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

Java Edition
V2.0.8

修订历史纪录

日期	版本	说明	作者
2003/11/26	V0.1	草案	网上支付平台开发小组
2003/12/10	V0.2	修改	网上支付平台开发小组
2003-12-18	V0.3	修改	网上支付平台开发小组
2004-01-05	V1.0	支持硬件签名	网上支付平台开发小组
2004-03-29	V1.0.1	加强商户接收交易结果的说明	网上支付平台开发小组
2004-04-04	V1.0.2	更改接口开发软件包文件名	网上支付平台开发小组
2004-04-16	V1.0.3	新增商户指定支付类型，用以支持国际卡支付	网上支付平台开发小组
2004-05-31	V1.0.4	新增对于 zip 包里新加文件的说明	网上支付平台开发小组
2004-06-16	V1.1	新增对商户开发支付请求和接收页面时的提示	网上支付平台开发小组
2004-08-12	V1.1.1	新增对商户设定支付类型时的注意事项	网上支付平台开发小组
2004-09-27	V1.5	新增对通知商户支付结果的两种方式的说明	网上支付平台开发小组
2005-05-11	V2.0	1、修正无法写入日志的问题 2、优化程序执行效能 3、支持压缩下载对帐单 4、支持保存对帐单至文件、由文件读取对帐单 5、修正接收支付平台直接通知页面范例	网上支付平台开发小组
2005-01-06	V2.0.1	1、添加基金网上支付 2、添加指定日期指定时间交易对账单下载	网上支付平台开发小组
2008-02-14	V2.0.2	新增客户身份验证交易	网上支付平台开发小组
2008-06-20	V2.0.3	新增批量退款功能	网上支付平台开发小组
2009-05-31	V2.0.4	1、 新增批量退款功能 2、 新增委托扣款功能 3、 新增贷记卡支付功能	网上支付平台开发小组
2009-07-01	V2.0.5	1、 新增商户通过页面提交访问农行 b2c 服务的功能	网上支付平台开发小组

2010-03-01	V2.0.6	1. 农行借记卡和贷记卡支付方式合并	网上支付平台开发小组
2010-08-01	V2.0.7	防钓鱼支付修改，接口发送请求中增加买方 IP 地址字段 BuyIP	网上支付平台开发小组
2010-11-25	V2.0.8	加入保险支付的交易，删除基金相关交易，增加对退款订单号的支持	网上支付平台开发小组

目录

1. 简介 6
1.1 目的 6
1.2 功能描述 6
1.3 总体架构图 6
2. 接口开发软件包说明 7
3. 安装步骤 10
3.1 安装前检查 10
3.2 接口开发软件包安装 10
3.3 接口开发软件包配置 10
3.4 配置测试应用 10
4. 配置文件说明 11
5. 交易说明 13
5.1 交易流程 13
5.1.1 支付交易 13
5.1.2 确保支付结果正确送达商户网站的措施 15
5.1.3 其它交易 16
5.2 交易使用时机 17
5.3 支付请求 18
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 18
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 20
5.4 两种接收支付结果方式的区别 22
5.4.1 通过显示给消费者的支付结果接收页面通知商户 22
5.4.2 通过支付平台服务器通知商户 23
5.4.3 区别 25
5.5 支付结果接收页面 26
5.6 退货请求 26
5.7 订单查询 28
5.8 交易对账单下载 30
5.9 指定日期指定时间段交易对账单下载 31
5.10 农行卡身份验证交易请求 32
5.11 卡验证结果接收页面 32
5.12 身份验证交易请求 33
5.13 身份验证结果接收页面 33
5.14 批量退款发送请求 34
5.15 批量退款结果查询请求 35
5.16 委托扣款签约请求 36
5.17 委托扣款解约请求 37
5.18 委托扣款单笔代扣请求 38
5.19 委托扣款批量请求 39
5.20 委托扣款批量结果查询 40
5.21 贷记卡交易对账单下载 41
5.22 保险直销支付请求 42
5.22.1 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 42
5.22.2 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 45
5.23 付款信息发送请求 45

5.24 付款交易结果查询请求 46
5.25 付款银行卡状态查询请求 48

附录一、程序范例 50
A、支付请求范例 50
B、支付结果接收范例 52
C、从服务器直接接收支付结果页面范例 54
D、退货交易范例 55
E、订单查询交易范例 57
F、交易对账单下载范例 59
G、指定时间段交易对账单下载 61
H、卡验证请求范例 62
I、卡身份验证结果接收范例 63
J、身份验证请求范例 64
K、身份验证结果接收范例 65
L、批量退款发送请求范例 66
M、批量退款交易结果查询请求范例 69
N、委托扣款签约交易请求范例 70
O、委托扣款解约交易请求范例 71
P、委托扣款单笔代扣交易请求范例 73
Q、委托扣款批量交易请求范例 74
R、委托扣款批量结果查询交易请求范例 77
S、贷记卡对账单下载交易请求范例 79
T、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 81
U、商户本地返回错误页面范例 83
V、支付平台返回错误页面范例 84
W、保险直销支付请求范例 84
Y、商户通过页面传参数提交表单方式进行保险支付请求范例 87
Z、网上付款信息发送请求范例 90
AA、网上付款交易结果查询请求范例 92
AB、网上付款银行卡状态验证请求范例 94

