1.0.7

NotifyType      支付结果通知方式

5、若请求失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.6 退货请求

退货的交易由商户自行发起，不需要消费者的参与，银行对该交易的响应将立即回传。商户发起退货交易的步骤说明如下：

1、生成退货请求对象 `com.hitrust.trustpay.client.b2c.RefundRequest`

2、设定退货请求对象的属性

`OrderNo`            订单号（必要信息）

`TrxAmount`        退货金额（必要信息）

3、调用退货请求对象的 `postRequest()`方法传送退货请求并取得交易结果对象

4、使用交易结果对象的 `isSuccess()`方法辨别交易是否成功

5、若交易成功，则商户可以使用交易结果对象的 `getValue()`方法取得下列回传值来进行后续的作业：

`OrderNo`            订单号

`TrxAmount`        退货金额


`BatchNo`           交易批次号

`VoucherNo`        交易凭证号（用于交易对账时使用）

`HostDate`         银行交易日期（YYYY/MM/DD）

`HostTime`         银行交易时间（HH:MM:SS）

6、若请求失败，可以使用交易结果对象的 `ReturnCode` 及 `ErrorMessage` 属性取得交易失败原因。

 <b>中国农业银行</b> <small>AGRICULTURAL BANK OF CHINA</small>	农行网上支付平台
	商户接口编程指南– ASP.NET2.0 Edition – V1.0.7

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.7 订单查询

商户可以发起订单查询请求，查询订单的状态。商户发起订单查询交易的步骤说明如下：

1、生成订单查询请求对象 `com.hitrust.trustpay.client.b2c.QueryOrderRequest`

2、设定订单查询请求对象的属性

`OrderNo`            订单号（必要信息）

`DetailQuery`       是否查询详细信息（必要信息） `true / false`

3、调用订单查询请求对象的 `postRequest()`方法传送订单查询请求并取得交易结果对象

4、使用交易结果对象的 `isSuccess()`方法辨别交易是否成功

5、若交易成功，则使用交易结果对象生成订单对象，取得订单的信息。

`OrderNo`            订单编号

`OrderAmount`       订单金额

`OrderDesc`           订单说明（查询详细信息时才回传）

`OrderDate`           订单日期（查询详细信息时才回传）

`OrderTime`           订单时间（查询详细信息时才回传）

`OrderURL`           订单网址（查询详细信息时才回传）

`PayAmount`           支付金额

`RefundAmount`       退货金额

`OrderStatus`        订单状态

◆ 00：订单已取消

- ◆ 01: 订单已建立，等待支付
- ◆ 02: 消费者已支付，等待支付结果
- ◆ 03: 订单已支付（支付成功）
- ◆ 04: 订单已结算（支付成功）
- ◆ 05: 订单已退款
- ◆ 99: 订单支付失败

6、若为详细查询，可以使用订单对象的 **OrderItems** 属性取得订单明细。

7、若请求失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.8 交易对账单下载

商户发起交易对账单下载的步骤说明如下：

1、生成对账单下载请求对象 `com.hitrust.trustpay.client.b2c.SettleRequest`

2、设定对账单下载请求对象的属性

`SettleDate`      对账日期（必要信息）

`SettleType`      对账类型（必要信息），需设定为 `SettleFile.SETTLE_TYPE_TRX`

3、调用对账单下载请求对象的 `postRequest()` 方法传送对账单下载请求对象并取得交易结果对象

4、使用交易结果对象的 `isSuccess()` 方法辨别交易是否成功

5、若交易成功，则使用交易结果对象生成交易对账单对象 `com.hitrust.trustpay.client.SettleFile`

`SettleDate`      对账日期

`SettleType`      对账类型

`NumOfPayments`      该批次支付成功的交易总笔数

`SumOfPayAmount`      该批次支付成功的交易总金额

`NumOfRefunds`      该批次退货成功的交易总笔数

`SumOfRefundAmount`      该批次退货成功的交易总金额

6、使用交易对账单对象的 `getDetailRecords()` 方法取得交易明细，每笔交易明细为字符串类型，

使用逗号分隔不同的字段。字段信息如下：

交易类型 P: 支付交易      R: 退货交易

订单号

交易金额

凭证号

交易时间

7、若请求失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。



## 5.9 多商户交易方式

多商户配置状况因为有多多个商户的配置存在，所以必须在提交交易时指定此交易所属的商户，在前面各节所描述的交易过程中 `postRequest()` 方法将会默认使用第一个商户配置属性。若需要指定其它的商户配置则使用 `extendPostRequest(int 商户配置编号)`方法来提交交易。

### 范例：

若多商户配置的配置如下：

MerchantID=103100070113810, 123456789012345, 103456464564564

要使用商户号 103100070113810 提交交易时，可以使用下列方法：

`postRequest()` 或  
`extendPostRequest(1)`

要使用商户号 123456789012345 提交交易时，必须使用下列方法：

`extendPostRequest(2)`

要使用商户号 103456464564564 提交交易时，必须使用下列方法：

`extendPostRequest(3)`

## 5.10 指定日期指定时间段交易对账单下载

商户发起交易指定时间段对账单下载的步骤说明如下：

### 1、生成对账单下载请求对象 `com.hitrust.trustpay.client.b2c.SettleRequest`

### 2、设定对账单下载请求对象的属性

`SettleDate`          对账日期（必要信息）

`SettleStartHour`   对账开始时间段（必要信息，0-23 之间）

`SettleEndHour`     对账截止时间段（必要信息，0-23 之间）

`SettleType`          对账类型（必要信息），设定为 `SettleFile.SETTLE_TYPE_TRX_BYHOUR`

### 3、调用对账单下载请求对象的 `postRequest()` 方法传送对账单下载请求对象并取得交易结果对象

### 4、使用交易结果对象的 `isSuccess()` 方法辨别交易是否成功

### 5、若交易成功，则使用交易结果对象生成交易对账单对象 `com.hitrust.trustpay.client.SettleFile`

`SettleDate`          对账日期

`SettleType`          对账类型

`NumOfPayments`      该时间段支付成功的交易总笔数

`SumOfPayAmount`    该时间段支付成功的交易总金额

`NumOfRefunds`        该时间段退货成功的交易总笔数

`SumOfRefundAmount` 该时间段退货成功的交易总金额

### 6、使用交易对账单对象的 `DetailRecords` 取得交易明细，每笔交易明细为字符串类型，使用逗号分隔不同的字段。字段信息如下：

交易类型 P: 支付交易      R: 退货交易

订单号

交易金额

凭证号

交易时间

7、若交易失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.11 农行卡身份验证交易请求

商户发起交易验证用户农行卡身份的步骤说明如下：

1、生成卡验证交易请求对象 **com.hitrust.trustpay.client.b2c.CardVerifyRequest**

2、设定卡验证请求对象的属性

**CertificateID** 证件类型（必要信息，参照证件类型列表说明）

**CertificateNo** 证件号码（必要信息）

**ResultNotifyURL** 接收验证结果通知 URL（必要信息）

3、调用卡验证请求对象的 **postRequest()** 方法传送卡验证请求对象并取得交易结果对象

4、使用交易结果对象的 **isSuccess()** 方法辨别交易是否成功

5、若交易成功，可以使用交易结果对象的 **getValue("VerifyURL")** 方法取得卡身份验证页面网址，并将消费者的浏览器导向到此支付页面网址进行支付。

6、若请求失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

## 5.12 卡验证结果接收页面

消费者在网上支付平台上进行卡身份验证的操作，当支付成功后，网上支付平台会将支付的

结果导向到商户请求发送过来的卡验证结果接收页面。商户取得银行通知信息的步骤说明如下：

注意：对于同一个验证请求结果信息，网上支付平台有可能会多次提交到商户指定的验证结果页面，所以提醒商户在开发过程中注意处理。

- 1、取得网上支付平台 **post** 的 **MSG** 参数
- 2、利用此参数生成支付结果对象 **com.hitrust.trustpay.client.b2c.PaymentResult**
- 3、使用支付结果对象的 **isSuccess()** 方法辨别卡验证是否成功
- 4、若卡验证成功，则商户可以进行后续的作业

OrderNo                  订单号

- 5、若卡验证失败，可以使用支付结果对象的 **ReturnCode** 及 **ErrorMessage** 取得支付失败原因。

支付结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.13 身份验证交易请求

商户发起交易验证用户农行卡身份的步骤说明如下：

- 1、生成身份验证交易请求对象 **com.hitrust.trustpay.client.b2c.IdentityVerifyRequest**
- 2、设定身份验证请求对象的属性

CertificateType          证件类型（必要信息，参照证件类型列表说明）

CertificateNo            证件号码（必要信息）

ResultNotifyURL        接收验证结果通知 URL（必要信息）

BankCardNo              银行卡号（必要信息）

OrderDate                      请求发起日期（必要信息）

OrderTime                      请求发起时间（必要信息）

3、调用卡验证请求对象的 **postRequest()**方法传送卡验证请求对象并取得交易结果对象

4、使用交易结果对象的 **isSuccess()**方法辨别交易是否成功

5、若交易成功，可以使用交易结果对象的 **getValue("VerifyURL")**方法取得身份验证页面网址，并将消费者的浏览器导向到此验证页面网址进行身份验证。

6、若请求失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

## 5.14 身份验证结果接收页面

消费者在网上支付平台上进行身份验证的操作，当验证成功后，网上支付平台会将验证的结果导向到商户请求发送过来的验证结果接收页面。商户取得银行通知信息的步骤说明如下：

注意：对于同一个验证请求结果信息，网上支付平台有可能会多次提交到商户指定的验证结果页面，所以提醒商户在开发过程中注意处理。

1、取得网上支付平台 **post** 的 **MSG** 参数

2、利用此参数生成支付结果对象 **com.hitrust.trustpay.client.b2c.PaymentResult**

3、使用支付结果对象的 **isSuccess()**方法辨别验证是否成功

4、若验证成功，则商户可以进行后续的作业

AccountName                      户名

5、若验证失败，可以使用支付结果对象的 **ReturnCode** 及 **ErrorMessage** 取得支付失败原因。

支付结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.15 批量退款发送请求

退款批量发送的交易由商户自行发起，不需要消费者的参与，银行对该交易的响应将立即回传。商户发起退款批量发送交易的步骤说明如下：

### 1、生成批量退款发送请求对象

`com.hitrust.trustpay.client.b2c.OverdueRefundRequest`

### 2、设定批量退款发送请求对象的属性

<b>TotalCount</b>	批量退款订单总笔数（必要信息）
<b>TotalAmount</b>	批量退款总金额（必要信息）
<b>Remark</b>	批量退款备注信息
<b>OrderNo</b>	批量退款订单号码（每笔订单必要信息，最多为 100 条）
<b>RefundAmount</b>	批量退款订单退款金额（每笔订单必要信息，最多为 100 条）

3、调用批量退款发送请求对象的 **postRequest()** 方法传送批量退款发送请求并取得交易结果对象

4、使用交易结果对象的 **isSuccess()** 方法辨别交易是否成功

5、若交易成功，则商户可以取得交易结果对象的其他属性来进行后续的作业

<b>TrxType</b>	交易类型
<b>TotalCount</b>	批量退款订单总笔数
<b>TotalAmount</b>	批量退款退款总金额

SerialNumber 批量退款流水号

HostDate 服务器返回日期

HostTime 服务器返回时间

ResultMessage 返回结果信息

6、若交易失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

7、需要注意的是，发送到支付平台的报文大小每次不能超过 **4KB**，即发送的批量退款订单信息有可能不到 100 笔，但是报文长度超过了 **4KB**，系统回提示如下信息：“商户提交的交易资料不合法 - 报文长度超过 4096Bytes”。

## 5.16 批量退款结果查询请求

批量退款结果查询的交易由商户自行发起，不需要消费者的参与，银行对该交易的响应将立即回传。商户发起批量退款结果查询交易的步骤说明如下：

### 1、生成退款批量结果查询请求对象

`com.hitrust.trustpay.client.b2c.QueryOverdueRefundRequest`

### 2、设定批量退款结果查询请求对象的属性

**SerialNumber** 批量退款请求的流水号号（必要信息）

3、调用退款批量结果查询请求对象的 **postRequest()** 方法传送批量退款结果查询请求并取得交易结果对象

4、使用交易结果对象的 **isSuccess()** 方法辨别交易是否成功

5、若交易成功，则使用交易结果对象生成退款批量和订单对象，取得退款批量和订单的信息

SerialNumber	批量退款流水号
RefundAmount	批量退款总金额
RefundCount	批量退款总笔数
BatchStatus	批量退款状态
OrderNo	订单号
RefundAmount	订单退款金额
OrderStatus	订单状态（0：未处理；1：处理成功；2：处理失败）
OrderDesc	订单退款失败原因

6、若交易失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.17 委托扣款签约请求

商户网站可以将与商户在线签约的支付方客户，引导到农行电子商务平台签约界面，进行委托扣款的三方签约。委托扣款签约交易的步骤说明如下：

1、生成委托扣款签约请求对象

```
com.hitrust.trustpay.client.b2c.OverdueRefundRequest.B2CAgentSignContractRequest
```

2、设置委托扣款签约请求对象的属性

- ◆ **OrderNo**                      订单编号（必要信息）
- ◆ **CertificateType**            证件类型（必要信息，目前只支持公民身份证，请设置为“1”）



注意：非阿拉伯数字“1”）

- ◆ **CertificateNo**      证件号码（必要信息，公民身份证号码）
- ◆ **CardType**          农行卡类型（必要信息，1:农行借记卡准贷记卡 2:农行贷记卡  
A:农行卡合并）
- ◆ **ResultNotifyURL**    接收验证结果通知 URL（必要信息）
- ◆ **RequestDate**        请求日期（必要信息 - YYYY/MM/DD）
- ◆ **RequestTime**        请求时间（必要信息 - HH:MM:SS）
- ◆ **InvalidDate**        过期日期（必要信息 - YYYY/MM/DD，电子商务系统留此日期  
作为扩展功能之用，根据当前业务需求，请设置为较大的日期，如 2100/01/01）
- ◆ **NotifyType**          设定支付结果通知方式（必要信息，0: URL 页面通知 1: 服  
务器通知）

3、调用委托扣款签约请求对象的 **postRequest()**方法传送委托扣款签约请求并取得交易结果对象

4、使用交易结果对象的 **isSuccess()**方法辨别交易是否成功

5、若交易成功，可以使用交易结果对象的 **getValue("B2CAgentSignContractURL")**方法取得委托扣款签约页面网址，并将消费者的浏览器导向到此签约页面网址进行签约。

6、若请求失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.18 委托扣款解约请求

商户网站可以将与商户在线解约的支付方客户，引导到农行电子商务平台解约界面，进行委托扣款解约。委托扣款解约交易的步骤说明如下：

### 1、生成委托扣款解约请求对象

`com.hitrust.trustpay.client.b2c.OverdueRefundRequest.B2CAgentUnsignContractRequest`

### 2、设置委托扣款解约请求对象的属性

- ◆ **OrderNo**                      订单编号（必要信息）
- ◆ **CertificateType**            证件类型（必要信息，目前只支持居民身份证，请设置为“1”  
注意：非阿拉伯数字“1”）
- ◆ **CertificateNo**                证件号码（必要信息，公民身份证号码）
- ◆ **ResultNotifyURL**            接收验证结果通知 URL（必要信息）
- ◆ **RequestDate**                请求日期（必要信息 - YYYY/MM/DD）
- ◆ **RequestTime**                请求时间（必要信息 - HH:MM:SS）
- ◆ **AgentSignNo**                签约协议号（必要信息）
- ◆ **NotifyType**                  设定支付结果通知方式（必要信息，0：URL 页面通知 1：服务器通知）

### 3、调用委托扣款解约请求对象的 `postRequest()` 方法传送委托扣款解约请求并取得交易结果对象

### 4、使用交易结果对象的 `isSuccess()` 方法辨别交易是否成功

5、若交易成功，可以使用交易结果对象的 `getValue("B2CAgentSignContractURL")` 方法取得委托扣款解约页面网址，并将消费者的浏览器导向到此解约页面网址进行解约。

6、若请求失败，可以使用交易结果对象的 `ReturnCode` 及 `ErrorMessage` 属性取得交易失败原因。

交易结果对象的 `ReturnCode` 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.19 委托扣款单笔代扣请求

商户网站可以将支付客户提交农行支付的委托扣款账单上传到电子商务平台，电子商务平台将账单发送后台系统进行自动扣款处理。

### 1、生成委托扣款单笔代扣请求对象

`com.hitrust.trustpay.client.b2c.OverdueRefundRequest.B2CAgentPaymentRequest`

### 2、设置委托扣款解约请求对象的属性

- ◆ `OrderNo`                      订单编号（必要信息）
- ◆ `ExpiredDate`                  订单有效期（必要信息）
- ◆ `CertificateNo`                证件号码（必要信息，公民身份证号码）
- ◆ `RequestDate`                请求日期（必要信息 - YYYY/MM/DD）
- ◆ `RequestTime`                请求时间（必要信息 - HH:MM:SS）
- ◆ `AgentSignNo`                签约协议号（必要信息）
- ◆ `Currency`                    账单币种（必要信息，设置为：“RMB”）

- ◆ **Amount**                      账单金额（必要信息, 小数点后最多两位）
- ◆ **ProductId**                    商品编号（必要信息）
- ◆ **ProductName**                商品名称（必要信息）
- ◆ **Quantity**                    商品数量（必要信息）

3、调用委托扣款单笔代扣请求对象的 **postRequest()** 方法传送委托扣款单笔代扣请求并取得交易结果对象

4、使用交易结果对象的 **isSuccess()** 方法辨别交易是否成功

5、若交易成功，则显示交易成功信息

6、若请求失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.20 委托扣款批量请求

委托扣款批量请求的交易由商户自行发起，不需要消费者的参与，银行对该交易的响应将立即回传。商户发起委托扣款批量交易的步骤说明如下：

1、生成委托扣款批量请求对象

```
com.hitrust.trustpay.client.b2c.B2CAgentBatchRequest
```

2、设定委托扣款批量请求对象的属性

**BatchNo**                      批次编号（必要信息）

<b>BatchDate</b>	批次日期（必要信息）
<b>AgentCount</b>	批次总比数（必要信息，每批次最多包含 100 笔）
<b>AgentAmount</b>	批次总金额（必要信息）
<b>OrderNo</b>	订单编号（每笔订单必要信息，最多为 100 条）
<b>OrderAmount</b>	订单金额（每笔订单必要信息，最多为 100 条）
<b>CertificateNo</b>	证件号码（每笔订单必要信息，最多为 100 条）
<b>ContractID</b>	签约协议号（每笔订单必要信息，最多为 100 条）
<b>ProductID</b>	商品编号（每笔订单必要信息，最多为 100 条）
<b>ProductName</b>	商品名称（每笔订单必要信息，最多为 100 条）
<b>ProductNum</b>	商品数量（每笔订单必要信息，最多为 100 条）

- 3、调用委托扣款批量请求对象的 **postRequest()** 方法传送委托扣款批量请求并取得交易结果对象
- 4、使用交易结果对象的 **isSuccess()** 方法辨别交易是否成功
- 5、若交易成功，则显示交易成功信息
- 6、若交易失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.21 委托扣款批量结果查询

委托扣款批量结果查询请求的交易由商户自行发起，不需要消费者的参与，银行对该交易的响应将立即回传。商户发起委托扣款批量结果查询交易的步骤说明如下：

- 1、生成委托扣款批量请求对象

com.hitrust.trustpay.client.b2c.B2CAgentBatchRequest

## 2、设定委托扣款批量请求对象的属性

**BatchNo**            批次编号（必要信息）

**BatchDate**        批次日期（必要信息）

## 3、调用委托扣款批量结果查询请求对象的 **postRequest()** 方法传送委托扣款批量结果查询请求并取得交易结果对象

## 4、使用交易结果对象的 **isSuccess()** 方法辨别交易是否成功

## 5、若交易成功，则使用交易结果对象生成委托扣款批量和订单对象，取得委托扣款批量和订单的信息

**BatchNo**            批量编号

**BatchDate**        批量日期

**BatchTime**        批量时间

**AgentAmount**    委托扣款总金额

**AgentCount**      委托扣款总比数

**BatchStatus**      批量状态

**BatchStatusZH**   批量状态中文显示

订单对象中包含以下信息：

**OrderNo**           订单编号

**OrderAmount**    订单金额

**CertificateNo**    证件号码

**ContractID**       签约协议号

**ProductID**        商品编号

ProductName	商品名称
ProductNum	商品数量
ExpiredDate	订单有效期
OrderStatus	订单状态
OrderStatusZH	订单状态中文显示

6、若交易失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.22 委托扣款签约结果查询

委托扣款签约结果查询请求的交易由商户自行发起，不需要消费者的参与，银行对该交易的响应将立即回传。商户发起委托扣款签约查询交易的步骤说明如下：

### 1、委托扣款签约查询请求对象

`com.hitrust.trustpay.client.b2c. QueryAgentSignRequest`

### 2、设定委托扣款签约查询对象的属性

<b>tAgentSignNo</b>	签约号（必要信息）
---------------------	-----------

3、调用委托扣款签约查询请求对象的 **postRequest()** 方法传送委托扣款签约查询请求并取得交易结果对象

4、使用交易结果对象的 **isSuccess()** 方法辨别交易是否成功

5、若交易成功，则使用交易结果对象生成委托扣款签约对象，取得委托扣款签约查询的信息

AgentSignNo	签约协议号
MerchantNo	商户终端代码
CertificateType	证件类型  I：公民身份证
CertificateNo	证件号码
CardNo	账号后四位
SignDate	签约日期
UnSignDate	解约日期
SignStatus	协议状态  00：签约请求建立，等待签约  01：签约成功  02：签约失败  03：解约成功  99：签约状态未知
AccountType	签约银行卡类型  401：农行借记卡  403：农行准贷记卡  404：农行贷记卡



6、若交易失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.23 贷记卡交易对账单下载

商户发起贷记卡交易对账单下载的步骤说明如下：（接口端逻辑跟 5.9 章所述的一样，新增一种对账类型）

1、生成对账单下载请求对象 **com.hitrust.trustpay.client.b2c.SettleRequest**

2、设定对账单下载请求对象的属性

**SettleDate**          对账日期（必要信息）

**SettleType**          对账类型（必要信息），需设定为

**SettleFile.SETTLE\_TYPE\_CREDIT\_TRX**

3、调用对账单下载请求对象的 **postRequest()** 方法传送对账单下载请求对象并取得交易结果对象

4、使用交易结果对象的 **isSuccess()** 方法辨别交易是否成功

5、若交易成功，则使用交易结果对象生成交易对账单对象 **com.hitrust.trustpay.client.SettleFile**

**SettleDate**          对账日期

**SettleType**          对账类型

**NumOfPayments**      该批次支付成功的交易总笔数

**SumOfPayAmount**    该批次支付成功的交易总金额

**NumOfRefunds**        该批次退货成功的交易总笔数

**SumOfRefundAmount** 该批次退货成功的交易总金额

6、使用交易对账单对象的 `getDetailRecords()` 方法取得交易明细，每笔交易明细为字符串类型，使用逗号分隔不同的字段。字段信息如下：

交易类型 P: 支付交易      R: 退货交易

订单号

交易金额

凭证号

交易时间

7、若交易失败，可以使用交易结果对象的 `ReturnCode` 及 `ErrorMessage` 属性取得交易失败原因。

交易结果对象的 `ReturnCode` 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.24 保险直销支付请求

### 5.24.1 方式 1：通过与农行服务器建立连接访问农行 b2c 支付平台服务

当消费者在商户网站购买保险时，商户网站产生订单，订单确认后，选择使用网上支付付款。商户首先提交保险直销支付请求给网上支付平台，接着将消费者的浏览器导到农行网上支付平台的支付页面。消费者在网上支付平台上进行在线支付的操作，支付成功后，网上支付平台会将支付结果通知给商户，目前通知方式有两种：

- ◆ 页面通知：网上支付平台会将支付的结果通知到商户指定的支付结果通知页面。
- ◆ 服务器通知：网上支付平台会将支付的结果通过支付平台的服务器直接发送到商户知道的 URL 连接。

提示：开发过程中，当商户将消费者的浏览器导到农行网上支付平台的支付页面时，有可能会受到消费者安装了某些拦截弹出窗口软件（例如 **3721**）的影响，所以提醒商户请选择正确的页

面跳转方式（例如通过按钮点击或者重新刷新页面等方式）。

#### 1、生成订单对象 `com.hitrust.trustpay.client.b2c.Order`

#### 2、设定订单对象的属性

OrderNo	订单编号（必要信息）
ExpiredDate	订单有效期（必要信息）
OrderDesc	订单说明
OrderDate	订单日期（必要信息 – YYYY/MM/DD）
OrderTime	订单时间（必要信息 – HH:MM:SS）
OrderAmount	订单金额（必要信息）
OrderURL	订单查询网址（必要信息）

#### 3、生成订单明细对象 `com.hitrust.trustpay.client.b2c.OrderItem`，并将订单明细加入订单中（可选信息）

#### 4、生成保险对象 `com.hitrust.trustpay.client.b2c.Insure`

#### 5、设定保险对象属性

Type 保险支付类型

保险支付有三种类型，分别是 1 一般支付、2 理财支付和 3 指定支付

Furl 身份验证失败信息的商户 URL

#### 6、生成保单对象 `com.hitrust.trustpay.client.b2c.InsureOrder`

#### 7、生成保险明细对象 `com.hitrust.trustpay.client.b2c.InsureOrderItem`，并将保险明细加入到保单对象中。

## 8、设定保险明细对象属性

Name	保险名称
Code	保险代码
Category	险种信息
Mode	销售方式
Amount	投保金额

9、生成保单用户对象 `com.hitrust.trustpay.client.b2c.InsureUser`，只有是保险理财支付或者是指定支付才需要设置保单用户信息。

## 10、设置保单用户对象属性

Name	保险人姓名
CertificateType	证件类型
CertificateNo	证件号码
CardNo	银行卡号

## 11、生成支付请求对象 `com.hitrust.trustpay.client.b2c.PaymentRequest`

## 12、设定支付请求对象的属性

Order	订单对象（必要信息）
ProductType	设定商品种类（必要信息）
PaymentType	设定支付类型（必要信息）

注意：目前支付类型分为农行卡支付、国际卡支付、贷记卡、农行卡和贷记卡支付合并以及基于第三方的跨行支付五种，但是保险支付目前只支持农行卡支付。

如果商户设定了农行卡支付，成功提交保险支付请求后就会给消费者直接导向到农行卡支付页面；

NotifyType                      设定支付结果通知方式（必要信息）

ResultNotifyURL              支付结果地址（必要信息）

注意：

如果支付结果通知方式选择了页面通知，此处填写就是支付结果回传网址；

如果支付结果通知方式选择了服务器通知，此处填写的就是接收支付平台服务器发送响应信息的地址。

MerchantRemarks            商户备注信息

Insure                          设定保险信息

13、使用支付请求对象的 `postRequest()` 方法传送支付请求并取得交易结果对象

14、使用交易结果对象的 `isSuccess()` 方法辨别支付请求是否成功

15、若请求成功，可以使用交易结果对象的 `getValue("PaymentURL")` 方法取得支付页面网址，并将消费者的浏览器导向到此支付页面网址进行支付。

支付平台页面支持多种语言，如果商户需要将支付页面导向其他语种界面，支付页面 URL 网址需设定为：`getValue("PaymentURL") + &language=语种代码`

语种代码如下：    中文-----ZH

英文-----EN

法语-----FR

德语-----DE

韩语-----KO

语种代码不区分大小写，目前支付平台只支持中文和英文两种语言系统，如果不传入语种代码参数，则默认为中文。

16、若请求失败，可以使用交易结果对象的 `getReturnCode ()` 及 `getErrorMessage()` 方法取得交易失败原因。

17、交易结果对象的 `getReturnCode ()` 所回传的响应码请参考《附录二、响应码一览表》的说明。

#### 5.24.2 方式 2：通过页面传参提交表单方式访问农行 b2c 支付平台服务

该种访问方式用于某些商户服务器不能通过方式 1 访问农行 b2c 支付平台的情况；通过这种方式访问农行 b2c 支付平台服务时，接口程序的处理流程步骤 1-5 与方式 1 一样（参见方式 1 的步骤 1-5）。

6. 使用支付请求对象的 `genSignature ()` 方法产生经过商户服务器证书签名的交易报文。在此过程中如果发生错误，会将错误结果和错误码返回到 `MerchantIssueFailure.aspx` 页面，该页面需要商户开发。

7. 如果没有错误，则将该参数通过表单提交方式转到农行支付平台进行处理；商户还需要开发一个错误页面 `MerchantInsureFailure.aspx`，将其访问地址作为另外一个参数传给农行支付平台（对应配置文件中的 `MerchantErrorURL` 项）；当支付平台处理该商户请求报文如果有错误发生时，会将错误结果和错误码返回到商户的 `MerchantInsureFailure.aspx` 页面，以便商户进行后续处理。农行支付平台的入口地址是：

<https://easyabc.95599.cn/b2c/trustpay/ReceiveMerchantIERequestServlet>（对应配置文件中的 `TrustPayIETrxURL` 项，请在具体配置时与农行人员联系）；

8.如果支付平台处理商户的请求报文成功，则支付平台自动将消费者的浏览器导向到支付页面网址进行支付。

## 5.25 网上付款信息发送请求

网上付款信息发送的交易由商户自行发起，不需要收款方的参与，银行对该交易的响应将立即回传。商户发起网上付款交易的步骤说明如下：

### 1、生成网上付款信息发送请求对象

`com.hitrust.trustpay.client.b2c.OnlineRemitRequest`

### 2、设定网上付款信息发送请求对象的属性

<b>SerialNumber</b>	商户付款流水号（必要信息，不可与以往流水号重复）
<b>TotalCount</b>	网上付款批量总笔数（必要信息）
<b>TotalAmount</b>	网上付款批量退款总金额（必要信息）
<b>Remark</b>	网上付款批量备注信息
<b>NO</b>	网上付款单笔序号（每笔付款必要信息，批次内不能重复）
<b>CardNo</b>	网上付款单笔收款方账号（每笔付款必要信息，最多为 100 条）
<b>CardName</b>	网上付款单笔收款方户名（每笔付款必要信息，最多为 100 条）
<b>RemitAmount</b>	网上付款单笔付款金额（每笔付款必要信息，最多为 100 条）
<b>Purpose</b>	网上付款单笔付款用途（最多为 100 条）

### 3、调用网上付款信息发送请求对象的 `postRequest()` 方法传送付款发送请求并取得交易结果对象

### 4、使用交易结果对象的 `isSuccess()` 方法辨别交易是否成功

### 5、若交易成功，则商户可以取得交易结果对象的其他属性来进行后续的作业

TrxType	交易类型
TotalCount	网上付款付款总笔数
TotalAmount	网上付款付款总金额
SerialNumber	网上付款批量流水号
ReturnCode	返回信息代码（0000 为成功，其它为失败）
ErrorMessage	返回结果信息（ReturnCode 为“0000”时为“交易成功”，其它则为具体错误信息）

6、若交易失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.26 网上付款交易结果查询请求

网上付款交易结果查询的交易由商户自行发起，不需要收款方的参与，银行对该交易的响应将立即回传。商户发起网上付款交易结果查询交易的步骤说明如下：

1、生成付款结果查询请求对象 `com.hitrust.trustpay.client.b2c.OnlineRmtQueryResultRequest`

2、设定网上付款交易结果查询请求对象的属性

<b>SerialNumber</b>	网上付款交易请求流水号（与 <b>ReceiveAccount</b> 至少填一个）
<b>ReceiveAccount</b>	收款方账号（与 <b>SerialNumber</b> 至少填一个）
<b>StartTime</b>	所要查询流水的起始时间
<b>EndTime</b>	所要查询流水的截止时间

3、调用网上付款交易结果查询请求对象的 **postRequest()** 方法传送网上付款交易结果查询请求并取得交易结果对象



4、使用交易结果对象的 **isSuccess()**方法辨别交易是否成功

5、若交易成功，则使用交易结果对象生成网上付款交易结果查询结果对象，取得网上付款交易结果查询结果信息，返回结果集分为两种情况：

收款方账号（**ReceiveAccount**）为空时：

<b>SerialNumber</b>	网上付款批量流水号
<b>TrnxTime</b>	交易时间
<b>TotalCount</b>	批量总笔数
<b>TotalAmount</b>	批量总金额
<b>Status</b>	批量状态（0：未处理；1：处理中；2：处理成功；3：处理失败）
<b>SuccessAmount</b>	处理成功金额
<b>SuccessCount</b>	处理成功笔数
<b>FailAmount</b>	处理失败金额
<b>FailCount</b>	处理失败笔数
<b>No</b>	批量内部序号
<b>PayAccount</b>	付款方账号
<b>PayAccountName</b>	付款方账户名
<b>ReceiveAccount</b>	款方账号
<b>ReceiveAccountName</b>	款方账户名
<b>Purpose</b>	付款用途
<b>PayAmount</b>	付款金额
<b>Status</b>	付款状态（0：未处理；1：处理中；2：处理成功；3：处理失败）
<b>FailReason</b>	失败原因

收款方账号（**ReceiveAccount**）不为空时：

SerialNumber	网上付款批量流水号
TrnxTime	交易时间
No	批量内部序号
PayAccount	付款方账号
PayAccountName	付款方账户名
ReceiveAccount	款方账号
ReceiveAccountName	款方账户名
Purpose	付款用途
PayAmount	付款金额
Status	付款状态（0：未处理；1：处理中；2：处理成功；3：处理失败）
FailReason	失败原因

6、若交易失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 5.27 网上付款银行卡状态验证请求

网上付款银行卡状态验证的交易由商户自行发起，不需要收款方的参与，银行对该交易的响应将立即回传。商户发起网上付款银行卡状态验证的步骤说明如下：

### 1、生成网上付款银行卡状态验证请求对象

```
com.hitrust.trustpay.client.b2c.OnlineRmtCardVerifyRequest
```

### 2、设定网上付款银行卡状态验证请求对象的属性

BankCardNo            银行卡卡号

AccountName           银行卡户名

3、调用网上付款银行卡状态验证请求对象的 **postRequest()** 方法传送付款发送请求并取得交易结果对象

4、使用交易结果对象的 **isSuccess()** 方法辨别交易是否成功

5、若交易成功，则商户可以取得交易结果对象的其他属性来进行后续的作业

TrxType                交易类型

ReturnCode            返回信息代码（0000 为成功，其它为失败）

ErrorMessage        返回结果信息（ReturnCode 为“0000”时为“账户状态正常”，其它则为具体错误信息）

6、若交易失败，可以使用交易结果对象的 **ReturnCode** 及 **ErrorMessage** 属性取得交易失败原因。

交易结果对象的 **ReturnCode** 所回传的响应码请参考《附录二、响应码一览表》的说明。

## 6. 附录一、程序范例

### A、配置文件 web.config 范例

此配置文件假定了以下事项：

- TrustPay.cer、abc.truststore和.pfx存放于D:\abc目录（也可以在其它盘符的其它目录）下；
- .pfx密码为：1a2b3c4d。（.pfx文件名称和密码可以改变，例如这里的文件名叫200905060000111.pfx）

注意红色标注的部分，部署时主要修改这些内容。

```
<?xml version="1.0"?>

<configuration>

  <configSections>

    <section name="log4net" type="System.Configuration.IgnoreSectionHandler"/>

  </configSections>

  <appSettings>

    <!--

    #=====

    # 网上支付平台系统配置段 - 生产环境 - 请勿更改

    #=====

    #网上支付平台通讯方式 (http / https)

    -->

    <add key="TrustPayConnectMethod" value="https"/>

    <!--#网上支付平台服务器名-->
```

```
<add key="TrustPayServerName" value="www.95599.cn"/>

<!--#网上支付平台交易端口-->

<add key="TrustPayServerPort" value="443"/>

<!--#网上支付平台接口特性-->

<add key="TrustPayNewLine" value="2"/>

<!--#网上支付平台交易网址-->

<add key="TrustPayTrxURL" value="/b2c/trustpay/ReceiveMerchantTrxReqServlet"/>

<!--#商户通过浏览器提交网上支付平台交易网址-->

<add key="TrustPayIETrxURL"
value="https://www.95599.cn/b2c/trustpay/ReceiveMerchantIERequestServlet"/>

<!--

=====

# 网上支付平台系统配置段 - 生产环境 - 更改证书存放路径,使其和本地存放路径相匹配(绝对路径)

=====

#网上支付平台证书

-->

<add key="TrustPayCertFile" value="D:\abc\TrustPay.cer"/>

<!--#农行根证书文件-->

<add key="TrustStoreFile" value="D:\abc\abc.truststore"/>

<!--#农行根证书文件密码-->

<add key="TrustStorePassword" value="changeit"/>

<!--

=====

# 商户资料段 (请更改)

=====

#商户编号

-->

<add key="MerchantID" value="200905060000111"/>
```

```
<!--#商户通过浏览器提交网上支付平台交易失败网址-->

<add key="MerchantErrorURL"
value="http://172.30.7.94:52705/demo/IE/MerchantIEFailure.aspx"/>

<!--

#=====

# 商户系统配置段 (请更改)

#=====

#交易日志开关 (true: 表示写日志, false: 表示不写日志)

#如果为true, 需要填写 #交易日志文件存放目录

-->

<add key="EnableLog" value="true"/>

<!--

#证书储存媒体

#0: File

#1: Hardware

-->

<add key="MerchantKeyStoreType" value="0"/>

<!--#商户证书储存目录档名 (当 KeyStoreType=0 时, 必须设定)-->

<add key="MerchantCertFile" value="D:\abc\200905060000111.pfx"/>

<!--#商户私钥加密密码 (当 KeyStoreType=0 时, 必须设定)-->

<add key="MerchantCertPassword" value="1a2b3c4d"/>

<!--#Sign Server 地址 (当 KeyStoreType=1 时, 必须设定)-->

<add key="SignServerIP" value=""/>

<!--#Sign Server 端口 (当 KeyStoreType=1 时, 必须设定)-->

<add key="SignServerPort" value=""/>

<!--#Sign Server 密码 (当 KeyStoreType=1 时, 选择设定)-->

<add key="SignServerPassword" value=""/>

<!--

#=====
```

```
-->

</appSettings>

<!--

=====

# 网上支付平台日志配置段 - 生产环境 - 请更改交易日志文件存放目录

=====

-->

<log4net>

  <appender name="LogFileAppender" type="log4net.Appender.FileAppender">

    <!--#交易日志文件存放目录-->

    <file value="D:\abc\TrustPayClient.log" />

    <lockingModel type="log4net.Appender.FileAppender+MinimalLock" />

    <appendToFile value="true" />

    <rollingStyle value="Date" />

    <datePattern value="yyyy-MM-dd HH:mm:ss" />

    <layout type="log4net.Layout.PatternLayout">

      <footer value="by TrustPayClient" />

      <conversionPattern value="%date [%thread] %-5level %logger [%ndc]
      &lt;%property{auth}&gt; - %message%newline" />

    </layout>

  </appender>

  <root>

    <appender-ref ref="LogFileAppender" />

  </root>

</log4net>

<connectionStrings/>

<system.web>

<!--
```

设置 `compilation debug="true"` 将调试符号插入

已编译的页面中。但由于这会

影响性能，因此只在开发过程中将此值

设置为 `true`。

-->

```
<compilation debug="true">
```

```
    <assemblies>
```

```
        <add assembly="Microsoft.Web.Services3, Version=3.0.0.0,
Culture=neutral, PublicKeyToken=31BF3856AD364E35"/>
```

```
        <add assembly="System.Design, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=B03F5F7F11D50A3A"/>
```

```
        <add assembly="System.Management, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=B03F5F7F11D50A3A"/>
```

```
        <add assembly="System.Windows.Forms, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=B77A5C561934E089"/>
```

```
        <add assembly="System.ServiceProcess, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=B03F5F7F11D50A3A"/>
```

```
        <add assembly="System.Security, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=B03F5F7F11D50A3A"/>
```

```
        <add assembly="System.Configuration.Install, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=B03F5F7F11D50A3A"/></assemblies></compilation>
```

```
<!--
```