附录二、响应码一览表 95

附录三、TRUSTPAY CLIENT API 98
COM.HITRUST.TRUSTPAY.CLIENT.TRXRESPONSE 98
COM.HITRUST.TRUSTPAY.CLIENT.B2C.ORDER 99
COM.HITRUST.TRUSTPAY.CLIENT.B2C.ORDERITEM 101
COM.HITRUST.TRUSTPAY.CLIENT.B2C.PAYMENTREQUEST 103
COM.HITRUST.TRUSTPAY.CLIENT.B2C.PAYMENTRESULT 105
COM.HITRUST.TRUSTPAY.CLIENT.B2C.QUERYORDERREQUEST 107
COM.HITRUST.TRUSTPAY.CLIENT.B2C.VOIDPAYMENTREQUEST 108
COM.HITRUST.TRUSTPAY.CLIENT.B2C.REFUNDREQUEST 109
COM.HITRUST.TRUSTPAY.CLIENT.B2C.SETTLEREQUEST 111
COM.HITRUST.TRUSTPAY.CLIENT.B2C.SETTLEFILE 112
COM.HITRUST.TRUSTPAY.CLIENT.B2C.IDENTITYVERIFYREQUEST 113
COM.HITRUST.TRUSTPAY.CLIENT.B2C.BATCH 115
COM.HITRUST.TRUSTPAY.CLIENT.B2C.OVERDUEBATCH 116
COM.HITRUST.TRUSTPAY.CLIENT.B2C.BATCHSENDREQUEST 118
COM.HITRUST.TRUSTPAY.CLIENT.B2C.QUERYBATCHREQUEST 119
COM.HITRUST.TRUSTPAY.CLIENT.B2C.OVERDUEREFUNDREQUEST 119
COM.HITRUST.TRUSTPAY.CLIENT.B2C.QUERYOVERDUEREFUNDREQUEST 120
COM.HITRUST.TRUSTPAY.CLIENT.B2C.B2CAGENTSIGNRESULT 122
COM.HITRUST.TRUSTPAY.CLIENT.B2C.B2CAGENTSIGNCONTRACTREQUEST 122

COM.HITRUST.TRUSTPAY.CLIENT.B2C.B2CAGENTUNSIGNCONTRACTREQUEST 124
COM.HITRUST.TRUSTPAY.CLIENT.B2C.B2CAGENTPAYMENTREQUEST 125
COM.HITRUST.TRUSTPAY.CLIENT.B2C.AGENTBATCH 126
COM.HITRUST.TRUSTPAY.CLIENT.B2C.AGENTBATCHDETAIL 128
COM.HITRUST.TRUSTPAY.CLIENT.B2C.B2CAGENTBATCHREQUEST 129
COM.HITRUST.TRUSTPAY.CLIENT.B2C.B2CAGENTBATCHQUERYREQUEST 130
COM.HITRUST.TRUSTPAY.CLIENT.B2C.INSURE 131
COM.HITRUST.TRUSTPAY.CLIENT.B2C.INSUREORDER 131
COM.HITRUST.TRUSTPAY.CLIENT.B2C.INSUREORDERITEM 132
COM.HITRUST.TRUSTPAY.CLIENT.B2C.INSUREUSER 133
COM.HITRUST.TRUSTPAY.CLIENT.B2C.ONLINEREMITREQUEST 133
COM.HITRUST.TRUSTPAY.CLIENT.B2C.ONLINERMTCARDVERIFYREQUEST 134
COM.HITRUST.TRUSTPAY.CLIENT.B2C.ONLINERMTQUERYRESULTREQUEST 135
COM.HITRUST.TRUSTPAY.CLIENT.B2C.ONLINERMTBATCH 136
COM.HITRUST.TRUSTPAY.CLIENT.B2C.QUERYRESULT 137
COM.HITRUST.TRUSTPAY.CLIENT.B2C.QUERYRESULTITEM 138

1. 简介

1.1 目的

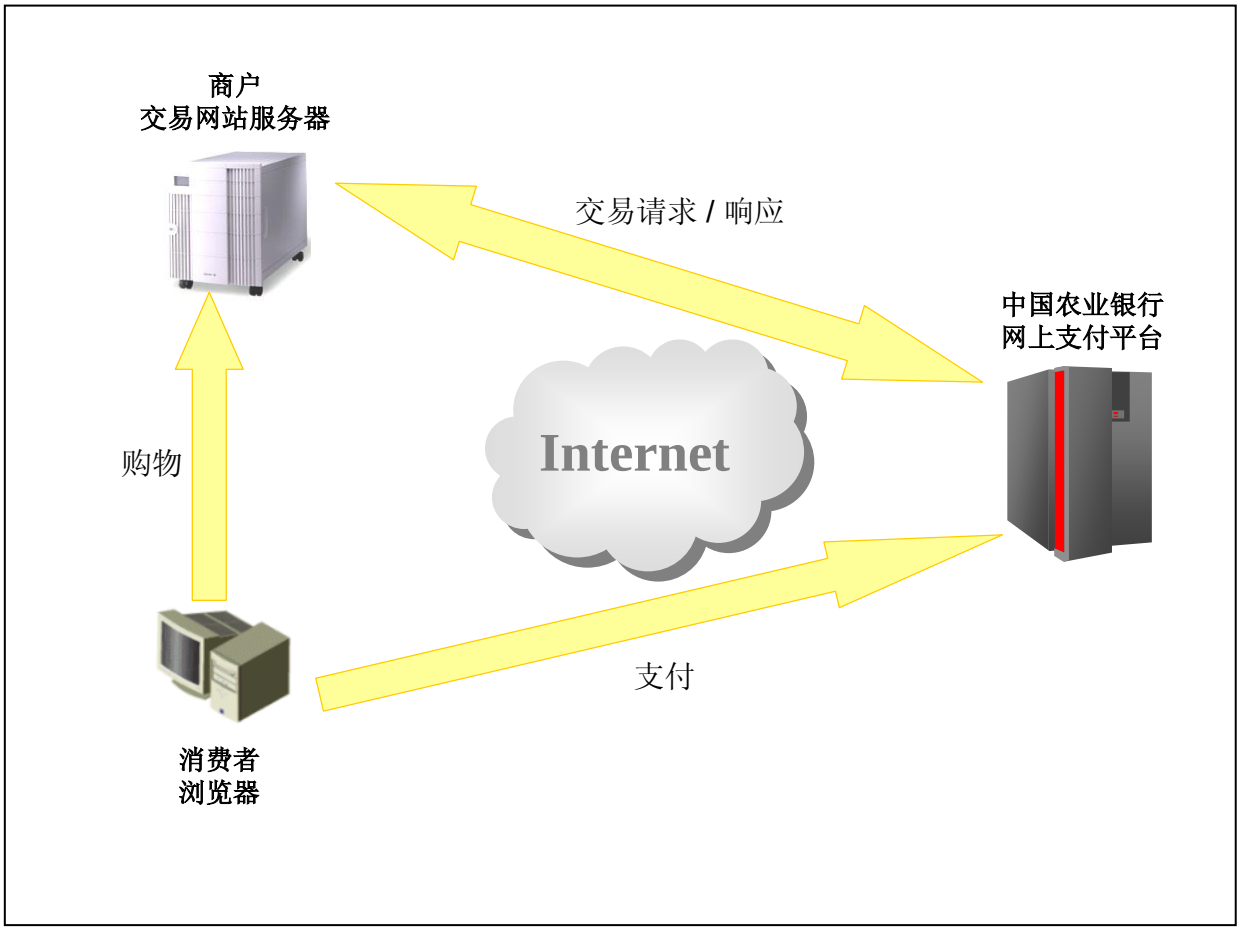
提供商户端交易网站通过中国农业银行网上支付平台提供的商户端开发软件包实现功能的编程指南。