通过 `<authentication>` 节可以配置 ASP.NET 使用的

安全身份验证模式，

以标识传入的用户。

-->

```
<authentication mode="Windows"/>
```

```
<!--
```

如果在执行请求的过程中出现未处理的错误，

则通过 `<customErrors>` 节可以配置相应的处理步骤。具体说来，

开发人员通过该节可以配置

要显示的 `html` 错误页



以代替错误堆栈跟踪。

```
<customErrors mode="RemoteOnly" defaultRedirect="GenericErrorPage.htm">

    <error statusCode="403" redirect="NoAccess.htm" />

    <error statusCode="404" redirect="FileNotFound.htm" />

</customErrors>

-->

<customErrors mode="Off"/>

</system.web>

</configuration>
```

## B、支付请求范例

### //1、生成订单对象

```
com.hitrust.trustpay.client.b2c.Order tOrder = new com.hitrust.trustpay.client.b2c.Order();

tOrder.ExpiredDate = int.Parse(txtExpiredDate.Text); //设定订单有效期

tOrder.OrderNo = txtOrderNo.Text;           //设定订单编号 （必要信息）

tOrder.OrderDesc = txtOrderDesc.Text;       //设定订单说明

tOrder.OrderDate = txtOrderDate.Text;       //设定订单日期 （必要信息 - YYYY/MM/DD）

tOrder.OrderTime = txtOrderTime.Text;       //设定订单时间 （必要信息 - HH:MM:SS）

tOrder.OrderAmount = double.Parse(txtOrderAmount.Text); //设定订单金额 （必要信息）

tOrder.OrderURL = txtOrderURL.Text;         //设定订单网址
```

### //2、生成订单对象，并将订单明细加入定单中（可选信息）

```
com.hitrust.trustpay.client.b2c.OrderItem tOrderItem = new
com.hitrust.trustpay.client.b2c.OrderItem();

tOrderItem.ProductID = "IP000001";

tOrderItem.ProductName = "中国移动 IP 卡";

tOrderItem.UnitPrice = 1.00;

tOrderItem.Qty = 1;

tOrder.addOrderItem(tOrderItem);

tOrderItem = new com.hitrust.trustpay.client.b2c.OrderItem();

tOrderItem.ProductID = "IP000002";

tOrderItem.ProductName = "网通 IP 卡";

tOrderItem.UnitPrice = 1.00;

tOrderItem.Qty = 2;

tOrder.addOrderItem(tOrderItem);
```

### //3、生成支付请求对象

#### //4、传送支付请求并取得支付网址



中国农业银行  
AGRICULTURAL BANK OF CHINA

农行网上支付平台

商户接口编程指南– ASP.NET2.0 Edition – V1.0.7

## C、支付结果接收范例

**//1、取得 MSG 参数，并利用此参数值生成支付结果对象**

```
com.hitrust.trustpay.client.b2c.PaymentResult tResult =  
  
    new com.hitrust.trustpay.client.b2c.PaymentResult();  
  
tResult.init(Request["MSG"]);
```

**//2、判断支付结果状态，进行后续操作**

```
StringBuilder strMessage = new StringBuilder("");  
  
if (tResult.isSuccess())
```

```
{
```

**//3、支付成功**

```
    strMessage.Append("TrxType"           = [" + tResult.getValue("TrxType") + "]<br/>");  
  
    strMessage.Append("OrderNo"           = [" + tResult.getValue("OrderNo") + "]<br/>");  
  
    strMessage.Append("Amount"            = [" + tResult.getValue("Amount") + "]<br/>");  
  
    strMessage.Append("BatchNo"           = [" + tResult.getValue("BatchNo") + "]<br/>");  
  
    strMessage.Append("VoucherNo"         = [" + tResult.getValue("VoucherNo") + "]<br/>");  
  
    strMessage.Append("HostDate"          = [" + tResult.getValue("HostDate") + "]<br/>");  
  
    strMessage.Append("HostTime"          = [" + tResult.getValue("HostTime") + "]<br/>");  
  
    strMessage.Append("MerchantRemarks" = [" + tResult.getValue("MerchantRemarks") + "]<br/>");  
  
    strMessage.Append("PayType"           = [" + tResult.getValue("PayType") + "]<br/>");  
  
    strMessage.Append("NotifyType"        = [" + tResult.getValue("NotifyType") + "]<br/>");
```

```
}
```

```
else
```

```
{
```

**//4、支付失败**

```
strMessage.Append("ReturnCode    = [" + tResult.ReturnCode + "]<br/>");  
  
strMessage.Append("ErrorMessage = [" + tResult.ErrorMessage + "]<br/>");  
  
}  
  
Response.Write(strMessage.ToString());
```

## D、从服务器直接接收支付结果页面范例

```
<%@ Page Language="C#" %>

<script runat="server">

    private string tMerchantPage = "";

    protected void Page_Load(object sender, EventArgs e)

    {

        //1、取得 MSG 参数，并利用此参数值生成支付结果对象

        com.hitrust.trustpay.client.b2c.PaymentResult tResult =

            new com.hitrust.trustpay.client.b2c.PaymentResult();

        tResult.init(Request["MSG"]);

        //2、判断支付结果状态，进行后续操作

        if (tResult.isSuccess())

        {

            //3、支付成功

            tMerchantPage = "http://172.30.7.117/Demo/CustomerPage.aspx?请传入必要的参数"

        }

        else

        {

            //4、支付失败

            tMerchantPage = "http://172.30.7.117/Demo/MerchantFailure.aspx?请传入必要的参数"

        }

    }

</script>

<!--

<URL><%= tMerchantPage %></URL>
```

```
-->

<html xmlns="http://www.w3.org/1999/xhtml" >

<head>

    <meta http-equiv="refresh" content="0; url='<%= tMerchantPage %>'">

</head>

</html>
```



## E、退货交易范例

### //1、生成退货请求对象

```
com.hitrust.trustpay.client.b2c.RefundRequest tRequest =  
    new com.hitrust.trustpay.client.b2c.RefundRequest();  
  
tRequest.OrderNo = txtOrderNo.Text;           //订单号    (必要信息)  
  
tRequest.TrxAmount = double.Parse(txtTrxAmount.Text);    //退货金额 (必要信息)
```

### //2、传送退货请求并取得退货结果

```
com.hitrust.trustpay.client.TrxResponse tTrxResponse = tRequest.postRequest();
```

### //3、判断退货结果状态，进行后续操作

```
StringBuilder strMessage = new StringBuilder("");
```

```
if (tTrxResponse.isSuccess())
```

```
{
```

#### //4、退货成功

```
strMessage.Append("OrderNo    = [" + tTrxResponse.getValue("OrderNo") + "]<br/>");  
  
strMessage.Append("TrxAmount  = [" + tTrxResponse.getValue("TrxAmount") + "]<br/>");  
  
strMessage.Append("BatchNo    = [" + tTrxResponse.getValue("BatchNo") + "]<br/>");  
  
strMessage.Append("VoucherNo  = [" + tTrxResponse.getValue("VoucherNo") + "]<br/>");  
  
strMessage.Append("HostDate   = [" + tTrxResponse.getValue("HostDate") + "]<br/>");  
  
strMessage.Append("HostTime   = [" + tTrxResponse.getValue("HostTime") + "]<br/>");
```

```
}
```

```
else
```

```
{
```

#### //5、退货失败

```
strMessage.Append("ReturnCode = [" + tTrxResponse.ReturnCode + "]<br/>");
```

```
strMessage.Append("ErrorMessage = [" + tTrxResponse.ErrorMessage + "<br/>");  
  
}  
  
Response.Write(strMessage.ToString());
```

## F、订单查询交易范例

### //1、生成商户订单查询请求对象

```
com.hitrust.trustpay.client.b2c.QueryOrderRequest tRequest =  
    new com.hitrust.trustpay.client.b2c.QueryOrderRequest();  
  
tRequest.OrderNo = txtOrderNo.Text; //订单号    (必要信息)  
  
if (txtDetailQuery.Text == "0") //是否查询详细信息 (必要信息)  
{  
    tRequest.EnableDetailQuery = false;  
}  
  
else if (txtDetailQuery.Text == "1")  
{  
    tRequest.EnableDetailQuery = true;  
}  
  
else  
{  
    Response.Write("DetailQuery (0: 状态查询; 1: 详细查询) 设置不正确");  
    return;  
}
```

### //2、传送商户订单查询请求并取得订单查询结果

```
com.hitrust.trustpay.client.TrxResponse tTrxResponse = tRequest.postRequest();
```

### //3、判断商户订单查询结果状态，进行后续操作

```
StringBuilder strMessage = new StringBuilder("");  
  
if (tTrxResponse.isSuccess())  
{
```

### //4、生成订单对象

```
com.hitrust.trustpay.client.b2c.Order tOrder =  
  
    new com.hitrust.trustpay.client.b2c.Order();  
  
tOrder.initByString(tTrxResponse.getValue("Order"));  
  
strMessage.Append("OrderNo      = [" + tOrder.OrderNo + "]<br/>");  
  
strMessage.Append("OrderAmount  = [" + tOrder.OrderAmount + "]<br/>");  
  
strMessage.Append("OrderDesc    = [" + tOrder.OrderDesc + "]<br/>");  
  
strMessage.Append("OrderDate    = [" + tOrder.OrderDate + "]<br/>");  
  
strMessage.Append("OrderTime    = [" + tOrder.OrderTime + "]<br/>");  
  
strMessage.Append("OrderURL     = [" + tOrder.OrderURL + "]<br/>");  
  
strMessage.Append("PayAmount    = [" + tOrder.PayAmount + "]<br/>");  
  
strMessage.Append("RefundAmount = [" + tOrder.RefundAmount + "]<br/>");  
  
strMessage.Append("OrderStatus  = [" + tOrder.OrderStatus + "]<br/>");  
  
  
//5、取得订单明细  
  
foreach (com.hitrust.trustpay.client.b2c.OrderItem tOrderItem in tOrder.OrderItems)  
{  
  
    strMessage.Append("ProductID   = [" + tOrderItem.ProductID + "]<br/>");  
  
    strMessage.Append("ProductName = [" + tOrderItem.ProductName + "]<br/>");  
  
    strMessage.Append("UnitPrice   = [" + tOrderItem.UnitPrice + "]<br/>");  
  
    strMessage.Append("Qty         = [" + tOrderItem.Qty + "]<br/>");  
  
}  
  
}  
  
else  
{  
  
//6、商户订单查询失败  
  
strMessage.Append("ReturnCode   = [" + tTrxResponse.ReturnCode + "]<br/>");  
  
strMessage.Append("ErrorMessage = [" + tTrxResponse.ErrorMessage + "]<br/>");
```

```
}  
  
Response.Write(strMessage.ToString());
```

## G、交易对账单下载范例

### //1、生成商户对账单下载请求对象

```
com.hitrust.trustpay.client.b2c.SettleRequest tRequest =  
    new com.hitrust.trustpay.client.b2c.SettleRequest();  
  
tRequest.SettleDate = txtSettleDate.Text;    //对账日期 YYYY/MM/DD （必要信息）  
  
tRequest.SettleType = "TRX";                //对账类型 （必要信息）
```

### //2、传送商户对账单下载请求并取得对账单

```
com.hitrust.trustpay.client.TrxResponse tTrxResponse = tRequest.postRequest();
```

### //3、判断商户对账单下载结果状态，进行后续操作

```
StringBuilder strMessage = new StringBuilder("");  
  
if (tTrxResponse.isSuccess())  
{
```

### //4、商户对账单下载成功，生成对账单对象

```
com.hitrust.trustpay.client.b2c.SettleFile tSettleFile =  
    new com.hitrust.trustpay.client.b2c.SettleFile();  
  
tSettleFile.init(tTrxResponse);  
  
strMessage.Append("SettleDate      = [" + tSettleFile.SettleDate + "]<br/>");  
  
strMessage.Append("SettleType      = [" + tSettleFile.SettleType + "]<br/>");  
  
strMessage.Append("NumOfPayments  = [" + tSettleFile.NumOfPayments + "]<br/>");  
  
strMessage.Append("SumOfPayAmount = [" + tSettleFile.SumOfPayAmount + "]<br/>");  
  
strMessage.Append("NumOfRefunds   = [" + tSettleFile.NumOfRefunds + "]<br/>");  
  
strMessage.Append("SumOfRefundAmount = [" + tSettleFile.SumOfRefundAmount + "]<br/>");
```

### //5、取得对账单明细

```
ArrayList tRecords = tSettleFile.DetailRecords;
```

```
for (int i = 0; i < tRecords.Count; i++)  
{  
    strMessage.Append("Record-" + i + " = [" + tRecords[i] + "]<br/>");  
}  
}  
  
else  
{  
    //6、商户账单下载失败  
  
    strMessage.Append("ReturnCode    = [" + tTrxResponse.ReturnCode + "]<br/>");  
    strMessage.Append("ErrorMessage = [" + tTrxResponse.ErrorMessage + "]<br/>");  
}  
  
Response.Write(strMessage.ToString());
```

## H、指定时间段交易对账单下载

```
//1、生成商户对账单下载请求对象  
  
com.hitrust.trustpay.client.b2c.SettleRequest tRequest =  
    new com.hitrust.trustpay.client.b2c.SettleRequest();  
  
tRequest.SettleDate = txtSettleDate.Text;           //对账日期 YYYY/MM/DD （必要信息）  
  
tRequest.SettleStartHour = txtSettleStartHour.Text; //对账开始时间（必要信息）  
  
tRequest.SettleEndHour = txtSettleEndHour.Text;     //对账截止时间（必要信息）  
  
tRequest.SettleType = "TRXBYHOUR";                  //对账类型 （必要信息）  
  
  
//2、传送商户对账单下载请求并取得对账单  
  
com.hitrust.trustpay.client.TrxResponse tTrxResponse = tRequest.postRequest();
```

//3、判断商户对账单下载结果状态，进行后续操作

```
StringBuilder strMessage = new StringBuilder("");

if (tTrxResponse.isSuccess())

{

    //4、商户对账单下载成功，生成对账单对象

    com.hitrust.trustpay.client.b2c.SettleFile tSettleFile =

        new com.hitrust.trustpay.client.b2c.SettleFile();

    tSettleFile.init(tTrxResponse);

    strMessage.Append("SettleDate= [" + tSettleFile.SettleDate + "]<br/>");

    strMessage.Append("SettleType= [" + tSettleFile.SettleType + "]<br/>");

    strMessage.Append("NumOfPayments      = [" + tSettleFile.NumOfPayments + "]<br/>");

    strMessage.Append("SumOfPayAmount    = [" + tSettleFile.SumOfPayAmount + "]<br/>");

    strMessage.Append("NumOfRefunds      = [" + tSettleFile.NumOfRefunds + "]<br/>");

    strMessage.Append("SumOfRefundAmount = [" + tSettleFile.SumOfRefundAmount + "]<br/>");

}
```

//5、取得对账单明细

```
ArrayList tRecords = tSettleFile.DetailRecords;

for (int i = 0; i < tRecords.Count; i++)

{

    strMessage.Append("Record-" + i + " = [" + tRecords[i] + "]<br/>");

}

else
```

//6、商户账单下载失败

```
strMessage.Append("ReturnCode    = [" + tTrxResponse.ReturnCode + "]<br/>");

strMessage.Append("ErrorMessage = [" + tTrxResponse.ErrorMessage + "]<br/>");

}
```



```
Response.Write(strMessage.ToString());
```

## I、身份验证请求范例

//1、取得身份验证请求所需要的信息

//2、生成身份验证请求对象

```
com.hitrust.trustpay.client.b2c.IdentityVerifyRequest tRequest =  
    new com.hitrust.trustpay.client.b2c.IdentityVerifyRequest();  
  
tRequest.BankCardNo = txtBankCardNo.Text;           //银行帐号           (必要信息)  
tRequest.CertificateNo = txtCertificateNo.Text;       //证件号码           (必要信息)  
tRequest.CertificateType = ddlCertificateType.SelectedValue; //证件类型           (必要信息)  
tRequest.ResultNotifyURL = txtResultNotifyURL.Text;   //身份验证回传网址 (必要信息)  
tRequest.RequestDate = txtRequestDate.Text;           //验证请求日期 (必要信息 - YYYY/MM/DD)  
tRequest.RequestTime = txtRequestTime.Text;           //验证请求时间 (必要信息 - HH:MM:SS)
```

//3、传送身份验证请求并取得支付网址

```
com.hitrust.trustpay.client.TrxResponse tTrxResponse = tRequest.postRequest();  
  
StringBuilder strMessage = new StringBuilder("");  
  
if (tTrxResponse.isSuccess())
```

```
{
```

//4、身份验证请求提交成功，将客户端导向身份验证页面

```
Response.Redirect(tTrxResponse.getValue("VerifyURL"));
```

```
}
```

```
else
```

```
{
```

//5、身份验证请求提交失败，商户自定后续动作

```
strMessage.Append("ReturnCode = [" + tTrxResponse.ReturnCode + "<br/>");
```

```
strMessage.Append("ErrorMessage = [" + tTrxResponse.ErrorMessage + "]<br/>");  
  
}  
  
Response.Write(strMessage.ToString());
```

## J、身份验证结果接收范例

//1、取得 MSG 参数，并利用此参数值生成验证结果对象

```
com.hitrust.trustpay.client.b2c.PaymentResult tResult =  
    new com.hitrust.trustpay.client.b2c.PaymentResult();  
  
tResult.init(Request["MSG"]);
```

//2、判断验证结果状态，进行后续操作

```
StringBuilder strMessage = new StringBuilder("");
```

```
if (tResult.isSuccess())
```

```
{
```

//3、验证成功

```
strMessage.Append("户名 = [" + tResult.getValue("AccountName") + "]<br/>");
```

```
}
```

```
else
```

```
{
```

//4、验证失败

```
strMessage.Append("ReturnCode = [" + tResult.ReturnCode + "]<br/>");
```

```
strMessage.Append("ErrorMessage = [" + tResult.ErrorMessage + "]<br/>");
```

```
}
```

```
Response.Write(strMessage.ToString());
```

## K、批量退款发送请求范例

```
StringBuilder strMessage = new StringBuilder("");
```

## //1、取得退款所需要的信息

```
String tTotalCount = Request["TotalCount"];

String tTotalAmount = Request["TotalAmount"];

String tRemark = Request["Remark"];
```

## //取得列表项

```
String[] orderno_arr = null;

String[] orderamount_arr = null;

int iTTotalCount = int.Parse(tTotalCount);

double iTTotalAmount = double.Parse(tTotalAmount);

if (iTTotalCount == 1)

{

    String orderno = Request["orderno"];

    String orderamount = Request["orderamount"];

    orderno_arr = new String[] { orderno };

    orderamount_arr = new String[] { orderamount };

}

else

{

    orderno_arr = Request["orderno"].Split(',');

    orderamount_arr = Request["orderamount"].Split(',');

}
```

```
ArrayList tOrderList = new ArrayList();

for (int i = 0; i < orderno_arr.Length; i++)

{

    String[] torder = new String[2];

    torder[0] = orderno_arr[i];

    torder[1] = orderamount_arr[i];

    tOrderList.Add(torder);

}
```

#### //2、生成退款请求对象

```
com.hitrust.trustpay.client.b2c.OverdueRefundRequest tOverdueRequest =

    new com.hitrust.trustpay.client.b2c.OverdueRefundRequest();

tOverdueRequest.TotalCount = iTotalCount;           //总笔数   （必要信息）

tOverdueRequest.TotalAmount = iTotalAmount;         //总金额   （必要信息）

tOverdueRequest.Remark = tRemark;                   //备注

tOverdueRequest.OrderDital = tOrderList;
```