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、取消支付、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-Java-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。

文件名称	说明
/docs/商户接口编程指南-Java-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
/lib/TrustPayClient-Vx.x.jar	农行网上支付平台商户端接口 Java 软件包。
/lib/TrustMerchant.properties	接口配置文件
/demo/Merchant.html	接口范例首页
/demo/MerchantPayment.html	支付请求交易范例页面
/demo/MerchantPayment.jsp	支付请求交易范例程序
/demo/MerchantPaymentIE.html	商户通过页面传参数表单提交支付请求范例页面；根据接口包类型不同，demo 中可能没有该文件
/demo/MerchantPaymentIE.jsp	商户通过页面传参数表单提交支付请求范例程序；根据接口包类型不同，demo 中可能没有该文件
/demo/MerchantResult.jsp	支付结果接收范例程序
/demo/ErrorPage.jsp	商户通过页面传参数表单提交支付请求时，商户接收支付平台返回错误的范例程序；根据接口包类型不同，demo 中可能没有该文件
/demo/ErrorPageInternal.jsp	商户通过页面传参数表单提交支付请求时，商户本地返回错误页面范例程序；根据接口包类型不同，demo 中可能没有该文件
/demo/ReciveServerPage.jsp	直接接收服务器支付结果范例程序
/demo/MerchantVoidPayment.html	取消支付交易范例页面
/demo/MerchantVoidPayment.jsp	取消支付交易范例程序
/demo/MerchantQueryOrder.html	订单查询交易范例页面
/demo/MerchantQueryOrder.jsp	订单查询交易范例程序
/demo/MerchantRefund.html	退货交易范例页面

/demo/MerchantRefund.jsp	退货支付交易范例程序
/demo/MerchantVoidRefund.html	取消退货范例页面
/demo/MerchantVoidRefund.jsp	取消退货范例程序
/demo/MerchantSettle.html	结算对账单下载范例页面
/demo/MerchantSettle.jsp	结算对账单下载范例程序
/demo/MerchantTrxSettle.html	交易对账单下载范例页面
/demo/MerchantTrxSettle.jsp	交易对账单下载范例程序
/demo/MerchantTrxSettleByHour.html	交易对账单下载范例页面
/demo/MerchantTrxSettleByHour.jsp	交易对账单下载范例程序
/demo/MerchantCardVerify.jsp	卡验证范例程序
/demo/IdentityVerify.html	身份验证范例页面
/demo/IdentityVerify.jsp	身份验证范例程序
/demo/IdentityVerifyResult.jsp	身份验证结果接收范例程序
/demo/MerchantBatchSend.html	批量退款发送页面
/demo/MerchantBatchSend.jsp	批量退款发送程序
/demo/MerchantQueryBatch.html	批量退款结果查询页面
/demo/MerchantQueryBatch.jsp	批量退款结果查询程序
/demo/MerchantQueryOverdueRefund.html	批量退款结果查询页面
/demo/MerchantQueryOverdueRefund.jsp	批量退款结果查询页面
/demo/B2CAgentSignContract.html	委托扣款签约页面
/demo/B2CAgentSignContract.jsp	委托扣款签约程序
/demo/B2CAgentUnsignContract.html	委托扣款解约页面
/demo/B2CAgentUnsignContract.jsp	委托扣款解约程序
/demo/B2CAgentSignResult.jsp	委托扣款签约/解约结果接收范例
/demo/B2CAgentBatch.html	委托扣款批量页面
/demo/B2CAgentBatch.jsp	委托扣款批量程序
/demo/B2CQueryAgentBatch.html	委托扣款批量处理结果查询页面
/demo/B2CQueryAgentBatch.jsp	委托扣款批量处理结果查询程序
/demo/MerchantCreditTrxSettle.html	贷记卡交易对账单下载范例页面
/demo/MerchantCreditTrxSettle.jsp	贷记卡交易对账单下载范例程序
/demo/MerchantPaymentInsure.html	保险直销支付页面
/demo/MerchantPaymentInsure.jsp	保险直销支付程序