#### //3、传送退款请求并取得结果

```
com.hitrust.trustpay.client.TrxResponse tResponse = tOverdueRequest.postRequest();
```

#### //4、判断退款结果状态，进行后续操作

```
if (tResponse.isSuccess())

{
```

#### //5、退款成功

```
strMessage.Append("TrxType    = [" + tResponse.GetValue("TrxType") + "]<br/>");

strMessage.Append("TotalCount  = [" + tResponse.GetValue("TotalCount") + "]<br/>");

strMessage.Append("TotalAmount = [" + tResponse.GetValue("TotalAmount") + "]<br/>");

strMessage.Append("SerialNumber = [" + tResponse.GetValue("SerialNumber") + "]<br/>");

strMessage.Append("HostDate   = [" + tResponse.GetValue("HostDate") + "]<br/>");

strMessage.Append("HostTime   = [" + tResponse.GetValue("HostTime") + "]<br/>");

strMessage.Append("ResultMessage    = [" + tResponse.ErrorMessage + "]<br/>");

}

else

{

    //6 退款失败

    strMessage.Append("ReturnCode    = [" + tResponse.ReturnCode + "]<br/>");

    strMessage.Append("ErrorMessage = [" + tResponse.ErrorMessage + "]<br/>");

}

Response.Write(strMessage.ToString());
```

## L、批量退款交易结果查询请求范例

//1、取得退款结果查询请求所需要的信息

//2、生成退款结果查询请求对象

```
com.hitrust.trustpay.client.b2c.QueryOverdueRefundRequest tQueryBatchRequest =

    new com.hitrust.trustpay.client.b2c.QueryOverdueRefundRequest();

tQueryBatchRequest.SerialNumber = txtSerialNumber.Text; //设定批量退款结果查询请求的流水号（必要信息）
```

//3、传送退款结果查询请求并取得结果

```
com.hitrust.trustpay.client.TrxResponse tResponse = tQueryBatchRequest.postRequest();
```

**//4、判断退款结果查询状态，进行后续操作**

```
StringBuilder strMessage = new StringBuilder("");
```

```
if (tResponse.isSuccess())
```

```
{
```

**//5、生成批量对象**

```
com.hitrust.trustpay.client.b2c.OverdueBatch tBatch =
```

```
new com.hitrust.trustpay.client.b2c.OverdueBatch(
```

```
new com.hitrust.trustpay.client.XMLDocument(tResponse.getValue("QueryOverdueRefund")));
```

```
strMessage.Append("SerialNumber = [" + tBatch.SerialNumber + "]<br/>");
```

```
strMessage.Append("RefundAmount = [" + tBatch.RefundAmount + "]<br/>");
```

```
strMessage.Append("RefundCount = [" + tBatch.RefundCount + "]<br/>");
```

```
strMessage.Append("BatchStatus = [" + tBatch.Status + "]<br/>");
```

**//6、取得订单明细**

```
ArrayList tOrders = tBatch.Orders;
```

```
for (int i = 0; i < tOrders.Count; i++)
```

```
{
```

```
com.hitrust.trustpay.client.b2c.Order tOrder =
```

```
(com.hitrust.trustpay.client.b2c.Order)tOrders[i];
```

```
strMessage.Append("OrderNo = [" + tOrder.OrderNo + "]<br/>");
```

```
strMessage.Append("RefundAmount = [" + tOrder.RefundAmount + "]<br/>");
```

```
strMessage.Append("OrderStatus = [" + tOrder.OrderStatus + "]<br/>");
```

```
strMessage.Append("OrderDesc = [" + tOrder.OrderDesc + "]<br/>");
```

```
}
```

```
}
```

```
else
```

```
{
```

**//7、退款结果查询失败**

```
strMessage.Append("ReturnCode    = [" + tResponse.ReturnCode + "]<br/>");  
  
strMessage.Append("ErrorMessage = [" + tResponse.ErrorMessage + "]<br/>");  
  
}  
  
Response.Write(strMessage.ToString());
```

## M、委托扣款签约交易请求范例

//1、取得委托扣款签约请求所需要的信息

//2、生成委托扣款签约请求对象

```
com.hitrust.trustpay.client.b2c.B2CAgentSignContractRequest tRequest =  
  
    new com.hitrust.trustpay.client.b2c.B2CAgentSignContractRequest();  
  
tRequest.OrderNo = txtOrderNo.Text;                //订单编号（必要信息）  
  
tRequest.CertificateNo = txtCertificateNo.Text;      //证件号码      （必要信息）  
  
tRequest.CertificateType = ddlCertificateType.SelectedValue; //证件类型      （必要信息）  
  
tRequest.ResultNotifyURL = txtResultNotifyURL.Text; //身份验证回传网址（必要信息）  
  
tRequest.RequestDate = txtRequestDate.Text;         //验证请求日期 （必要信息 - YYYY/MM/DD）  
  
tRequest.RequestTime = txtRequestTime.Text;         //验证请求时间 （必要信息 - HH:MM:SS）  
  
tRequest.InvalidDate = txtInvalidDate.Text;  
  
tRequest.NotifyType = txtNotifyType.Text;  
  
tRequest.CardType = txtCardType.Text;               //农行卡类型
```

//3、传送委托扣款签约请求并取得签约网址

```
com.hitrust.trustpay.client.TrxResponse tTrxResponse = tRequest.postRequest();  
  
StringBuilder strMessage = new StringBuilder("");  
  
if (tTrxResponse.isSuccess())  
  
    {  
  
        //4、委托扣款签约请求提交成功，将客户端导向签约页面  
  
        Response.Redirect(tTrxResponse.getValue("B2CAgentSignContractURL"));  
  
    }  
  
else
```

```
{  
  
    //5、委托扣款签约请求提交失败，商户自定后续动作  
  
    strMessage.Append("ReturnCode    = [" + tTrxResponse.ReturnCode + "]<br/>");  
  
    strMessage.Append("ErrorMessage = [" + tTrxResponse.ErrorMessage + "]<br/>");  
  
}  
  
Response.Write(strMessage.ToString());
```

## N、委托扣款解约交易请求范例

```
//1、取得委托扣款解约请求所需要的信息  
  
//2、生成委托扣款解约请求对象  
  
com.hitrust.trustpay.client.b2c.B2CAgentUnsignContractRequest tRequest =  
  
    new com.hitrust.trustpay.client.b2c.B2CAgentUnsignContractRequest();  
  
tRequest.OrderNo = txtOrderNo.Text;                //订单编号（必要信息）  
  
tRequest.CertificateNo = txtCertificateNo.Text;      //证件号码        （必要信息）  
  
tRequest.CertificateType = ddlCertificateType.SelectedValue; //证件类型        （必要信息）  
  
tRequest.ResultNotifyURL = txtResultNotifyURL.Text; //身份验证回传网址（必要信息）  
  
tRequest.RequestDate = txtRequestDate.Text;         //验证请求日期  （必要信息 - YYYY/MM/DD）  
  
tRequest.RequestTime = txtRequestTime.Text;         //验证请求时间  （必要信息 - HH:MM:SS）  
  
tRequest.NotifyType = txtNotifyType.Text;  
  
tRequest.AgentSignNo = txtAgentSignNo.Text;  
  
  
//3、传送委托扣款解约请求并取得签约网址  
  
com.hitrust.trustpay.client.TrxResponse tTrxResponse = tRequest.postRequest();  
  
StringBuilder strMessage = new StringBuilder("");  
  
if (tTrxResponse.isSuccess())
```



```
{  
  
    //4、委托扣款解约请求提交成功，将客户端导向签约页面  
  
    Response.Redirect(tTrxResponse.GetValue("B2CAgentSignContractURL"));  
  
}  
  
else  
  
{  
  
    //5、委托扣款解约请求提交失败，商户自定后续动作  
  
    strMessage.Append("ReturnCode    = [" + tTrxResponse.ReturnCode + "]<br/>");  
  
    strMessage.Append("ErrorMessage = [" + tTrxResponse.ErrorMessage + "]<br/>");  
  
}  
  
Response.Write(strMessage.ToString());  
}
```

## 0、委托扣款单笔代扣交易请求范例

```
//1、取得委托扣款单笔代扣请求所需要的信息  
  
//2、生成委托扣款单笔代扣请求对象  
  
com.hitrust.trustpay.client.b2c.B2CAgentPaymentRequest tRequest =  
  
    new com.hitrust.trustpay.client.b2c.B2CAgentPaymentRequest();  
  
tRequest.OrderNo = txtOrderNo.Text;                //订单编号（必要信息）  
  
tRequest.ExpiredDate = int.Parse(txtExpiredDate.Text); //订单有效期  
  
tRequest.RequestDate = txtRequestDate.Text;         //验证请求日期（必要信息 - YYYY/MM/DD）  
  
tRequest.RequestTime = txtRequestTime.Text;         //验证请求时间（必要信息 - HH:MM:SS）  
  
tRequest.Currency = txtCurrency.Text;  
  
tRequest.Amount = txtAmount.Text;  
  
tRequest.ProductId = txtProductId.Text;
```

```
tRequest.ProductName = txtProductName.Text;

tRequest.Quantity = txtQuantity.Text;

tRequest.CertificateNo = txtCertificateNo.Text;

tRequest.AgentSignNo = txtAgentSignNo.Text;

//3、传送委托扣款单笔代扣请求

com.hitrust.trustpay.client.TrxResponse tTrxResponse = tRequest.postRequest();

StringBuilder strMessage = new StringBuilder("");

if (tTrxResponse.isSuccess())

{

    //4、委托扣款单笔代扣请求提交成功

    strMessage.Append("Success!!!");

}

else

{

    //5、委托扣款单笔代扣请求提交失败，商户自定后续动作

    strMessage.Append("ReturnCode    = [" + tTrxResponse.ReturnCode + "<br>");

    strMessage.Append("ErrorMessage = [" + tTrxResponse.ErrorMessage + "<br>");

}

Response.Write(strMessage.ToString());
```

## P、委托扣款批量交易请求范例

```
StringBuilder strMessage = new StringBuilder("");
```

## //1、取得委托扣款批量需要的信息

```
string batchNo = Request["BatchNo"];

string batchDate = Request["BatchDate"];

string agentCount = Request["AgentCount"];

string agentAmount = Request["AgentAmount"];

string[] orderno_arr = null;

string[] orderamount_arr = null;

string[] expireddate_arr = null;

string[] certificatenos_arr = null;

string[] contractid_arr = null;

string[] productid_arr = null;

string[] productname_arr = null;

string[] productnum_arr = null;

int iBatchSize = int.Parse(agentCount);

if (iBatchSize == 1)

{

    string orderno = Request["orderno"];

    string orderamount = Request["orderamount"];

    string expireddate = Request["expireddate"];

    string certificatenos = Request["certificatenos"];

    string contractid = Request["contractid"];

    string productid = Request["productid"];

    string productname = Request["productname"];

    string productnum = Request["productnum"];

    orderno_arr = new string[] { orderno };

    orderamount_arr = new string[] { orderamount };
```

```
expireddate_arr = new string[] { expireddate };

certificateno_arr = new string[] { certificateno };

contractid_arr = new string[] { contractid };

productid_arr = new string[] { productid };

productname_arr = new string[] { productname };

productnum_arr = new string[] { productnum };

}

else

{

    orderno_arr = Request["orderno"].Split(',');

    orderamount_arr = Request["orderamount"].Split(',');

    expireddate_arr = Request["expireddate"].Split(',');

    certificateno_arr = Request["certificateno"].Split(',');

    contractid_arr = Request["contractid"].Split(',');

    productid_arr = Request["productid"].Split(',');

    productname_arr = Request["productname"].Split(',');

    productnum_arr = Request["productnum"].Split(',');

}
```

## //2、生成委托扣款批量请求对象

```
com.hitrust.trustpay.client.b2c.AgentBatch iAgentBatch =

    new com.hitrust.trustpay.client.b2c.AgentBatch();

iAgentBatch.BatchNo = batchNo;

iAgentBatch.BatchDate = batchDate;

iAgentBatch.AgentAmount = double.Parse(agentAmount);

iAgentBatch.AgentCount = int.Parse(agentCount);

com.hitrust.trustpay.client.b2c.B2CAgentBatchRequest aB2CAgentBatchRequest =

    new com.hitrust.trustpay.client.b2c.B2CAgentBatchRequest();
```

```
aB2CAgentBatchRequest.AgentBatch = iAgentBatch;

//按照顺序号决定每个批次包含多少 AgentBatchDetail

for (int i = 0; i < orderno_arr.Length; i++)
{
    com.hitrust.trustpay.client.b2c.AgentBatchDetail aAgentBatchDetail =
        new com.hitrust.trustpay.client.b2c.AgentBatchDetail();

    aAgentBatchDetail.OrderNo = orderno_arr[i];

    aAgentBatchDetail.OrderAmount = double.Parse(orderamount_arr[i]);

    aAgentBatchDetail.ExpiredDate = int.Parse(expireddate_arr[i]);

    aAgentBatchDetail.CertificateNo = certificateno_arr[i];

    aAgentBatchDetail.ContractID = contractid_arr[i];

    aAgentBatchDetail.ProductID = productid_arr[i];

    aAgentBatchDetail.ProductName = productname_arr[i];

    aAgentBatchDetail.ProductNum = int.Parse(productnum_arr[i]);

    aB2CAgentBatchRequest.addAgentBatchDetail(aAgentBatchDetail);
}

com.hitrust.trustpay.client.TrxResponse tResponse = aB2CAgentBatchRequest.postRequest();

if (tResponse.isSuccess())
{
    strMessage.Append("批量受理成功!");
}
else
{
    strMessage.Append("ReturnCode    = [" + tResponse.ReturnCode + "]<br/>");

    strMessage.Append("ErrorMessage = [" + tResponse.ErrorMessage + "]<br/>");
}
```

```
}  
  
Response.Write(strMessage.ToString());
```

## Q、委托扣款批量结果查询交易请求范例

//1、取得赎回、分红、托收请求信息

//2、生成请求对象

```
com.hitrust.trustpay.client.b2c.B2CAgentBatchQueryRequest tRequest =  
    new com.hitrust.trustpay.client.b2c.B2CAgentBatchQueryRequest();  
  
tRequest.BatchNo = txtBatchNo.Text; //请求批次号      (必要信息)  
  
tRequest.BatchDate = txtBatchDate.Text; //请求日期      YYYY/MM/DD
```

//3、传送请求并取得结果

```
com.hitrust.trustpay.client.TrxResponse tResponse = tRequest.postRequest();
```

//4、判断结果状态，进行后续操作

```
StringBuilder strMessage = new StringBuilder("");  
  
if (tResponse.isSuccess())  
{
```

//5、请求成功

//5、商户对账单下载成功，生成对账单对象

```
com.hitrust.trustpay.client.b2c.AgentBatch tAgentBatch =  
    new com.hitrust.trustpay.client.b2c.AgentBatch(  
        new com.hitrust.trustpay.client.XMLDocument(tResponse.getValue("AgentBatch")));  
  
strMessage.Append("BatchNo = [" + tAgentBatch.BatchNo + "]<br/>");  
  
strMessage.Append("BatchDate = [" + tAgentBatch.BatchDate + "]<br/>");  
  
strMessage.Append("BatchTime = [" + tAgentBatch.BatchTime + "]<br/>");  
  
strMessage.Append("AgentAmount = [" + tAgentBatch.AgentAmount + "]<br/>");  
  
strMessage.Append("AgentCount = [" + tAgentBatch.AgentCount + "]<br/>");
```

```
strMessage.Append("BatchStatus    = [" + tAgentBatch.BatchStatus + "]<br/>");  
  
strMessage.Append("BatchStatusZH  = [" +  
    tAgentBatch.getBatchStatusChinese(tAgentBatch.BatchStatus) + "]<br/>");
```

#### //6、取得对账单明细

```
ArrayList tAgentBatchDetails = tAgentBatch.AgentBatchDetail;  
  
for (int i = 0; i < tAgentBatchDetails.Count; i++)  
{  
    com.hitrust.trustpay.client.b2c.AgentBatchDetail tAgentBatchDetail =  
        (com.hitrust.trustpay.client.b2c.AgentBatchDetail)tAgentBatchDetails[i];  
  
    strMessage.Append("SeqNo      = [" + (i + 1) + "],");  
  
    strMessage.Append("OrderNo    = [" + tAgentBatchDetail.OrderNo + "],");  
  
    strMessage.Append("OrderAmount = [" + tAgentBatchDetail.OrderAmount + "],");  
  
    strMessage.Append("CertificateNo = [" + tAgentBatchDetail.CertificateNo + "],");  
  
    strMessage.Append("ContractID = [" + tAgentBatchDetail.ContractID + "],");  
  
    strMessage.Append("ProductID   = [" + tAgentBatchDetail.ProductID + "],");  
  
    strMessage.Append("ProductName = [" + tAgentBatchDetail.ProductName + "],");  
  
    strMessage.Append("ProductNum  = [" + tAgentBatchDetail.ProductNum + "],");  
  
    strMessage.Append("OrderStatus = [" + tAgentBatchDetail.OrderStatus + "],");  
  
    strMessage.Append("OrderStatusZH = [" +  
        tAgentBatchDetail.getStatusChinese(tAgentBatchDetail.OrderStatus) + "]<br/>");  
}  
}  
  
else  
{  
  
    //6、批量结果查询失败  
  
    strMessage.Append("ReturnCode   = [" + tResponse.ReturnCode + "]<br/>");  
  
    strMessage.Append("ErrorMessage = [" + tResponse.ErrorMessage + "]<br/>");
```

```
}  
  
Response.Write(strMessage.ToString());
```

## R、贷记卡对账单下载交易请求范例

```
//1、取得商户对账单下载所需要的信息  
  
//2、生成商户对账单下载请求对象  
  
com.hitrust.trustpay.client.b2c.SettleRequest tRequest =  
    new com.hitrust.trustpay.client.b2c.SettleRequest();  
  
tRequest.SettleDate = txtSettleDate.Text;           //对账日期 YYYY/MM/DD （必要信息）  
  
tRequest.SettleType = com.hitrust.trustpay.client.b2c.SettleFile.SETTLE_TYPE_CREDIT_TRX; //对账  
类型 （必要信息）  
  
  
//3、传送商户对账单下载请求并取得对账单  
  
com.hitrust.trustpay.client.TrxResponse tResponse = tRequest.postRequest();  
  
  
//4、判断商户对账单下载结果状态，进行后续操作  
  
StringBuilder strMessage = new StringBuilder("");  
  
if (tResponse.isSuccess())  
{  
  
    //5、商户对账单下载成功，生成对账单对象  
  
    com.hitrust.trustpay.client.b2c.SettleFile tSettleFile =  
        new com.hitrust.trustpay.client.b2c.SettleFile();  
  
    tSettleFile.init(tResponse);  
  
    strMessage.Append("SettleDate= [" + tSettleFile.SettleDate + "]<br/>");  
  
    strMessage.Append("SettleType= [" + tSettleFile.SettleType + "]<br/>");  
  
    strMessage.Append("NumOfPayments      = [" + tSettleFile.NumOfPayments + "]<br/>");  
  
    strMessage.Append("SumOfPayAmount    = [" + tSettleFile.SumOfPayAmount + "]<br/>");  
}
```



```
strMessage.Append("NumOfRefunds      = [" + tSettleFile.NumOfRefunds + "]<br/>");

strMessage.Append("SumOfRefundAmount = [" + tSettleFile.SumOfRefundAmount + "]<br/>");

//6、取得对账单明细

ArrayList tRecords = tSettleFile.DetailRecords;

for (int i = 0; i < tRecords.Count; i++)

{

    strMessage.Append("Record-" + i + " = [" + tRecords[i] + "]<br/>");

}

}

else

{

    //7、商户账单下载失败

    strMessage.Append("ReturnCode    = [" + tResponse.ReturnCode + "]<br/>");

    strMessage.Append("ErrorMessage = [" + tResponse.ErrorMessage + "]<br/>");

}

Response.Write(strMessage.ToString());
```

## S、商户通过页面传参数提交表单方式访问农行 B2C 服务范例程序

```
string _ResponseString = "";

protected void btnButton_Click(object sender, EventArgs e)

{
```

//1、生成订单对象

```
com.hitrust.trustpay.client.b2c.Order tOrder = new
com.hitrust.trustpay.client.b2c.Order();

tOrder.OrderNo = txtOrderNo.Text;                                //设定订单编号 （必要信息）

tOrder.ExpiredDate = int.Parse(txtExpiredDate.Text);            //设定订单有效期 （必要信息）

//wdy add 20100812 tiedaobu
```

```

        tOrder.BuyIP = txtBuyIP.Text;                                //IP

        tOrder.OrderDesc = txtOrderDesc.Text;                      //设定订单说明

        tOrder.OrderDate = txtOrderDate.Text;                      //设定订单日期 （必要信息 -
        YYYY/MM/DD)

        tOrder.OrderTime = txtOrderTime.Text;                      //设定订单时间 （必要信息 -
        HH:MM:SS)

        tOrder.OrderAmount = double.Parse(txtOrderAmount.Text); //设定订单金额 （必要信息）

        tOrder.OrderURL = txtOrderURL.Text;                        //设定订单网址


        //2、生成定单订单对象，并将订单明细加入定单中（可选信息）

        com.hitrust.trustpay.client.b2c.OrderItem tOrderItem = new
        com.hitrust.trustpay.client.b2c.OrderItem();

        tOrderItem.ProductID = "IP000001";

        tOrderItem.ProductName = "中国移动 IP 卡";

        tOrderItem.UnitPrice = 1.00;

        tOrderItem.Qty = 1;

        tOrder.addOrderItem(tOrderItem);

        tOrderItem = new com.hitrust.trustpay.client.b2c.OrderItem();

        tOrderItem.ProductID = "IP000002";

        tOrderItem.ProductName = "网通 IP 卡";

        tOrderItem.UnitPrice = 1.00;

        tOrderItem.Qty = 2;

        tOrder.addOrderItem(tOrderItem);


        //3、生成支付请求对象

        com.hitrust.trustpay.client.b2c.PaymentRequest tPaymentRequest = new
        com.hitrust.trustpay.client.b2c.PaymentRequest();

        tPaymentRequest.Order = tOrder;                            //设定支付请求的订单 （必要信
        息)

        tPaymentRequest.ProductType = txtProductType.Text;        //设定商品种类 （必要信息）

        tPaymentRequest.PaymentType = txtPaymentType.Text;        //设定支付类型
  
```

```
tPaymentRequest.NotifyType = txtNotifyType.Text;           //设定支付结果通知类型

tPaymentRequest.ResultNotifyURL = txtResultNotifyURL.Text;  //设定支付结果回传网址（必要
信息）

tPaymentRequest.MerchantRemarks = txtMerchantRemarks.Text; //设定商户备注信息

tPaymentRequest.PayLinkType = txtPaymentLinkType.Text;      //设定支付接入方式

//PaymentRequest.PAY_LINK_TYPE_NET;1:internet 网络接入

//PaymentRequest.PAY_LINK_TYPE_TEL;2:手机网络接入

//PaymentRequest.PAY_LINK_TYPE_TV;3:数字电视网络接入

//PaymentRequest.PAY_LINK_TYPE_IC;4:智能客户端

//PaymentRequest.PAY_LINK_TYPE_MOBILE;5:电话网络

//

_ResponseString = "";

try

{

    string tSignature = tPaymentRequest.genSignature(1);

    string sTrustPayIETrxURL =
com.hitrust.trustpay.client.MerchantConfig.TrustPayIETrxURL;

    string sErrorUrl = com.hitrust.trustpay.client.MerchantConfig.MerchantErrorURL;

    form1.Visible = false;

    _ResponseString = "<form name=\"form2\" method=\"post\" action=\"\" +
sTrustPayIETrxURL + \"> \r\n\" +

                                "<input type=\"hidden\" name=\"Signature\" value=\"\" +
tSignature + \"> \r\n\" +

                                "<input type=\"hidden\" name=\"errorPage\" value=\"\" +
sErrorUrl + \"> \r\n\" +

                                "<input type=\"submit\" value=\"提交\"></form><br/> \r\n\" +

                                "<a href='MerchantPaymentIE.aspx'>回商户首页</a> \r\n\";

}

catch (com.hitrust.trustpay.client.TrxException ex)
```

```
{  
  
    form1.Visible = false;
```

```
    _ResponseString = "<center><br/> \r\n" +  
  
        "ReturnCode    = [" + ex.Code + "]"<br/> \r\n" +  
  
        "ErrorMessage = [" + ex.Message + "]"<br/><br/> \r\n" +  
  
        "<a href='MerchantPaymentIE.aspx'>回商户首页</a></center> \r\n";  
  
    }  
  
}
```

//对于表单提交：参数 1- Signature 通过接口程序自动生成，参数 2- errorPage，这是农行 b2c 支付平台抛出错误返回给商户的错误处理页面，需商户手动设置；表单提交时，将会调用农行 ReceiveMerchantIERequestServlet 进行处理。商户 前期测试时，可以连接 <https://www.test.95599.cn/b2c/b2c/ReceiveMerchantIERequestServlet>，当商户投入生产时，应将该连接改为 <https://www.95599.cn/b2c/b2c/ReceiveMerchantIERequestServlet>

## T、商户支付请求验证失败结果接收范例

```
<script runat="server">  
  
    protected void Page_Load(object sender, EventArgs e)  
  
    {  
  
        lblReturnCode.Text = Request["ReturnCode"];  
  
        lblErrorMessage.Text = Request["ErrorMessage"];  
  
    }  
  
</script>  
  
<html xmlns="http://www.w3.org/1999/xhtml">  
  
<head runat="server">  
  
    <title>农行网上支付平台-商户接口范例-支付请求验证失败结果接收</title>  
  
</head>  
  
<body>  
  
    <form id="form1" runat="server">  
  
        <div>  
  
            返回信息<br />  
  
            <asp:Label ID="lblReturnCode" runat="server" Text="Label"></asp:Label>
```

```
        :  
        <asp:Label ID="lblErrorMessage" runat="server" Text="Label"></asp:Label>  
    </div>  
    </form>  
    </body>  
    </html>
```

## U、商户理财验证失败结果接收范例

```
<script runat="server">  
    protected void Page_Load(object sender, EventArgs e)  
    {  
        lblReturnCode.Text = Request["ReturnCode"];  
        lblErrorMessage.Text = Request["ErrorMessage"];  
    }  
</script>  
<html xmlns="http://www.w3.org/1999/xhtml">  
    <head runat="server">  
        <title>农行网上支付平台-商户接口范例-理财验证失败结果接收</title>  
    </head>  
    <body>  
        <form id="form1" runat="server">  
            <div>  
                返回信息<br />  
                <asp:Label ID="lblReturnCode" runat="server" Text="Label"></asp:Label>  
                :  
                <asp:Label ID="lblErrorMessage" runat="server" Text="Label"></asp:Label>  
            </div>  
        </form>  
    </body>
```

&lt;/html&gt;

## V、保险直销支付请求范例

```
protected void btnButton_Click(object sender, EventArgs e)
{
    //1、生成订单对象

    com.hitrust.trustpay.client.b2c.Order tOrder = new
com.hitrust.trustpay.client.b2c.Order();

    tOrder.OrderNo = txtOrderNo.Text;                //设定订单编号 （必要信息）

    tOrder.ExpiredDate = int.Parse(txtExpiredDate.Text); //设定订单有效期

    //wdy add 20100812 tiedaobu

    tOrder.BuyIP = txtBuyIP.Text;                    //IP

    tOrder.OrderDesc = txtOrderDesc.Text;            //设定订单说明

    tOrder.OrderDate = txtOrderDate.Text;            //设定订单日期 （必要信息 -
YYYY/MM/DD）

    tOrder.OrderTime = txtOrderTime.Text;            //设定订单时间 （必要信息 -
HH:MM:SS）

    tOrder.OrderAmount = double.Parse(txtOrderAmount.Text); //设定订单金额 （必要信息）

    tOrder.OrderURL = txtOrderURL.Text;              //设定订单网址

    //2、生成定单订单对象，并将订单明细加入定单中（可选信息）

    com.hitrust.trustpay.client.b2c.OrderItem tOrderItem = new
com.hitrust.trustpay.client.b2c.OrderItem();

    tOrderItem.ProductID = "IP000001";

    tOrderItem.ProductName = "中国移动 IP 卡";

    tOrderItem.UnitPrice = 1.00;

    tOrderItem.Qty = 1;

    tOrder.addOrderItem(tOrderItem);

    tOrderItem = new com.hitrust.trustpay.client.b2c.OrderItem();

    tOrderItem.ProductID = "IP000002";
```

```

tOrderItem.ProductName = "网通 IP 卡";

tOrderItem.UnitPrice = 1.00;

tOrderItem.Qty = 2;

tOrder.addOrderItem(tOrderItem);

//3、生成支付请求对象

com.hitrust.trustpay.client.b2c.PaymentRequest tPaymentRequest = new
com.hitrust.trustpay.client.b2c.PaymentRequest();

tPaymentRequest.Order = tOrder; //设定支付请求的订单 （必要信
息）

tPaymentRequest.ProductType = txtProductType.Text; //设定商品种类 （必要信息）

tPaymentRequest.PaymentType = txtPaymentType.Text; //设定支付类型

tPaymentRequest.NotifyType = txtNotifyType.Text; //设定支付结果通知类型

tPaymentRequest.ResultNotifyURL = txtResultNotifyURL.Text; //设定支付结果回传网址 （必要
信息）

tPaymentRequest.MerchantRemarks = txtMerchantRemarks.Text; //设定商户备注信息

tPaymentRequest.PayLinkType = txtPaymentLinkType.Text; //设定支付接入方式

//PaymentRequest.PAY_LINK_TYPE_NET;1:internet 网络接入

//PaymentRequest.PAY_LINK_TYPE_TEL;2:手机网络接入

//PaymentRequest.PAY_LINK_TYPE_TV;3:数字电视网络接入

//PaymentRequest.PAY_LINK_TYPE_IC;4:智能客户端

//PaymentRequest.PAY_LINK_TYPE_MOBILE;5:电话网络

//

com.hitrust.trustpay.client.b2c.Issue tIssue = new
com.hitrust.trustpay.client.b2c.Issue();

tIssue.Type = int.Parse(ddlIssueType.SelectedValue);

if (tIssue.Type == com.hitrust.trustpay.client.b2c.Issue.ISSUE_TYPE_FINANCING)

tIssue.Furl = txtFurl.Text;

```

```
com.hitrust.trustpay.client.b2c.IssueOrder tIssueOrder = new
com.hitrust.trustpay.client.b2c.IssueOrder();

com.hitrust.trustpay.client.b2c.IssueOrderItem tIssueOrderItem = new
com.hitrust.trustpay.client.b2c.IssueOrderItem();

tIssueOrderItem.Category = txtOrderCategory.Text;

tIssueOrderItem.Code = txtOrderCode.Text;

tIssueOrderItem.Mode = txtOrderMode.Text;

tIssueOrderItem.Name = txtOrderName.Text;

tIssueOrderItem.Amount = double.Parse(txtIssueOrderAmount.Text);

tIssueOrder.addOrderItem(tIssueOrderItem);

tIssueOrderItem = new com.hitrust.trustpay.client.b2c.IssueOrderItem();

tIssueOrderItem.Category = txtOrderCategory.Text + "1";

tIssueOrderItem.Code = txtOrderCode.Text + "1";

tIssueOrderItem.Mode = txtOrderMode.Text + "1";

tIssueOrderItem.Name = txtOrderName.Text + "1";

tIssueOrderItem.Amount = double.Parse("1" + txtIssueOrderAmount.Text);

tIssueOrder.addOrderItem(tIssueOrderItem);

tIssue.Order = tIssueOrder;

if (tIssue.Type == com.hitrust.trustpay.client.b2c.Issue.ISSUE_TYPE_FINANCING ||
tIssue.Type == com.hitrust.trustpay.client.b2c.Issue.ISSUE_TYPE_APPOINTED)
{

    com.hitrust.trustpay.client.b2c.IssueUser tIssueUser = new
com.hitrust.trustpay.client.b2c.IssueUser();

    tIssueUser.CardNo = txtUserCardNo.Text;

    tIssueUser.CertificateNo = txtUserCertificateNo.Text;

    tIssueUser.CertificateType = ddlUserCertificateType.SelectedValue;
```



```
tIssueUser.Name = txtUserName.Text;

tIssue.User = tIssueUser;
}

tPaymentRequest.Issue = tIssue;

//4、传送支付请求并取得支付网址

com.hitrust.trustpay.client.TrxResponse tTrxResponse = tPaymentRequest.postRequest();

string strMessage = "";

if (tTrxResponse.isSuccess())

{

    //5、支付请求提交成功，将客户端导向支付页面

    Response.Redirect(tTrxResponse.getValue("PaymentURL"));

}

else

{

    //6、支付请求提交失败，商户自定后续动作

    strMessage = "ReturnCode    = [" + tTrxResponse.ReturnCode + "<br/>" +

        "ErrorMessage = [" + tTrxResponse.ErrorMessage + "<br/>";

}

lblMessage.Text = strMessage;

}

protected void ddlIssueType_SelectedIndexChanged(object sender, EventArgs e)

{

    if (ddlIssueType.SelectedValue == "1")
```

```
{

    tblFurl.Visible = false;

    tblUser.Visible = false;

}

else if (ddlIssueType.SelectedValue == "2")

{

    tblFurl.Visible = true;

    tblUser.Visible = true;

}

else if (ddlIssueType.SelectedValue == "3")

{

    tblFurl.Visible = false;

    tblUser.Visible = true;

}

}

protected void Page_Load(object sender, EventArgs e)

{

    if (!IsPostBack)

    {

        txtOrderNo.Text = "WDYEC2010" + DateTime.Now.Ticks.ToString();

    }

}
```

## W、页面提交保险支付请求范例

```
string _ResponseString = "";

protected void btnButton_Click(object sender, EventArgs e)

{

    //1、生成订单对象
```

```
com.hitrust.trustpay.client.b2c.Order tOrder = new
com.hitrust.trustpay.client.b2c.Order();

tOrder.OrderNo = txtOrderNo.Text; //设定订单编号 （必要信息）

tOrder.ExpiredDate = int.Parse(txtExpiredDate.Text); //设定订单有效期

//wdy add 20100812 tiedaobu

tOrder.BuyIP = txtBuyIP.Text; //IP

tOrder.OrderDesc = txtOrderDesc.Text; //设定订单说明

tOrder.OrderDate = txtOrderDate.Text; //设定订单日期 （必要信息 -
YYYY/MM/DD)

tOrder.OrderTime = txtOrderTime.Text; //设定订单时间 （必要信息 -
HH:MM:SS)

tOrder.OrderAmount = double.Parse(txtOrderAmount.Text); //设定订单金额 （必要信息）

tOrder.OrderURL = txtOrderURL.Text; //设定订单网址


//2、生成定单订单对象，并将订单明细加入定单中（可选信息）

com.hitrust.trustpay.client.b2c.OrderItem tOrderItem = new
com.hitrust.trustpay.client.b2c.OrderItem();

tOrderItem.ProductID = "IP000001";

tOrderItem.ProductName = "中国移动 IP 卡";

tOrderItem.UnitPrice = 1.00;

tOrderItem.Qty = 1;

tOrder.addOrderItem(tOrderItem);

tOrderItem = new com.hitrust.trustpay.client.b2c.OrderItem();

tOrderItem.ProductID = "IP000002";

tOrderItem.ProductName = "网通 IP 卡";

tOrderItem.UnitPrice = 1.00;

tOrderItem.Qty = 2;

tOrder.addOrderItem(tOrderItem);


//3、生成支付请求对象
```

```

    com.hitrust.trustpay.client.b2c.PaymentRequest tPaymentRequest = new
com.hitrust.trustpay.client.b2c.PaymentRequest();

    tPaymentRequest.Order = tOrder;                                //设定支付请求的订单 （必要信
息）

    tPaymentRequest.ProductType = txtProductType.Text;            //设定商品种类 （必要信息）

    tPaymentRequest.PaymentType = txtPaymentType.Text;            //设定支付类型

    tPaymentRequest.NotifyType = txtNotifyType.Text;              //设定支付结果通知类型

    tPaymentRequest.ResultNotifyURL = txtResultNotifyURL.Text;    //设定支付结果回传网址 （必要
信息）

    tPaymentRequest.MerchantRemarks = txtMerchantRemarks.Text;  //设定商户备注信息

    tPaymentRequest.PayLinkType = txtPaymentLinkType.Text;        //设定支付接入方式

    //PaymentRequest.PAY_LINK_TYPE_NET;1:internet 网络接入

    //PaymentRequest.PAY_LINK_TYPE_TEL;2:手机网络接入

    //PaymentRequest.PAY_LINK_TYPE_TV;3:数字电视网络接入

    //PaymentRequest.PAY_LINK_TYPE_IC;4:智能客户端

    //PaymentRequest.PAY_LINK_TYPE_MOBILE;5:电话网络

    com.hitrust.trustpay.client.b2c.Issue tIssue = new
com.hitrust.trustpay.client.b2c.Issue();

    tIssue.Type = int.Parse(ddlIssueType.SelectedValue);

    if (tIssue.Type == com.hitrust.trustpay.client.b2c.Issue.ISSUE_TYPE_FINANCING)

        tIssue.Furl = txtFurl.Text;

    com.hitrust.trustpay.client.b2c.IssueOrder tIssueOrder = new
com.hitrust.trustpay.client.b2c.IssueOrder();

    com.hitrust.trustpay.client.b2c.IssueOrderItem tIssueOrderItem = new
com.hitrust.trustpay.client.b2c.IssueOrderItem();

    tIssueOrderItem.Category = txtOrderCategory.Text;

    tIssueOrderItem.Code = txtOrderCode.Text;

    tIssueOrderItem.Mode = txtOrderMode.Text;

    tIssueOrderItem.Name = txtOrderName.Text;

    tIssueOrderItem.Amount = double.Parse(txtIssueOrderAmount.Text);
  
```

```
tIssueOrder.addOrderItem(tIssueOrderItem);

tIssueOrderItem = new com.hitrust.trustpay.client.b2c.IssueOrderItem();

tIssueOrderItem.Category = txtOrderCategory.Text + "1";

tIssueOrderItem.Code = txtOrderCode.Text + "1";

tIssueOrderItem.Mode = txtOrderMode.Text + "1";

tIssueOrderItem.Name = txtOrderName.Text + "1";

tIssueOrderItem.Amount = double.Parse("1" + txtIssueOrderAmount.Text);

tIssueOrder.addOrderItem(tIssueOrderItem);

tIssue.Order = tIssueOrder;

if (tIssue.Type == com.hitrust.trustpay.client.b2c.Issue.ISSUE_TYPE_FINANCING ||
tIssue.Type == com.hitrust.trustpay.client.b2c.Issue.ISSUE_TYPE_APPOINTED)

{

    com.hitrust.trustpay.client.b2c.IssueUser tIssueUser = new
com.hitrust.trustpay.client.b2c.IssueUser();

    tIssueUser.CardNo = txtUserCardNo.Text;

    tIssueUser.CertificateNo = txtUserCertificateNo.Text;

    tIssueUser.CertificateType = ddlUserCertificateType.SelectedValue;

    tIssueUser.Name = txtUserName.Text;

    tIssue.User = tIssueUser;

}

tPaymentRequest.Issue = tIssue;

_ResponseString = "";

try
```

```

    {

        //获取接口包中响应报文

        string tSignature =HttpUtility.HtmlEncode( tPaymentRequest.genSignature(1));

        string sTrustPayIETrxURL =
com.hitrust.trustpay.client.MerchantConfig.TrustPayIETrxURL;

        string sErrorUrl = txtFurl.Text;

        form1.Visible = false;

        //重新构造表单页面，把响应后的签名报文放置在表单隐藏域中，重新提交报文到网上支付平台

        _ResponseString = "<form name=\"form2\" method=\"post\" action=\"" +
sTrustPayIETrxURL + "\"> \r\n" +

                                "<input type=\"hidden\" name=\"Signature\" value=\"" +
tSignature + "\"> \r\n" +

                                "<input type=\"hidden\" name=\"errorPage\" value=\"" +
sErrorUrl + "\"> \r\n" +

                                "<input type=\"submit\" value=\"提交\"></form><br/> \r\n" +

                                "<a href='MerchantPaymentIssueIE.aspx'>回商户首页</a> \r\n";

    }

    catch (com.hitrust.trustpay.client.TrxException ex)

    {

        form1.Visible = false;

        _ResponseString = "<center><br/> \r\n" +

            "ReturnCode    = [" + ex.Code + "]"<br/> \r\n" +

            "ErrorMessage = [" + ex.Message + "]"<br/><br/> \r\n" +

            "<a href='MerchantPaymentIssueIE.aspx'>回商户首页</a></center> \r\n";

    }

}

protected void ddlIssueType_SelectedIndexChanged(object sender, EventArgs e)

{