/demo/MerchantPaymentInsureIE.html	商户通过页面传参数表单提交保险直销支付请求范例页面
/demo/MerchantPaymentInsureIE.jsp	商户通过页面传参数表单提交保险直销支付请求范例程序
/demo/MerchantPaymentInsure.html	保险支付请求范例页面
/demo/MerchantPaymentInsure.jsp	保险支付请求范例程序
/demo/MerchantPaymentInsureIE.html	浏览器提交模式保险支付请求范例页面
/demo/MerchantPaymentInsureIE.jsp	浏览器提交模式保险支付请求范例程序
/demo/SendOnlineRemitInfo.html	网上付款信息发送请求范例页面
/demo/SendOnlineRemitInfo.jsp	网上付款信息发送请求范例程序
/demo/OnlineRmtQueryResult.html	网上付款交易结果查询请求范例页面
/demo/OnlineRmtQueryResult.jsp	网上付款交易结果查询请求范例程序
/demo/OnlineRmtCardStatusQuery.html	网上付款银行卡状态查询请求范例页面
/demo/OnlineRmtCardStatusQuery.jsp	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，如果有指向农行网站的图片链接，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 JDK 1.3 标准。

2、请确定服务器已经安装了下列软件：

◆ Sun JSSE V1.0 或更高的版本

3.2 接口开发软件包安装

1、将银行提供的接口开发软件包 TrustPayClient-Java-Vx.x.x.zip（x.x.x 为接口开发软件包的版本号）解压缩到商户自定的安装目录中。

2、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、将 TrustPayClient-Vx.x.x.jar 加入应用服务器的 CLASSPATH 中。

3.3 接口开发软件包配置

1、开启接口配置文件 TrustMerchant.properties，依照银行提供的信息设定相对应的参数。并将 TrustMerchant.properties 所在的目录加入应用服务器的 CLASSPATH 中。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、配置 WebApp，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；商户通过浏览器提交，请输入：http://*your.server.name*/*your.virtual.directory*/MerchantPayment.html 确定接口软件包是否已正确安装及配置

4. 配置文件说明

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 http：使用 HTTP 通讯方式 https：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 1 或 2
	TrustPayTrxURL	字符串	网上支付平台交易网址
	TrustPayIETrxURL	字符串	商户通过浏览器提交方式，网上支付平台交易网址；根据接口包类型不同，配置文件中可能没有该项
	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
商户资料段	MerchantID	字符串	商户编号
商户系统配置段	LogPath	字符串	日志文件存放绝对目录。*请勿输入文件名
	MerchantErrorURL	字符串	商户通过浏览器提交接收网上支付平台返回错误页面。根据接口包类型不同，配置文件中可能没有该项
	MerchantKeyStoreType	数字	证书储存媒体 0: File 1: 硬件签名服务器
	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 KeyStoreType0 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。