```

```
        if (ddlIssueType.SelectedValue == "1")
        {
            tblFurl.Visible = false;

            tblUser.Visible = false;
        }

        else if (ddlIssueType.SelectedValue == "2")
        {
            tblFurl.Visible = true;

            tblUser.Visible = true;
        }

        else if (ddlIssueType.SelectedValue == "3")
        {
            tblFurl.Visible = false;

            tblUser.Visible = true;
        }
    }

    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            txtOrderNo.Text = "WDYEC2010" + DateTime.Now.Ticks.ToString();
        }
    }
}
```

## X、网上付款信息发送请求范例

```
protected void btnButton_Click(object sender, EventArgs e)
{
    StringBuilder strMessage = new StringBuilder("");
```

//1、取得网上付款所需要的信息

```
string tSerialNumber = Request["SerialNumber"];
string tTotalCount = Request["TotalCount"];
string tTotalAmount = Request["TotalAmount"];
string tCheckType = Request["CheckType"];
string tRemark = Request["Remark"];

//2、取得列表项
string[] NO_arr = null;
string[] CardNo_arr = null;
string[] CardName_arr = null;
string[] RemitAmount_arr = null;
string[] Purpose_arr = null;
int iTotalCount = 0;
double iTotalAmount = 0D;

try
{
    iTotalCount = int.Parse(tTotalCount);
}
catch
{
    strMessage.Append("ReturnCode = [1101]<br/>");
    strMessage.Append("ErrorMessage = [付款总笔数不合法: " + tTotalCount + "]<br/>");
    lblMessage.ForeColor = System.Drawing.Color.Red;
    lblMessage.Text = strMessage.ToString();
    return;
}
try
{
    iTotalAmount = double.Parse(tTotalAmount);
}
catch
{
    strMessage.Append("ReturnCode = [1101]<br/>");
    strMessage.Append("ErrorMessage = [付款总金额不合法" + tTotalAmount + "]<br/>");
    lblMessage.ForeColor = System.Drawing.Color.Red;
    lblMessage.Text = strMessage.ToString();
    return;
}
```



```
}

if (iTotalCount == 1)
{
    string NO = Request["NO"];
    string CardNo = Request["CardNo"];
    string CardName = Request["CardName"];
    string RemitAmount = Request["RemitAmount"];
    string Purpose = Request["Purpose"];

    NO_arr = new string[] { NO };
    CardNo_arr = new string[] { CardNo };
    CardName_arr = new string[] { CardName };
    RemitAmount_arr = new string[] { RemitAmount };
    Purpose_arr = new string[] { Purpose };
}
else
{
    NO_arr = Request["NO"].Split(',');
    CardNo_arr = Request["CardNo"].Split(',');
    CardName_arr = Request["CardName"].Split(',');
    RemitAmount_arr = Request["RemitAmount"].Split(',');
    Purpose_arr = Request["Purpose"].Split(',');
}

ArrayList tRemitList = new ArrayList();
for (int i = 0; i < CardNo_arr.Length; i++)
{
    string[] tremit = new String[5];
    tremit[0] = NO_arr[i];
    tremit[1] = CardNo_arr[i];
    tremit[2] = CardName_arr[i];
    tremit[3] = RemitAmount_arr[i];
    tremit[4] = Purpose_arr[i];
    tRemitList.Add(tremit);
}

//2、生成网上付款请求对象
com.hitrust.trustpay.client.b2c. OnlineRemitRequest tOnlineRemitRequest = new
```

```
com.hitrust.trustpay.client.b2c. OnlineRemitRequest ();

tOnlineRemitRequest.TotalCount = iTotalCount; //总笔数 （必要信息）
tOnlineRemitRequest.TotalAmount = iTotalAmount; //总金额 （必要信息）
tOnlineRemitRequest.SerialNumber = tSerialNumber;
//tOnlineRemitRequest.CheckType = tCheckType;
tOnlineRemitRequest.Remark = tRemark;//备注
tOnlineRemitRequest.RemitDetail = tRemitList;

//3、传送网上付款请求并取得结果
com.hitrust.trustpay.client.TrxResponse tResponse = tOnlineRemitRequest.postRequest();

//4、判断网上付款结果状态，进行后续操作
if (tResponse.isSuccess())
{
    //5、网上付款发送成功
    strMessage.Append("TrxType = [" + tResponse.getValue("TrxType") + "]<br/>");
    strMessage.Append("TotalCount = [" + tResponse.getValue("TotalCount") + "]<br/>");
    strMessage.Append("TotalAmount = [" + tResponse.getValue("TotalAmount") + "]<br/>");
    strMessage.Append("SerialNumber = [" + tResponse.getValue("SerialNumber") + "]<br/>");
    strMessage.Append("ResultMessage = [" + tResponse.ErrorMessage + "]<br/>");
    lblMessage.ForeColor = System.Drawing.Color.Black;
}
else
{
    //6、网上付款发送失败
    strMessage.Append("ReturnCode = [" + tResponse.ReturnCode + "]<br/>");
    strMessage.Append("ErrorMessage = [" + tResponse.ErrorMessage + "]<br/>");
    lblMessage.ForeColor = System.Drawing.Color.Red;
}

lblMessage.Text = strMessage.ToString();
}

</script>
```

## Y、网上付款交易结果查询请求范例

```
protected void btnButton_Click(object sender, EventArgs e)
```

```
{

    StringBuilder strMessage = new StringBuilder("");

    string tSerialNumber = SerialNumber.Text;
    string tReceiveAccount = ReceiveAccount.Text;

    //1、生成网上付款交易结果查询请求对象

    com.hitrust.trustpay.client.b2c. OnlineRmtQueryResultRequest tRequest = new
com.hitrust.trustpay.client.b2c. OnlineRmtQueryResultRequest();

    tRequest.SerialNumber = tSerialNumber;
    //tRequest.PayAccount = PayAccount.Text;
    tRequest.ReceiveAccount = tReceiveAccount;
    tRequest.StartTime = StartTime.Text;
    tRequest.EndTime = EndTime.Text;
    //tRequest.Remark = Remark.Text;

    //2、传送网上付款交易结果查询请求并取得结果

    com.hitrust.trustpay.client.TrxResponse tTrxResponse = tRequest.postRequest();
    if (tTrxResponse.isSuccess())
    {
        //生成批量对象

        com.hitrust.trustpay.client.b2c. OnlineRmtBatch tBatch = new
com.hitrust.trustpay.client.b2c. OnlineRmtBatch(new
com.hitrust.trustpay.client.XMLDocument(tTrxResponse.getValue("ResultSets")));

        //3、输出查询结果

        if (!string.IsNullOrEmpty(tSerialNumber) && string.IsNullOrEmpty(tReceiveAccount))
        {
            strMessage.Append("SerialNumber    = [" + tTrxResponse.getValue("SerialNumber") + "]<br/>");
            strMessage.Append("TrnxTime      = [" + tTrxResponse.getValue("TrnxTime") + "]<br/>");
            strMessage.Append("TotalCount   = [" + tTrxResponse.getValue("TotalCount") + "]<br/>");
            strMessage.Append("TotalAmount  = [" + tTrxResponse.getValue("TotalAmount") + "]<br/>");
            strMessage.Append("Status       = [" + tTrxResponse.getValue("Status") + "]<br/>");
            strMessage.Append("SuccessAmount = [" + tTrxResponse.getValue("SuccessAmount") + "]<br/>");
            strMessage.Append("SuccessCount  = [" + tTrxResponse.getValue("SuccessCount") + "]<br/>");
            strMessage.Append("FailAmount   = [" + tTrxResponse.getValue("FailAmount") + "]<br/>");
            strMessage.Append("FailCount    = [" + tTrxResponse.getValue("FailCount") + "]<br/>");

            //4、取得查询结果明细

            ArrayList tQueryResults = tBatch.QueryResults;
```

```

        for (int i = 0; i < tQueryResults.Count; i++)
        {
            com.hitrust.trustpay.client.b2c.QueryResult tQueryResult =
            (com.hitrust.trustpay.client.b2c.QueryResult)tQueryResults[i];

            strMessage.Append("No" = [" + tQueryResult.No + "]<br/>");
            strMessage.Append("PayAccountNo" = [" + tQueryResult.PayAccountNo + "]<br/>");
            strMessage.Append("PayAccountName" = [" + tQueryResult.PayAccountName + "]<br/>");
            strMessage.Append("ReceiveAccountNo" = [" + tQueryResult.ReceiveAccountNo + "]<br/>");
            strMessage.Append("ReceiveAccountName" = [" + tQueryResult.ReceiveAccountName + "]<br/>");
            strMessage.Append("Purpose" = [" + tQueryResult.Purpose + "]<br/>");
            strMessage.Append("PayAmount" = [" + tQueryResult.PayAmount + "]<br/>");
            strMessage.Append("Status" = [" + tQueryResult.Status + "]<br/>");
            strMessage.Append("FailReason" = [" + tQueryResult.FailReason + "]<br/>");
        }
    }

    else if (string.IsNullOrEmpty(tSerialNumber) && !string.IsNullOrEmpty(tReceiveAccount))
    {
        //5、取得查询结果明细
        ArrayList tQueryResults = tBatch.QueryResults;

        for (int i = 0; i < tQueryResults.Count; i++)
        {
            com.hitrust.trustpay.client.b2c.QueryResult tQueryResult =
            (com.hitrust.trustpay.client.b2c.QueryResult)tQueryResults[i];

            strMessage.Append("No" = [" + tQueryResult.No + "]<br/>");
            strMessage.Append("SerialNumber" = [" + tQueryResult.SerialNumber + "]<br/>");
            strMessage.Append("TrnxTime" = [" + tQueryResult.TrnxTime + "]<br/>");
            strMessage.Append("PayAccountNo" = [" + tQueryResult.PayAccountNo + "]<br/>");
            strMessage.Append("PayAccountName" = [" + tQueryResult.PayAccountName + "]<br/>");
            strMessage.Append("ReceiveAccountNo" = [" + tQueryResult.ReceiveAccountNo + "]<br/>");
            strMessage.Append("ReceiveAccountName" = [" + tQueryResult.ReceiveAccountName + "]<br/>");
            strMessage.Append("Purpose" = [" + tQueryResult.Purpose + "]<br/>");
            strMessage.Append("PayAmount" = [" + tQueryResult.PayAmount.ToString("0.00") +
            "]<br/>");

            strMessage.Append("Status" = [" + tQueryResult.Status + "]<br/>");
            strMessage.Append("FailReason" = [" + tQueryResult.FailReason + "]<br/>");
        }
    }

    else if (!string.IsNullOrEmpty(tSerialNumber) && !string.IsNullOrEmpty(tReceiveAccount))
    {

```

## //5、取得查询结果明细

```
ArrayList tQueryResults = tBatch.QueryResults;

for (int i = 0; i < tQueryResults.Count; i++)
{
    com.hitrust.trustpay.client.b2c.QueryResult tQueryResult =
    (com.hitrust.trustpay.client.b2c.QueryResult)tQueryResults[i];

    strMessage.Append("No" = [" + tQueryResult.No + "]<br/>");
    strMessage.Append("SerialNumber" = [" + tQueryResult.SerialNumber + "]<br/>");
    strMessage.Append("TrnxTime" = [" + tQueryResult.TrnxTime + "]<br/>");
    strMessage.Append("PayAccountNo" = [" + tQueryResult.PayAccountNo + "]<br/>");
    strMessage.Append("PayAccountName" = [" + tQueryResult.PayAccountName + "]<br/>");
    strMessage.Append("ReceiveAccountNo" = [" + tQueryResult.ReceiveAccountNo + "]<br/>");
    strMessage.Append("ReceiveAccountName" = [" + tQueryResult.ReceiveAccountName + "]<br/>");
    strMessage.Append("Purpose" = [" + tQueryResult.Purpose + "]<br/>");
    strMessage.Append("PayAmount" = [" + tQueryResult.PayAmount + "]<br/>");
    strMessage.Append("Status" = [" + tQueryResult.Status + "]<br/>");
    strMessage.Append("FailReason" = [" + tQueryResult.FailReason + "]<br/>");
}
}

else
{

    strMessage.Append("ReturnCode" = [" + tTrxResponse.ReturnCode + "]<br/>");
    strMessage.Append("ErrorMessage" = [" + tTrxResponse.ErrorMessage + "]<br/>");
}

lblMessage.ForeColor = System.Drawing.Color.Red;
lblMessage.Text = strMessage.ToString();
}
```

## Z、网上付款银行卡状态验证请求范例

```
protected void btnButton_Click(object sender, EventArgs e)
{
    //1、取得身份验证请求所需要的信息
    //2、生成身份验证请求对象

    com.hitrust.trustpay.client.b2c.OnlineRmtCardVerifyRequest tRequest = new
    com.hitrust.trustpay.client.b2c.OnlineRmtCardVerifyRequest();
}
```

```
tRequest.BankCardNo = txtBankCardNo.Text;           //银行帐号      (必要信息)
tRequest.AccountName = txtAccountName.Text;         //持卡人姓名    (必要信息)

//3、传送身份验证请求并取得支付网址
com.hitrust.trustpay.client.TrxResponse tTrxResponse = tRequest.postRequest();
StringBuilder strMessage = new StringBuilder("");
if (tTrxResponse.isSuccess())
{
    //4、身份验证请求提交成功，返回结果。
    string tCardVerifyResult = tTrxResponse.getValue("Status").ToString();
    strMessage.Append(tCardVerifyResult);
}
else
{
    //5、身份验证请求提交失败，商户自定后续动作
    strMessage.Append("ReturnCode = [" + tTrxResponse.ReturnCode + "<br/>");
    strMessage.Append("ErrorMessage = [" + tTrxResponse.ErrorMessage + "<br/>");
    lblMessage.ForeColor = System.Drawing.Color.Red;
}

lblMessage.Text = strMessage.ToString();

}
```

## AA、委托扣款签约结果查询请求范例

```
protected void btnButton_Click(object sender, EventArgs e)
{
    //1、取得商户委托扣款签约查询所需要的信息
    String tAgentSignNo = AgentSignNo.Text;

    //2、生成商户委托扣款签约查询请求对象
    com.hitrust.trustpay.client.b2c.QueryAgentSignRequest tRequest = new
com.hitrust.trustpay.client.b2c.QueryAgentSignRequest();

    tRequest.AgentSignNo = tAgentSignNo; //签约号      (必要信息)

    //3、传送商户委托扣款签约查询请求并取得订单查询结果
```

```
com.hitrust.trustpay.client.TrxResponse tResponse = tRequest.postRequest();

StringBuilder strMessage = new StringBuilder();

//4、判断商户委托扣款签约查询结果状态，进行后续操作
if (tResponse.isSuccess())
{
    //5、生成委托扣款签约对象

    com.hitrust.trustpay.client.b2c.AgentSign tAgentSign = new
com.hitrust.trustpay.client.b2c.AgentSign(new
com.hitrust.trustpay.client.XMLDocument(tResponse.getValue("TrxResponse")));

    strMessage.Append("AgentSignNo      = [" + tAgentSign.AgentSignNo + "<br>");
    strMessage.Append("MerchantNo      = [" + tAgentSign.MerchantNo + "<br>");
    strMessage.Append("CardNo        = [" + tAgentSign.CardNo + "<br>");
    strMessage.Append("AccountType    = [" + tAgentSign.AccountType + "<br>");
    strMessage.Append("CertificateNo   = [" + tAgentSign.CertificateNo + "<br>");
    strMessage.Append("CertificateType  = [" + tAgentSign.CertificateType + "<br>");
    strMessage.Append("SignStatus= [" + tAgentSign.SignStatus + "<br>");
    strMessage.Append("SignDate      = [" + tAgentSign.SignDate + "<br>");
    strMessage.Append("UnSignDate    = [" + tAgentSign.UnSignDate + "<br>");

}
else
{
    //7、商户委托扣款签约查询失败

    strMessage.Append("ReturnCode  = [" + tResponse.ReturnCode + "<br>");
    strMessage.Append("ErrorMessage = [" + tResponse.ErrorMessage + "<br>");

}

lblMessage.Text = strMessage.ToString();
}
```


## 7. 附录二、响应码一览表

响应码	响应类型	说明
0000	网上支付平台返回	交易成功
1000	本地返回	无法读取商户端配置文件
1001	本地返回	商户端配置文件中参数设置错误
1002	本地返回	无法读取商户证书文档
1003	本地返回	无法读取商户私钥
1004	本地返回	无法写入交易日志文档
1005	本地返回	证书过期
1006	本地返回	证书格式错误
1100	本地返回	商户提交的交易资料不完整
1101	本地返回	商户提交的交易资料不合法
1102	本地返回	签名交易报文时发生错误
1103	本地返回	无法连线签名服务器
1104	本地返回	签名服务器返回签名错误
1201	本地返回	无法连线网上支付平台
1202	本地返回	提交交易时发生网络错误
1203	本地返回	无法接收到网上支付平台的响应
1204	本地返回	接收网上支付平台响应报文时发生网络错误
1205	本地返回	无法辨识网上支付平台的响应报文
1206	本地返回	网上支付平台服务暂时停止
1301	本地返回	网上支付平台的响应报文不完整
1302	本地返回	网上支付平台的响应报文签名验证失败
1303	本地返回	无法辨识网上支付平台的交易结果
1999	本地返回	系统发生无法预期的错误



2000	网上支付平台返回	无法读取网上支付平台系统配置文件
2001	网上支付平台返回	网上支付平台系统配置文件中参数设置错误
2002	网上支付平台返回	无法读取网上支付平台证书
2003	网上支付平台返回	无法读取网上支付平台私钥
2004	网上支付平台返回	数据库错误
2006	网上支付平台返回	证书格式错误
2100	网上支付平台返回	商户提交的交易资料不完整
2101	网上支付平台返回	商户提交的交易资料不合法
2102	网上支付平台返回	签名响应报文时发生错误
2201	网上支付平台返回	无法连线银行后台系统
2202	网上支付平台返回	接收商户交易请求时发生网络错误
2205	网上支付平台返回	无法辨识商户提交的交易请求报文
2301	网上支付平台返回	商户提交的交易请求报文不完整
2302	网上支付平台返回	商户提交的交易请求报文签名验证失败
2303	网上支付平台返回	商户提交的商户号与签名所用的证书不匹配
2304	网上支付平台返回	商户状态不允许交易
2305	网上支付平台返回	商户不存在
2306	网上支付平台返回	订单状态不允许进行此种交易
2307	网上支付平台返回	无此订单
2308	网上支付平台返回	商户无可用的支付方式
2309	网上支付平台返回	无法取得商户证书
2310	网上支付平台返回	网上支付平台未开放此种类的交易
2311	网上支付平台返回	商户未开放指定的商品种类
2400	网上支付平台返回	后台系统响应交易失败
2500	网上支付平台返回	所有交易已测试通过，请通知银行开放可以进行正式交易

2501	网上支付平台返回	测试交易种类错误，请按照网上支付平台所指示的顺序进行测试
2600	网上支付平台返回	未到可以下载对账单的时间，请在可以下载对账单的时间再下载
2999	网上支付平台返回	系统发生无法预期的错误

 中国农业银行 AGRICULTURAL BANK OF CHINA	农行网上支付平台
	商户接口编程指南– ASP.NET2.0 Edition – V1.0.7