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》

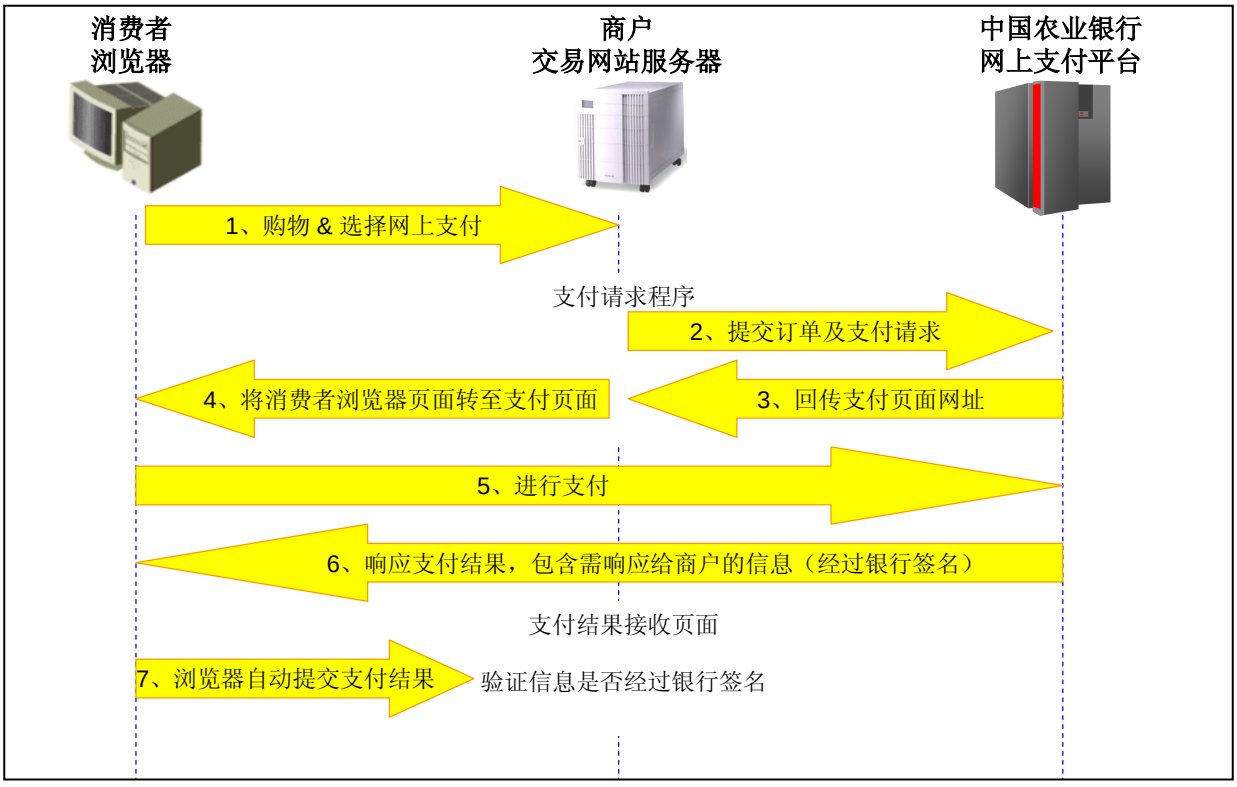
5.1 交易流程

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。

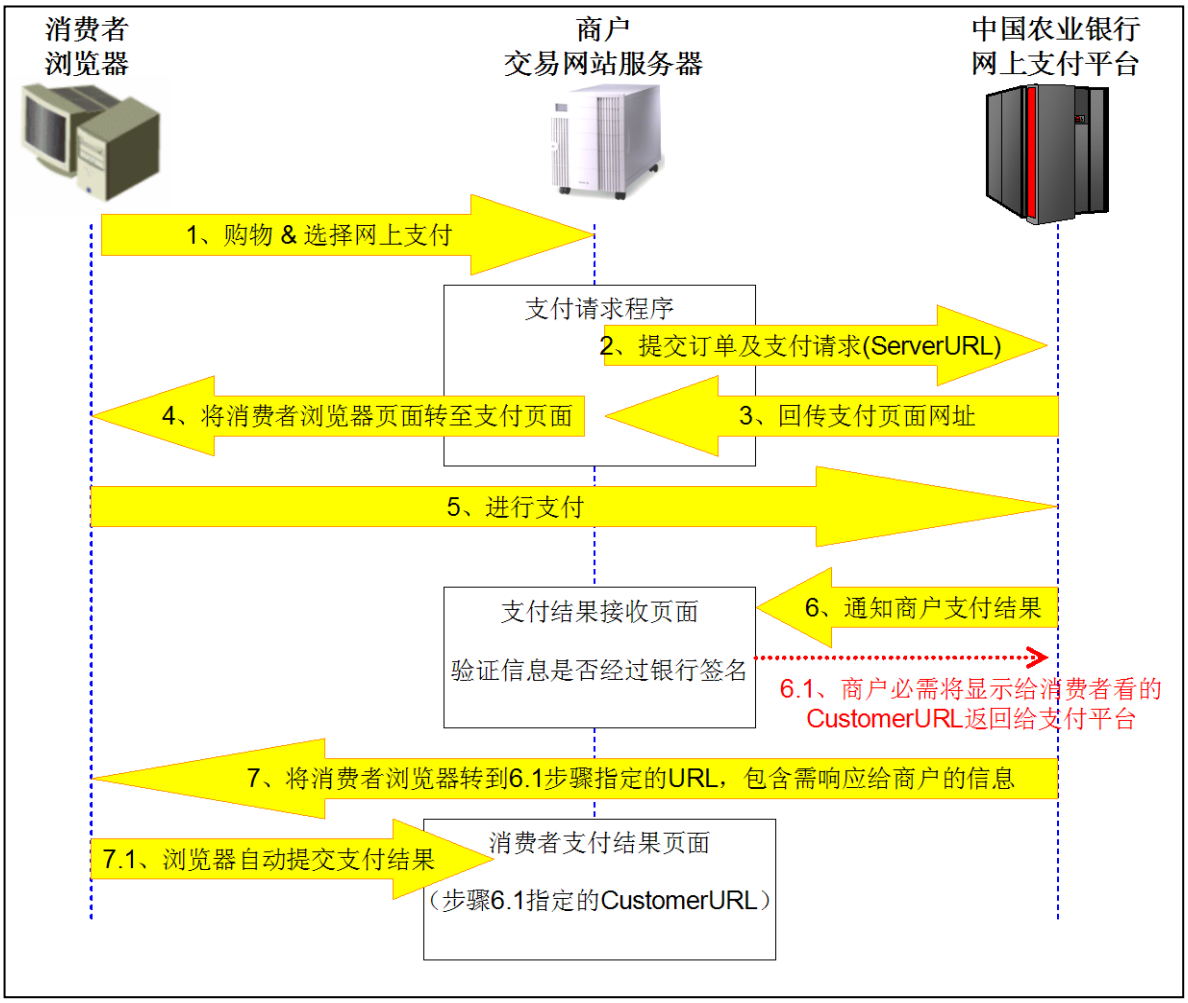
5.1.1 支付交易

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“支付结果接收程序”。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述：

页面通知支付结果方式：



服务器通知支付结果方式：



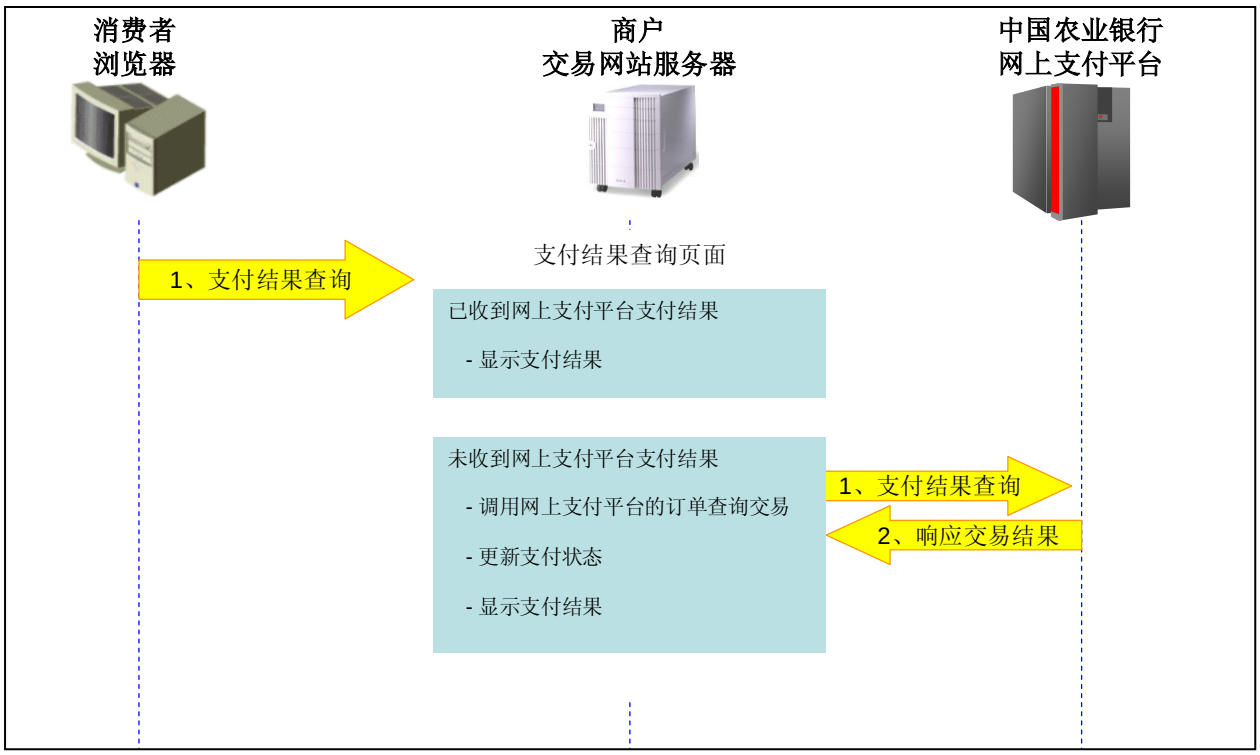
5.1.2 确保支付结果正确送达商户网站的措施

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。

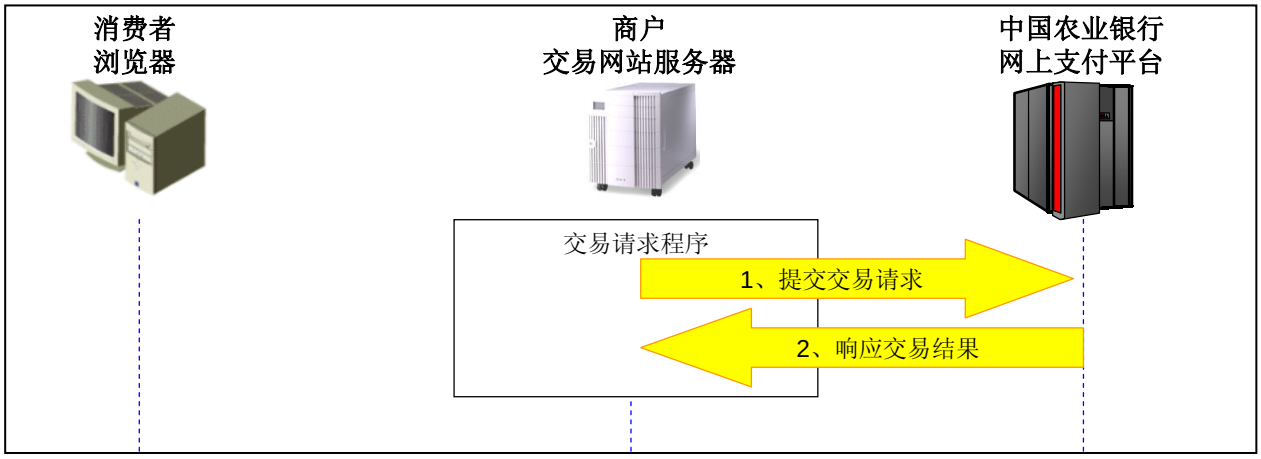


◆ 通知商户支付成功

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（取消支付、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。交易过程如下图所述：



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。

- **取消支付**

所有当日的订单（22:00 为切账时间）商户可以使用取消支付的交易来撤销已经支付成功的订单。隔日的订单将无法取消支付，必须使用退货。

- **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 22:00 进行切账作业，切账作业后批次号将会累加。商户可在每日早上 08:00 后进行交易对账单的下载，确定是否有未回传的成功交易。

例如：商户发起下载 2003/11/20 的交易对账单请求，网上支付平台将会回传 2003/11/19 22:00 到 2003/11/20 22:00 中间所有的成功交易。

- **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）
- 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和国际卡支付以及贷记卡支付三种，另外为了满足商户要求，我们还增加了一种将农行卡和贷记卡两支付方式一起显示的支付方式。

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。

◆ 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 ErrorPageInternal.jsp 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 ErrorPage.jsp，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 ErrorPag.jsp 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