## 8. 附录三、TrustPay Client API

`com.hitrust.trustpay.client.TrxResponse`

### 说明

商户端接口软件包实体类，代表网上支付平台对商户提交交易的响应。

### 属性

属性	属性名称	类型	说明
<code>ReturnCode</code>	响应码	字符串	响应码请参考《附录二、响应码一览表》的说明
<code>ErrorMessage</code>	错误信息	字符串	交易错误原因的本文说明

### 方法

```
public bool isSuccess()
```

回传交易是否成功。**true** 表示交易成功； **false** 表示交易失败。

```
public String getValue(System.String aTag)
```

回传指定参数名的参数值。

## com.hitrust.trustpay.client.b2c.Order

### 说明

商户端接口软件包实体类，代表消费者在商户网站购买的订单。

### 属性

属性	属性名称	类型	说明
OrderNo	订单编号	字符串 最大长度 50 个字符	支付请求时必须设定
ExpiredDate	订单有效期	正整数 大于等于 30 天	支付请求时必须设定
OrderDesc	订单说明	字符串 最大长度 100 个字符	支付请求时可设定
OrderDate	订单日期	字符串 YYYY/MM/DD	支付请求时必须设定
OrderTime	订单时间	字符串 HH:MM:SS	支付请求时必须设定
OrderAmount	订单金额	浮点数 单位为人民币元 小数点后最多两位	支付请求时必须设定
OrderItems	订单明细	为 OrderItem 对象的 ArrayList 集合	支付请求时可设定
OrderURL	订单查询网址	字符串 最大长度 200 个字符	支付请求时必须设定
PayAmount	支付金额	浮点数 单位为人民币元	不需设定 网上支付平台响应

		小数点后最多两位	
RefundAmount	退货金额	浮点数 单位为人民币元 小数点后最多两位	不需设定 网上支付平台响应
OrderStatus	订单状态	字符串代码	不需设定 网上支付平台响应  00: 订单取消。 01: 订单建立，等待支付。 02: 消费者已支付，等待支付结果。 03: 订单已支付（支付成功） 04: 订单已结算（支付成功） 05: 订单已退款。 99: 订单支付失败。

## 方法

```
public Order initByString(String aOrderString)
```

初始订单对象。


参数：aOrderString – 网上支付平台响应的订单查询结果

```
public Order addOrderItem(OrderItem aOrderItem)
```

新增一笔订单明细。

参数：aOrderItem – 订单明细对象

```
public Order clearOrderItems()
```

 <b>中国农业银行</b> <small>AGRICULTURAL BANK OF CHINA</small>	农行网上支付平台
	商户接口编程指南– ASP.NET2.0 Edition – V1.0.7

清除订单对象中的订单明细。

## com.hitrust.trustpay.client.b2c.OrderItem

### 说明

商户端接口软件包实体类，代表消费者在商户网站购买的商品明细。

### 属性

属性	属性名称	类型	说明
ProductID	产品代码	字符串 最大长度 20 个字符	支付请求时必须设定
ProductName	产品名称	字符串 最大长度 50 个字符	支付请求时必须设定
UnitPrice	产品单价	浮点数 单位为人民币元 小数点后最多两位	支付请求时必须设定
Qty	购买数量	整数	支付请求时必须设定
Amount	金额小计	浮点数 单位为人民币元 小数点后最多两位	不能设定，只能读取。 等于 $\text{UnitPrice} * \text{Qty}$

## com.hitrust.trustpay.client.b2c.PaymentRequest

### 说明

商户端接口软件包业务处理类，负责商户提交支付请求的处理。

### 属性

属性	属性名称	类型	说明
Order	订单	Order 对象	必须设定
ProductType	商品种类	字符串代码	必须设定 1: 非实体商品，如服务、IP 卡、下载 MP3、... 2: 实体商品
PaymentType	支付类型	字符串代码	若不指定，系统默认是农行卡支付。 1: 农行卡支付 网上支付平台将会回传农行卡支付页面。 2: 国际卡支付 网上支付平台将会回传国际卡支付页面。 3: 贷记卡支付 网上支付平台将会回传贷记卡支付页面。 A: 农行卡和贷记卡支付合并 网上支付平台将会回传农行卡和贷记卡支付合并支付页面。



			<b>5: 第三方的跨行支付</b> 网上支付平台将会回传第三方的跨行支付页面。
<b>NotifyType</b>	支付结果通知类型	字符串代码	必须设定 <b>0: 页面通知</b> 网上支付平台将会回传支付结果网址。 <b>1: 服务器通知</b> 网上支付平台将会把支付结果消息直接通知到商户指定的地址。
<b>ResultNotifyURL</b>	支付结果回传网址	字符串 最大长度 200 个字符	必须设定 消费者支付成功后，网上支付平台会将支付结果信息发送到此网址。
<b>MerchantRemarks</b>	商户备注信息	字符串 最大长度 500 个字符	网上支付平台的支付结果信息将包含此备注信息。 商户可以用来传递所需的信息。

## 方法

`public TrxResponse postRequest()` 单一商户配置

`public TrxResponse extendPostRequest(int aMerchantNo)` 多商户配置

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的

`getValue(System.String aTag)`方法取得下列信息。



参数	参数名称	说明
TrxType	交易种类	PayReq: 支付请求
PaymentURL	支付网址	网上支付平台的支付网址，商户必须将消费者的浏览器页面导至这个支付网址，消费者将在此页面进行支付。

## com.hitrust.trustpay.client.b2c.PaymentResult

### 说明

商户端接口软件包业务处理类，继承 `com.hitrust.trustpay.client.TrxResponse`，表示网上支付平台响应的支付结果。

在取得对象的属性前，必须先以 `init()` 方法初始对象。

### 属性

### 方法

```
public PaymentResult init(System.String aMessage)
```

初始支付结果对象的属性。

参数：**aMessage** – 网上支付平台发送到商户支付结果接收页面的 **MSG** 参数。商户支付结果接收页面可以使用 `Request("MSG")` 来取得。

返回值：支付结果对象。如果交易响应成功，则可以使用支付结果对象的 `getValue(System.String aTag)` 方法取得下列信息。

参数	参数名称	说明
TrxType	交易种类	PayResult: 支付结果
OrderNo	订单号	
Amount	订单金额	
BatchNo	交易批次号	
VoucherNo	交易凭证号	用于交易对账时使用

HostDate	银行交易日期	YYYY/MM/DD
HostTime	银行交易时间	HH:MM:SS
MerchantRemarks	商户备注信息	商户在支付请求时所提交的信息
PayType	消费者支付方式	<b>PAY01: 银行网点注册客户支付</b> <b>PAY02: 网上注册客户支付</b> <b>PAY03: 威士(VISA)国际卡支付</b> <b>PAY04: 万事达(MasterCard)国际卡支付</b>
NotifyType	支付结果通知方式	0: 页面通知;    1: 服务器通知

## com.hitrust.trustpay.client.b2c.QueryOrderRequest

### 说明

商户端接口软件包业务处理类，负责商户提交订单状态查询的处理。

### 属性

属性	属性名称	类型	说明
OrderNo	订单编号	字符串 最大长度 50 个字符	必须设定
EnableDetailQuery	是否查询 详细信息	布尔型	必须设定 <b>true:</b> 查询详细信息 <b>false:</b> 不查询详细信息

### 方法


public TrxResponse postRequest() 单一商户配置

public TrxResponse extendPostRequest(int aMerchantNo) 多商户配置

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的  
getValue(System.String aTag)方法取得下列信息。

参数	参数名称	说明
TrxType	交易种类	<b>Query:</b> 订单状态查询
Order	订单信息	以字符串表示的订单信息。

 <b>中国农业银行</b> <small>AGRICULTURAL BANK OF CHINA</small>	农行网上支付平台
	商户接口编程指南– ASP.NET2.0 Edition – V1.0.7

		可用来初始订单对象。
--	--	------------

## com.hitrust.trustpay.client.b2c.RefundRequest

### 说明

商户端接口软件包业务处理类，负责商户提交退货交易的处理。

### 属性

属性	属性名称	类型	说明
OrderNo	订单编号	字符串 最大长度 50 个字符	必须设定
TrxAmount	退货金额	浮点数 单位为人民币元 小数点后最多两位	必须设定

### 方法

public TrxResponse postRequest() 单一商户配置

public TrxResponse extendPostRequest(int aMerchantNo) 多商户配置

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的  
 getValue(System.String aTag)方法取得下列信息。。

参数	参数名称	说明
TrxType	交易种类	Refund: 退货请求
OrderNo	订单号	



TrxAmount	退货金额	
BatchNo	交易批次号	
VoucherNo	交易凭证号	用于交易对账时使用
HostDate	银行交易日期	YYYY/MM/DD
HostTime	银行交易时间	HH:MM:SS



## com.hitrust.trustpay.client.b2c.SettleRequest

### 说明

商户端接口软件包业务处理类，负责商户对账单下载的处理。

### 属性

属性	属性名称	类型	说明
SettleDate	对账日期	字符串 YYYY/MM/DD	必须设定
SettleType	对账类型	字符串代码	必须设定 TRX: 交易对账单下载

### 方法

public TrxResponse postRequest() 单一商户配置

public TrxResponse extendPostRequest(int aMerchantNo) 多商户配置

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象来初始对账单对象（com.hitrust.trustpay.client.SettleFile）。

## com.hitrust.trustpay.client.b2c.SettleFile

### 说明

商户端接口软件包实体类，代表商户下载的对账单。

在取得对象的属性前，必须先以 `init()` 方法初始对象

### 属性

属性	属性名称	类型	说明
SettleDate	对账日期	字符串 YYYY/MM/DD	
SettleType	对账类型	字符串代码	TRX: 交易对账单
NumOfPayments	支付交易 总笔数	整数	
SumOfPayAmount	支付交易 总金额	浮点数 单位为人民币元 小数点后最多两位	
NumOfRefunds	退货交易 总笔数	整数	
SumOfRefundAmount	退货交易 总金额	浮点数 单位为人民币元 小数点后最多两位	
DetailRecords	交易明细	字符串数组	

每笔交易明细为字符串类型，使用逗号分隔不同的字段。字段信息如下：

交易类型      P: 支付交易      R: 退货交易

订单号

交易金额

凭证号

交易时间          YYYY/MM/DD HH:MM:SS

## 方法

`public SettleFile save(String aFileName)` 保存对账单至指定的文件

`public SettleFile load(String aFileName)` 由文件中读取对账单

## `com.hitrust.trustpay.client.b2c.IdentityVerifyRequest`

### 说明

商户端接口软件包业务处理类，负责商户提交客户身份验证交易的处理。

### 属性

属性	属性名称	类型	说明
<code>ResultNotifyURL</code>	身份验证 结果回传 网址	String  最小长度 20 个字符 最大长度 200 个字符	必须设定
<code>BankCardNo</code>	银行卡号	String  最大长度 20 个字符	必须设定
<code>CertificateType</code>	证件类型	String  如果是身份证号码， 则必须是 15 位或者 18 位	必须设定
<code>CertificateNo</code>	证件号码	String  最大长度 30 个字符	必须设定

RequestDate	请求日期	String YYYY/MM/DD 的日期格式	必须设定
RequestTime	请求时间	String HH:MM:SS 的时间格式	必须设定

## 方法

```
public TrxResponse postRequest()
```

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的  
 getValue(String aTag)方法取得下列信息。。

参数	参数名称	说明
VerifyURL	身份验证请求网址	客户到此网址输入支付密码进行身份验证

`com.hitrust.trustpay.client.b2c.Batch`

### 说明

商户端接口软件包实体类，代表退款批量对象。

### 属性

属性	属性名称	类型	说明
SerialNumber	退款批量流水号	String	退款批量查询请求时必须设定
RefundAmount	退款总金额	double	不需设定
RefundCount	退款总笔数	int	不需设定
Status	退款批量状态	String	不需设定 网上支付平台响应 0: 批量待复核。 1: 复核通过待发送。 2: 复核驳回。 3: 复核通过后已发送/等待处理。 4: 退款处理成功，指整个批量处理完成，与批量内订单退款处理状态无关。 5: 退款处理失败，指整个批量处理发生异常，与批量内订单退款处理状态无关。

Order	订单对象	Order	不需设定
-------	------	-------	------

## 方法

```
public Batch()
```

**Batch** 的构造函数，生成一个空的退款批量对象。

```
public Batch(XMLDocument aXMLDocument)
```

构造 **Batch** 对象，并使用 XML 文件初始对象的属性。

参数: **aXMLDocument** – 网上支付平台响应的订单查询结果

```
public Batch addOrder(Order aOrder)
```

新增一笔订单。

参数: **aOrder** – 订单对象

**com.hitrust.trustpay.client.b2c.OverdueBatch**

## 说明

商户端接口软件包实体类，代表批量退款对象。

## 属性

属性	属性名称	类型	说明
SerialNumber	批量退款流水号	String	批量退款查询请求时必须设定

RefundAmount	批量退款总金额	double	不需设定
RefundCount	批量退款总笔数	int	不需设定
Status	退款批量状态	String	不需设定 网上支付平台响应 0: 批量待复核。 1: 复核通过待发送。 2: 复核驳回。 3: 复核通过后已发送/等待处理。 4: 退款处理成功, 指整个批量处理完成, 与批量内订单退款处理状态无关。 5: 退款处理失败, 指批量退款整个批量处理发生异常, 与批量内订单退款处理状态无关。
Order	订单对象	Order	不需设定

## 方法

```
public OverdueBatch()
```

**OverdueBatch** 的构造函数, 生成一个空的退款批量对象。

```
public OverdueBatch(XMLDocument aXMLDocument)
```

构造 **OverdueBatch** 对象, 并使用 XML 文件初始对象的属性。

参数: **aXMLDocument** – 网上支付平台响应的订单查询结果



```
public Batch addOrder(Order aOrder)
```

新增一笔订单。

参数：aOrder – 订单对象



## com.hitrust.trustpay.client.b2c.BatchSendRequest

### 说明

商户端接口软件包业务处理类，负责商户提交批量退款发送交易的处理。

### 属性

属性	属性名称	类型	说明
SerialNumber	退款批量流水号	String 最大长度 30 个字符	必须设定
OperatorNo	操作员号	String 最大长度 4 个字符	必须设定

### 方法

```
public TrxResponse postRequest()
```

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的

getValue(String aTag)方法取得下列信息。。

参数	参数名称	说明
TrxType	交易种类	RefundBatchSendReq: 退款批量发送请求
SerialNumber	退款批量流水号	
SendTime	发送时间	

## com.hitrust.trustpay.client.b2c.QueryBatchRequest

### 说明

商户端接口软件包业务处理类，负责商户提交批量退款结果查询交易的处理。

### 属性

属性	属性名称	类型	说明
SerialNumber	退款批量流水号	String 最大长度 30 个字符	必须设定

### 方法

```
public TrxResponse postRequest()
```

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的  
getValue(String aTag)方法取得下列信息。。

参数	参数名称	说明
RefundBatch	批量退款对象	

## com.hitrust.trustpay.client.b2c.OverdueRefundRequest

### 说明

商户端接口软件包业务处理类，负责商户提交批量退款请求交易的处理。

## 属性

属性	属性名称	类型	说明
TotalAmount	批量退款订单总金额	double	必须设定
TotalCount	批量退款订单总笔数	int	必须设定
Remark	批量退款备注信息	String	最长 30 个字符
Orderlist	批量退款订单列表	ArrayList	必须设定（信息必须与总金额和总笔数一致）

## 方法

```
public TrxResponse postRequest()
```

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的

getValue(String aTag)方法取得下列信息。

参数	参数名称	说明
TrxType	交易种类	OverdueRefund: 批量退款发送请求
TotalCount	批量总笔数	
TotalAmount	批量总金额	
SerialNumber	批量退款批量流水号	
HostDate	服务器返回日期	
HostTime	服务器返回时间	
ResultMessage	服务器返回信息	

## com.hitrust.trustpay.client.b2c.QueryOverdueRefundRequest

### 说明

商户端接口软件包业务处理类，负责商户提交批量退款结果查询交易的处理。

### 属性

属性	属性名称	类型	说明
SerialNumber	批量退款批量流水号	String 最大长度 30 个字符	必须设定

### 方法

```
public TrxResponse postRequest()
```

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的  
 getValue(String aTag)方法取得下列信息：

参数	参数名称	说明
SerialNumber	批量退款流水号	
RefundCount	批量总笔数	
RefundAmount	批量总金额	
BatchStatus	批量状态	0：批量待复核。 1：复核通过待发送。 2：复核驳回。

		<b>3：</b> 复核通过后已发送/等待处理。  <b>4：</b> 退款处理成功，指整个批量处理完成，与批量内订单退款处理状态无关。  <b>5：</b> 退款处理失败，指批量退款整个批量处理发生异常，与批量内订单退款处理状态无关。
OrderNo	订单号	为详细订单信息，个数与退款总笔数一致。订单状态：  <b>0：</b> 未处理； <b>1：</b> 处理成功； <b>2：</b> 处理失败
RefundAmount	退款金额	
OrderStatus	订单状态	
OrderDesc	失败原因	

`com.hitrust.trustpay.client.b2c.B2CAgentSignResult`

### 说明

商户端接口软件包实体类，代表委托扣款签约/委托扣款解约交易结果。

### 属性

无

### 方法

```
public B2CAgentSignResult(String aMessage)
```

以接收到的“MSG”作为参数，构建委托扣款签约/委托扣款解约处理结果对象

回传值：签约解约处理结果对象的 `getValue(String aTag)` 方法取得下列信息：

参数	参数名称	说明
TrxType	交易类型	“AgentSign” 或 “AgentUSign”
OrderNo	请求编号	签约结果和解约结果对象都有此信息
AgentSignNo	签约协议号	只有签约结果对象有此信息
Last4CardNo	签约卡号后四位	只有签约结果对象有此信息

## `com.hitrust.trustpay.client.b2c.B2CAgentSignContractRequest`

### 说明

商户端接口软件包业务处理类，负责商户提交委托扣款签约交易的处理。

### 属性

属性	属性名称	类型	说明
OrderNo	订单编号	String 最大长度 50 个字符	必须设定
RequestDate	请求日期	String YYYY/MM/DD 的日期格式	必须设定
RequestTime	请求时间	String HH:MM:SS 的时间格式	必须设定
InvaIdDate	过期日期	String YYYY/MM/DD 的日期格式	必须设定

CertificateType	证件类型	String 当前系统必须设置为公民身份证类型，值为'I'	必须设定
CertificateNo	证件号码	当前系统必须设置为公民身份证	必须设定
NotifyType	支付结果通知类型	字符串代码	必须设定  1: 页面通知  网上支付平台将会回传支付结果网址。  2: 服务器通知  网上支付平台将会把支付结果消息直接通知到商户指定的地址。
ResultNotifyURL	支付结果回传网址	String 最大长度 200 个字符	必须设定  消费者支付成功后，网上支付平台会将支付结果信息发送到此网址。
CardType	农行卡类型	给定值：1:农行借记卡 准贷记卡 2:农行贷记卡 A:农行卡合并	必须设定

## 方法

```
public TrxResponse postRequest()
```

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的

getValue(String aTag)方法取得下列信息：

参数	参数名称	说明
B2CAgentSignContractURL	委托扣款签约网址	

`com.hitrust.trustpay.client.b2c.B2CAgentUnsignContractRequest`

说明

商户端接口软件包业务处理类，负责商户提交委托扣款解约交易的处理。

属性

属性	属性名称	类型	说明
OrderNo	订单编号	String 最大长度 50 个字符	必须设定
RequestDate	请求日期	String YYYY/MM/DD 的日期格式	必须设定
RequestTime	请求时间	String HH:MM:SS 的时间格式	必须设定
CertificateType	证件类型	String 当前系统必须设置为公民身份证类型，值为'I'	必须设定
CertificateNo	证件号码	当前系统必须设置为公民身份证	必须设定
NotifyType	支付结果通知类	字符串代码	必须设定



	型		<b>1: 页面通知</b>  网上支付平台将会回传支付结果网址。  <b>2: 服务器通知</b>  网上支付平台将会把支付结果消息直接通知到商户指定的地址。
ResultNotifyURL	支付结果回传网址	String 最大长度 200 个字符	必须设定  消费者支付成功后，网上支付平台会将支付结果信息发送到此网址。
AgentSignNo	签约协议号	String	必须设定