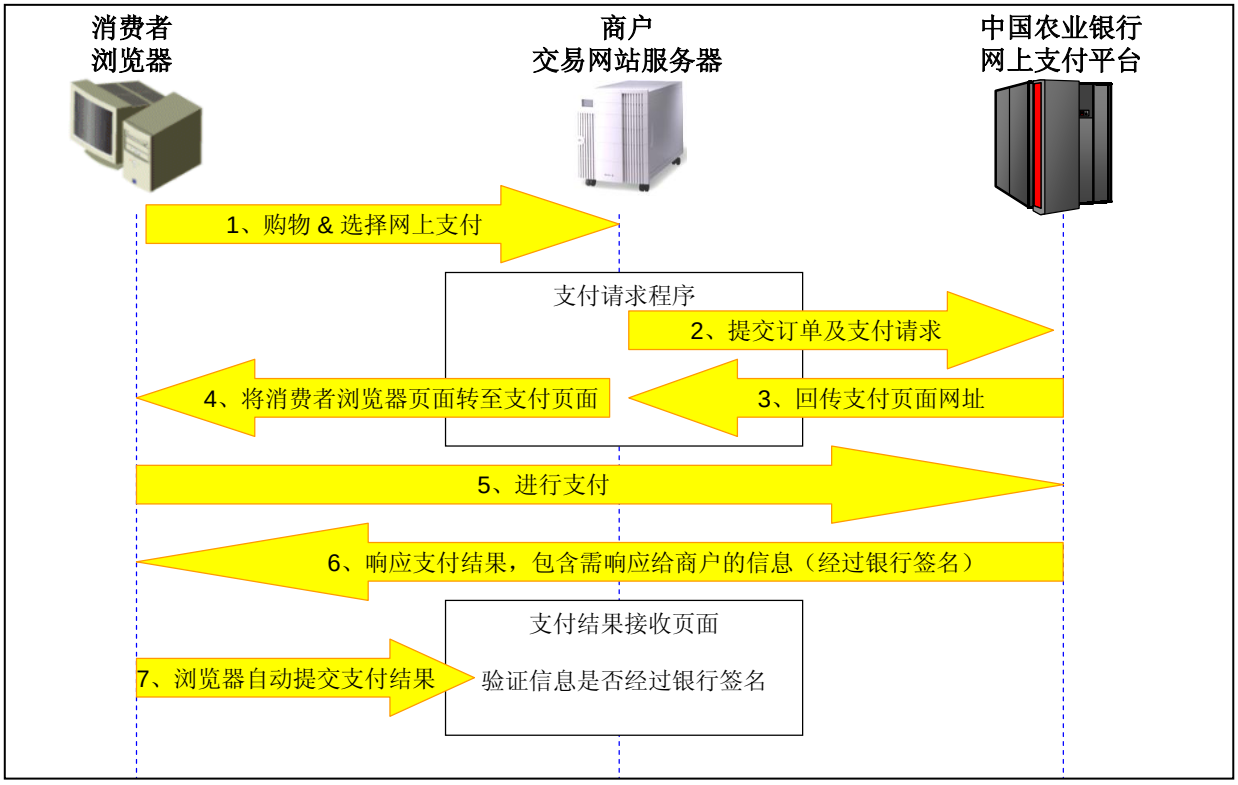
消费者在网上支付平台上进行在线支付的操作，支付成功后，网上支付平台会将支付结果通知给商户，目前通知方式有两种：**通过显示给消费者的支付结果接收页面通知商户**和**通过支付平台服务器通知商户**

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：



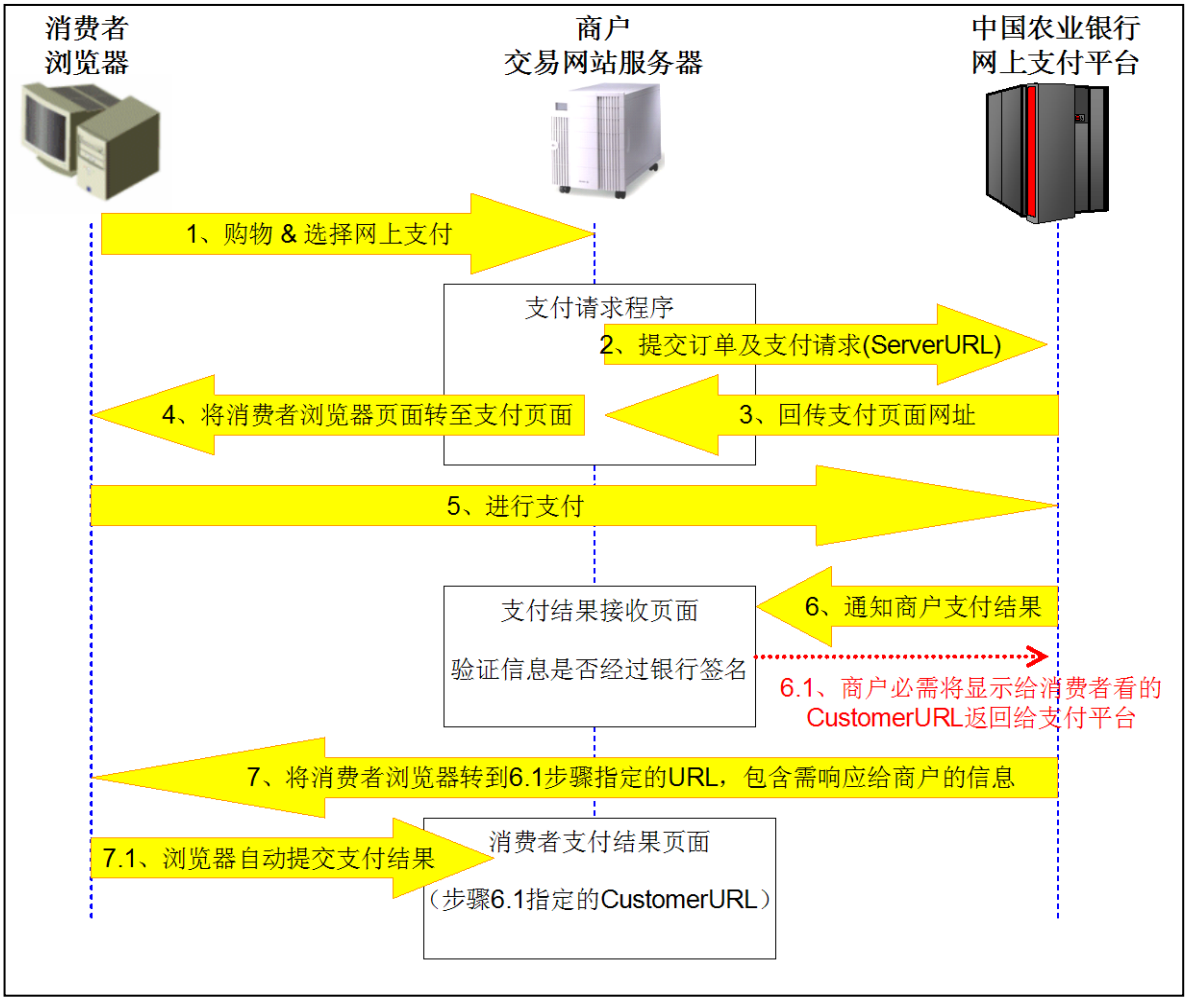
5.4.2 通过支付平台服务器通知商户

商户选择此种接收支付结果通知的方式，需要开发两支页面：

- 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 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。