## 方法

```
public TrxResponse postRequest()
```

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的

getValue(String aTag)方法取得下列信息：

参数	参数名称	说明
B2CAgentSignContractURL	委托扣款解约网址	

## com.hitrust.trustpay.client.b2c.B2CAgentPaymentRequest

### 说明

商户端接口软件包业务处理类，负责商户提交委托扣款单笔代扣交易的处理。

### 属性

属性	属性名称	类型	说明
OrderNo	订单编号	String 最大长度 50 个字符	必须设定
ExpiredDate	订单有效期	Int 大于等于 30 天	必须设定
RequestDate	请求日期	String YYYY/MM/DD 的日期格式	必须设定
RequestTime	请求时间	String HH:MM:SS 的时间格式	必须设定
CertificateNo	证件号码	当前系统必须设置为公民身份证	必须设定
AgentSignNo	签约协议号	String	必须设定
Currency	账单币种	String	必须设定, 设置为: “RMB”
Amount	账单金额	String 小数点后最多两位	必须设定
ProductId	商品编号	String	必须设定
ProductName	商品名称	String	必须设定

Quantity	商品数量	String	必须设定
----------	------	--------	------

## 方法

```
public TrxResponse postRequest()
```

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：根据返回码判断交易是否成功。

`com.hitrust.trustpay.client.b2c.AgentBatch`

## 说明

商户端接口软件包实体类，代表委托扣款批量对象。

## 属性

属性	属性名称	类型	说明
MerchantNo	商户编号	String	不需设定 网上支付平台响应
BatchNo	批量编号	String	构建委托扣款批量请求时必须设置
BatchDate	批量日期	String	构建委托扣款批量请求时必须设置
BatchTime	批量时间	long	不需设定 网上支付平台响应
AgentAmount	委托扣款总金额	double	构建委托扣款批量请求时必须设置

AgentCount	委托扣款总笔数	int	构建委托扣款批量请求时必须设置
TrnxTypeNo	交易类型编码	String	
BatchStatus	批量状态	String	不需设定 网上支付平台响应 0: 批量待复核 1: 复核通过待发送 2: 复核驳回 3: 复核通过后已发送(等待处理) 4: 处理成功 5: 处理失败
AgentBatchDetail	委托扣款批量订单明细。	ArrayList	AgentBatchDetail 对象的 ArrayList 集合。

## 方法

```
public AgentBatch()
```

AgentBatch 的构造函数，生成一个空的委托扣款批量对象。

```
public AgentBatch(XMLDocument aXMLDocument)
```

构造 AgentBatch 对象，并使用 XML 文件初始对象的属性。

参数：aXMLDocument – 网上支付平台响应的订单查询结果

```
public AgentBatch addAgentBatchDetail(AgentBatchDetail aAgentBatchDetail)
```

新增一笔订单。

参数: aAgentBatchDetail– 订单对象

com.hitrust.trustpay.client.b2c.AgentSign

### 说明

商户端接口软件包实体类，代表委托扣款签约结果查询对象

### 属性

属性	属性名称	类型	说明
AgentSignNo	签约协议号	String	需设定
MerchantNo	商户终端代码	String	不需设定，网上支付平台响应
CertificateType	证件类型	String	不需设定，网上支付平台响应 I 为公民身份证
CertificateNo	证件号码	String	不需设定，网上支付平台响应
CardNo	账号后四位	String	不需设定，网上支付平台响应
SignDate	签约日期	String 格式为 YYYY-MM-DD HH:MM:SS	不需设定，网上支付平台响应
UnSignDate	解约日期	String 格式为 YYYY-MM-DD HH:MM:SS	不需设定，网上支付平台响应

SignStatus	协议状态	String	不需设定，网上支付平台响应 00 签约请求建立，等待签约 01 签约成功 02 签约失败 03 解约成功 99 签约状态未知
AccountType	签约银行卡类型	String	不需设定，网上支付平台响应 401 农行借记卡 403 农行准贷记卡 404 农行贷记卡

## 方法

`public AgentSign(XMLDocument aXMLDocument)`

构造 `AgentSign` 对象，并使用 XML 文件初始对象的属性。

参数：aXMLDocument – 网上支付平台响应的订单查询结果

`com.hitrust.trustpay.client.b2c.QueryAgentSignRequest`

## 说明

商户端接口软件包处理类，负责委托扣款签约结果查询的处理。

## 属性

属性	属性名称	类型	说明
----	------	----	----

AgentSignNo	签约协议号	String	需设定
-------------	-------	--------	-----

## 方法

1. `public TrxResponse postRequest ()`

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的  
`getValue(String aTag)`方法取得下列信息。

参数	参数名称	说明
TrxResponse	支付平台返回报文	以字符串表示的签约信息。 可用来初始签约对象。

`com.hitrust.trustpay.client.b2c.AgentBatchDetail`

## 说明

商户端接口软件包实体类，代表委托扣款订单对象。

## 属性

属性	属性名称	类型	说明
MerchantNo	商户编号	String	不需设定 网上支付平台响应

BatchNo	批量编号	String	
BatchDate	批量日期	String	
OrderNo	订单编号	String	构建委托扣款批量请求时必须设置
OrderAmount	订单金额	double	构建委托扣款批量请求时必须设置
Currency	订单币种	String	
CertificateNo	证件号码	String	构建委托扣款批量请求时必须设置
ContractID	签约协议号	String	构建委托扣款批量请求时必须设置
ProductID	商品编号	String	构建委托扣款批量请求时必须设置
ProductName	商品名称	String	构建委托扣款批量请求时必须设置
ProductNum	商品数量	int	构建委托扣款批量请求时必须设置
ExpiredDate	订单有效期	int	构建委托扣款批量请求时必须设置
OrderStatus	订单状态	String	不需设定 网上支付平台响应 X: 未处理 0: 批量待复核 1: 成功 2: 失败 3: 无回应

## 方法

```
public AgentBatchDetail()
```

**AgentBatchDetail** 的构造函数，生成一个空的委托扣款订单对象。

```
public AgentBatchDetail(XMLDocument aXMLDocument)
```



构造 `AgentBatchDetail` 对象，并使用 XML 文件初始对象的属性。

参数: `aXMLDocument` – 网上支付平台响应的订单查询结果

## `com.hitrust.trustpay.client.b2c.B2CAgentBatchRequest`

### 说明

商户端接口软件包业务处理类，负责商户提交委托扣款批量交易的处理。

### 属性

属性	属性名称	类型	说明
<code>iAgentBatch</code>	委托扣款批量对象	<code>AgentBatch</code>	必须设定
<code>iAgentBatchDetailList</code>	批次明细对象集合	<code>Vector</code> <code>AgentBatchDetail</code> 对象的 <code>Vector</code> 集合	必须设定

### 方法

```
public TrxResponse postRequest()
```

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：根据返回码判断交易是否成功。

```
public void setAgentBatch(AgentBatch aAgentBatch)
```

设置委托扣款批量对象。

```
public void addAgentBatchDetail(AgentBatchDetail iAgentBatchDetail)
```

## 添加批次明细对象

`com.hitrust.trustpay.client.b2c.B2CAgentBatchQueryRequest`

## 说明

商户端接口软件包业务处理类，负责商户提交委托扣款批量结果查询交易的处理。

## 属性

属性	属性名称	类型	说明
BatchNo	批次号	String	必须设定
BatchDate	批次日期	String	必须设定

## 方法

```
public TrxResponse postRequest()
```

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的

`getValue(String aTag)`方法取得下列信息：

参数	参数名称	说明
AgentBatch	委托扣款批量对象	

`com.hitrust.trustpay.client.b2c.Insure`

## 说明

商户端接口软件包实体类，代表消费者购买的保险信息。

## 属性

属性	属性名称	类型	说明
Type	保险支付类型	string	必须设定，保险支付类型包括三种：1 一般支付、2 理财支付、3 指定支付。
Furl	验证失败信息的商户 Url	string	如果支付类型是保险理财支付时必须设定，其他支付类型没有此要求。
Order	保单信息	InsureOrder	必须设定
User	用户信息	InsureUser	Type 是 2 理财支付后者 3 指定支付时必须设定。

## 方法

```
public Insure()
```

默认构造函数，构造 **Insure** 对象

```
public Insure(XMLDocument aXMLDocument)
```

带参数的构造函数

参数 XMLDocument：请求报文

```
public XMLDocument getXMLDocument()
```

获取保单报文，返回报文的 XML 文档格式

`com.hitrust.trustpay.client.b2c.InsureOrder`

### 说明

商户端接口软件包实体类，代表消费者在商户网站购买的保单明细。

### 属性

属性	属性名称	类型	说明
OrderItems	保单明细	ArrayList	必须设定
XMLDocument	取得对象的 XML 文档	XMLDocument	只读属性

### 方法

```
public InsureOrder()
```

默认构造函数，构造 **InsureOrder** 对象

```
public InsureOrder(XMLDocument aXMLDocument)
```

带参数的构造函数

参数 XMLDocument：请求报文

```
public InsureOrder addOrderItem(InsureOrderItem aOrderItem)
```

将保单明细添加到保单中。

```
public InsureOrder ClearOrderItem()
```

清除保单明细

`com.hitrust.trustpay.client.b2c.InsureOrderItem`

### 说明

商户端接口软件包实体类，代表消费者购买的保单明细。

## 属性

属性	属性名称	类型	说明
Name	保险名称	string	必须设定
Code	保险代码	string	必须设定
Category	险种信息	string	必须设定
Mode	销售方式	string	必须设定
Amount	投保金额	double	必须设定
XMLDocument	取得对象的 XML 文档	XMLDocument	只读属性

## 方法

```
public InsureOrderItem()
```

默认构造函数，构造 InsureOrderItem 对象

```
public InsureOrderItem(double aAmount, string aName, string aCode, string  
aCategory, string aMode)
```

带参数的构造函数，参数类型请参考 InsureOrderItem 对象属性说明。

```
public InsureOrderItem(XMLDocument aXMLDocument)
```

带参数的构造函数

参数 XMLDocument：请求报文

**com.hitrust.trustpay.client.b2c.InsureUser**

## 说明

商户端接口软件包实体类，代表保险直销支付中投保人的信息。

## 属性

属性	属性名称	类型	说明
Name	投保人姓名	string	必须设定
CertificateType	投保人证件类型	string	必须设定
CertificateNo	投保人证件号码	string	必须设定
CardNo	银行卡号	string	必须设定
XMLDocument	取得对象的 XML 文档	XMLDocument	只读属性

## 方法

```
public InsureUser()
```

默认构造函数，构造 InsureUser 对象

```
public InsureUser(string aName, string aCertificateType, string aCertificateNo,  
string aCardNo)
```

带参数的构造函数，参数类型请参考 InsureUser 对象的属性说明。

```
public InsureUser(XMLDocument aXMLDocument)
```

带参数的构造函数

参数 XMLDocument：请求报文

**com.hitrust.trustpay.client.b2c.OnlineRemitRequest**

## 说明

商户端接口软件包业务处理类，负责网上付款信息发送交易的处理。

## 属性

属性	属性名称	类型	说明
TotalCount	付款总笔数	int	必须设定
TotalAmount	付款总金额	double	必须设定
SerialNumber	付款批量批次号	string	必须设定
CheckType	确认形式	string	目前暂时只支持,0 有待业务提需求再开发
Remark	备注	string	
RemitDetail	付款详情	ArrayList	

## 方法

```
public OnlineRemitRequest()
```

构造 OnlineRemitRequest 对象

```
protected internal override void checkRequest()
```

网上付款请求信息是否合法

```
protected internal override TrxResponse constructResponse(XMLDocument  
aResponseMessage)
```

回传交易响应对象。

**com.hitrust.trustpay.client.b2c.OnlineRmtCardVerifyRequest**

## 说明

商户端接口软件包业务处理类，负责客户身份验证请求的处理。

## 属性

属性	属性名称	类型	说明
BankCardNo	银行卡号	string	必须设定
AccountName	户名	string	必须设定

## 方法

```
public OnlineRmtCardVerifyRequest()
```

构造 **OnlineRmtCardVerifyRequest** 对象

```
public OnlineRmtCardVerifyRequest(XMLDocument aXMLDocument)
```

使用 XML 文件初始对象的属性

```
protected internal override void checkRequest()
```

支付请求信息是否合法

```
protected internal override TrxResponse constructResponse(XMLDocument  
aResponseMessage)
```

回传交易响应对象

```
com.hitrust.trustpay.client.b2c.OnlineRmtQueryResultRequest
```



## 说明

商户端接口软件包业务处理类，负责网上付款交易结果查询的处理。

## 属性

属性	属性名称	类型	说明
SerialNumber	付款流水号	string	必须设定
PayAccount	付款方账号	string	必须设定
ReceiveAccount	收款方账号	string	
StartTime	起始日期	string	
EndTime	截止日期	string	

## 方法

```
public OnlineRmtQueryResultRequest()
```

构造 OnlineRmtQueryResultRequest 对象

```
protected internal override void checkRequest()
```

网上付款交易结果查询信息是否合法

```
protected internal override TrxResponse constructResponse(XMLDocument  
aResponseMessage)
```

回传交易响应对象。

**com.hitrust.trustpay.client.b2c.OnlineRmtBatch**

## 说明

网上批次付款信息

## 属性

属性	属性名称	类型	说明
QueryResults	查询结果明细	ArrayList	必须设定

## 方法

```
public OnlineRmtBatch()
```

构造 OnlineRmtBatch 对象

```
public OnlineRmtBatch addQueryResult(QueryResult aQueryResult)
```

新增订单

**com.hitrust.trustpay.client.b2c.QueryResult**

## 说明

商户端接口软件包实体类，代表网上付款交易结果查询结果

## 属性

属性	属性名称	类型	说明
QueryResultItems	订单明细	ArrayList	必须设定
SerialNumber	付款流水号	string	

TrnxTime	交易时间	string	
No	付款流水号	string	
PayAccountNo	付款方账号	string	
PayAccountName	付款方户名	string	
ReceiveAccountNo	收款方账号	string	
ReceiveAccountName	收款方户名	string	
PayAmount	付款金额	double	
Purpose	用途	string	
Status	交易状态	string	
FailReason	失败原因	string	

## 方法

```
public QueryResult()
```

构造 QueryResult 对象

```
public QueryResult(XMLDocument aXMLDocument)
```

构造 QueryResult 对象，并使用 XML 文件初始对象的属性。

```
public QueryResult initByString(String aOrderString)
```

通过字符串构造 QueryResult 对象

```
private QueryResult initByXMLDocument(XMLDocument aXMLDocument)
```

通过 XmlDocument 构造 QueryResult 对象

```
public virtual QueryResult addQueryResultItem(QueryResultItem
aQueryResultItem)
```

新增查询结果明细

`com.hitrust.trustpay.client.b2c.QueryResultItem`

**说明**

商户端接口软件包实体类，代表网上付款交易结果查询明细。

**属性**

属性	属性名称	类型	说明
SerialNumber	付款流水号	string	
No	付款流水号	string	
PayAccountNo	付款方账号	string	
PayAccountName	付款方户名	string	
ReceiveAccountNo	收款方账号	string	
ReceiveAccountName	收款方户名	string	
PayAmount	付款金额	double	
Purpose	用途	string	
Status	交易状态	string	
FailReason	失败原因	string	

## 方法

```
public QueryResultItem()
```

QueryResultItem 默认构造函数

```
public QueryResultItem(XMLDocument aXMLDocument)
```

QueryResultItem 构造函数。使用 XML 文件初始对象的属性。

**com.hitrust.trustpay.client.b2c.AgentSign**

## 说明

商户端接口软件包实体类，代表委托扣款签约信息。

## 属性

属性	属性名称	类型	说明
AgentSignNo	委托扣款签约协议号	String	支付平台返回信息
MerchantNo	商户终端代码	String	支付平台返回信息
CardNo	签约银行卡后四位卡号	String	支付平台返回信息
AccountType	签约银行卡类型	String	支付平台返回信息 401: 农行借记卡 403: 农行准贷记卡 404: 农行贷记卡
CertificateNo	签约证件号码	String	支付平台返回信息

CertificateType	签约证件类型	String,	支付平台返回信息， 1： 公民身份证
SignStatus	签约状态	String	支付平台返回信息  00： 委托扣款签约请求建立， 等待签约  01： 签约成功  02： 签约失败  03： 解约成功  99： 签约状态未知
SignDate	签约日期	String,格式为 YYYY-MM-DD HH:MM:SS	支付平台返回信息
UnSignDate	解约日期	String ， 格 式 为 YYYY-MM-DD HH:MM:SS	支付平台返回信息

## 方法

1. public AgentSign(XMLDocument aXMLDocument)

构造 AgentSign 对象，并使用 XML 文件初始对象的属性。

参数：aXMLDocument – 网上支付平台响应的签约查询结果

**com.hitrust.trustpay.client.b2c.QueryAgentSignRequest**

## 说明

商户端接口软件包业务处理类，负责商户提交委托扣款签约查询交易的处理。

## 属性

属性	属性名称	类型	说明
AgentSignNo	委托扣款签约协议号	String	必须设定

## 方法

2. `public TrxResponse postRequest ()`

发送交易请求至网上支付平台，并接收网上支付平台的响应。

回传值：交易响应对象。如果交易响应成功，则可以使用交易响应对象的  
`getValue(String aTag)`方法取得下列信息。

参数	参数名称	说明
TrxResponse	支付平台返回报文	以字符串表示的签约信息。 可用来初始签约对象。

## 9. 附录四、常见问题及解决办法

根据日常维护经验，这里总结了一些最常见的问题，以及解决问题的思路。

### 1. 无法连接

#### ◆ 一般表现：

页面提示信息“1202 无法连线网上支付平台”；

页面提示信息“无法连接 443 端口”；

#### ◆ 原因分析：

商户服务器网络环境不通；

商户服务器网络环境权限控制引起；

#### ◆ 问题解决思路：

在商户服务器上用浏览器访问 [www.95599.cn](http://www.95599.cn)，查看网络是否通畅，如果没有问题，可

能是防火墙或者其他网络设置阻止了通讯，需要商户根据实际情况去查找具体原因；

### 2. WSE3.0 安装不上

#### ◆ 问题解决思路：

请上 microsoft 官方网站查找问题解决办法，网址：<http://www.microsoft.com/en-us/download/details.aspx?id=14089>；

### 3. 无法读取证书文档

#### ◆ 一般表现：

页面提示信息：TrustPayClient 错误‘80131500’无法读取证书文档；



#### ◆ 原因分析：

配置文件中提到 3 个证书文档（网上支付平台证书、农行根证书文件和商户证书储存目录档名），能够引起这种错误的情况很多，主要是以下几种：

证书文档路径不正确；

权限不够；

证书文档密码不正确；

其他原因；

#### ◆ 问题解决思路：

检查配置文件中各个证书文件的路径及密码是否正确；

检查这三个证书文件是否给 ASPNET 用户读的权限；

检查其他情况，主要是商户服务器的操作系统和网络环境权限设置等；

### 4. 返回报文签名验证失败

#### ◆ 一般表现：

页面提示信息：签名验证失败“2302”；

#### ◆ 原因分析：

商户号和证书文件不匹配；

其他原因；

#### ◆ 问题解决思路：

检查商户号和证书文件是否匹配；

### 5. 类型初始值设定引发异常

◆ 一般表现:

页面提示信息: ‘80131534’ “com.hitrust.trustpay.client.MerchantConfig” 的类型  
初始值设定项引发异常;

◆ 原因分析:

WSE 没有安装;

WSE 版本问题;

其他原因;

◆ 问题解决思路:

检查 WSE 是否已经安装, 版本为: Microsoft WSE 3.0 (不支持其他版本);

## 6. 服务器换 IP 后的通知问题

◆ 一般表现:

商户 B2C 服务器换 IP 后, 农行的通知消息仍然发到老的 IP 地址上, 新的 IP 地址没有接  
受到通知消息;

◆ 原因分析:

IP 地址变更后, DNS 仍然会缓存记住一段时间;

◆ 问题解决思路:

用 IP 地址的方式请求通知, 而不是原先域名的方式;