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，则商户可以取得支付结果对象的其他属性来进行后续的作业

- OrderNo 订单号
- Amount 订单金额
- BatchNo 交易批次号
- VoucherNo 交易凭证号（建议使用 iRspRef 作为对账依据）
- 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
- NotifyType 支付结果通知方式
- iRspRef 银行返回交易流水号

5、若支付失败，可以使用支付结果对象的 getReturnCode ()及 getErrorMessage()方法取得支付失败原因。

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

5.6 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）
- NewOrderNo 退款订单号
- 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，则商户可以取得交易结果对象的其他属性来进行后续的作业

- OrderNo 订单号
- NewOrderNo 退款订单号
- 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）
- iRspRef 银行返回交易流水号

6、若交易失败，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。

5.7 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 getOrderItems()方法取得订单明细。

7、若交易失败，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。

5.8 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、若交易失败，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。

5.9 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 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 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。

5.10 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 getReturnCode ()及 getErrorMessage()方法取得交易失败原因。

5.11 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 getReturnCode ()及 getErrorMessage()方法取得支付失败原因。

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

5.12 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 getReturnCode ()及 getErrorMessage()方法取得交易失败原因。

5.13 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 getReturnCode ()及 getErrorMessage()方法取得支付失败原因。

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

5.14 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

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

- TrxType 交易类型

- TotalCount 批量退款订单总笔数
- TotalAmount 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 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。

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”。

5.15 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 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。

5.16 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“I”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）
- 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 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。

5.17 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，目前只支持公民身份证，请设置为“I”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 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。

5.18 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）
- 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 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。

5.19 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）
◆ BatchDate 批次日期（必要信息）
◆ AgentCount 批次总比数（必要信息，每批次最多包含 100 笔）
◆ AgentAmount 批次总金额（必要信息）
◆ OrderNo 订单编号（每笔订单必要信息，最多为 100 条）
◆ OrderAmount 订单金额（每笔订单必要信息，最多为 100 条）
◆ CertificateNo 证件号码（每笔订单必要信息，最多为 100 条）
◆ ContractID 签约协议号（每笔订单必要信息，最多为 100 条）
◆ ProductID 商品编号（每笔订单必要信息，最多为 100 条）
◆ ProductName 商品名称（每笔订单必要信息，最多为 100 条）
◆ ProductNum 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 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。

5.20 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
- 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 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。

5.21 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，新增加一种对账类型）

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 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。

5.22 保险直销支付请求

5.22.1 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）
- 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，分别是一般支付、理财支付和指定支付
- Furl 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，国际卡支付两种，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。

5.22.2 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 ErrorPageInternal.jsp 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 ErrorPage.jsp，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 ErrorPag.jsp 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

5.23 付款信息发送请求

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

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

5.24 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

SerialNumber	网上付款交易请求流水号（与 ReceiveAccount 至少填一个）
ReceiveAccount	收款方账号（与 SerialNumber 至少填一个）

StartTime	所要查询流水的起始时间
EndTime	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：

SerialNumber	网上付款批量流水号
TrnxTime	交易时间
TotalCount	批量总笔数
TotalAmount	批量总金额
Status	批量状态（0：未处理；1：处理中；2：处理成功；3：处理失败）
SuccessAmount	处理成功金额
SuccessCount	处理成功笔数
FailAmount	处理失败金额
FailCount	处理失败笔数
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

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。

5.25 付款银行卡状态查询请求

付款银行卡状态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。

附录一、程序范例

A、支付请求范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<%
//1、取得支付请求所需要的信息
String tOrderNo         = request.getParameter("OrderNo"        );
String tOrderDesc       = request.getParameter("OrderDesc"      );
String tOrderDate       = request.getParameter("OrderDate"      );
String tOrderTime       = request.getParameter("OrderTime"      );
String tOrderAmountStr  = request.getParameter("OrderAmount"    );
String tOrderURL        = request.getParameter("OrderURL"       );
String tProductType     = request.getParameter("ProductType"    );
String tPaymentType     = request.getParameter("PaymentType"    );
String tNotifyType      = request.getParameter("NotifyType"     );
String tResultNotifyURL = request.getParameter("ResultNotifyURL");
String tMerchantRemarks = request.getParameter("MerchantRemarks");
double  tOrderAmount     = Double.parseDouble(tOrderAmountStr);
String tPaymentLinkType = request.getParameter("PaymentLinkType");
String tBuyIP =            request.getParameter("BuyIP");
//2、生成订单对象
Order tOrder = new Order();
tOrder.setOrderNo    (tOrderNo    ); //设定订单编号 (必要信息)
tOrder.setOrderDesc  (tOrderDesc  ); //设定订单说明
tOrder.setOrderDate  (tOrderDate  ); //设定订单日期 (必要信息 - YYYY/MM/DD)
tOrder.setOrderTime  (tOrderTime  ); //设定订单时间 (必要信息 - HH:MM:SS)
tOrder.setOrderAmount(tOrderAmount); //设定订单金额 (必要信息)
tOrder.setOrderURL   (tOrderURL   ); //设定订单网址
```

```
    tOrder.setBuyIP    (tBuyIP    ); //设定订单 IP

    //3、生成订单订单对象，并将订单明细加入订单中（可选信息）

    tOrder.addOrderItem(new OrderItem("IP000001", "中国移动 IP 卡", 100.00f, 1));

    tOrder.addOrderItem(new OrderItem("IP000002", "网通 IP 卡"    ,  90.00f, 2));


    //4、生成支付请求对象

    PaymentRequest tPaymentRequest = new PaymentRequest();

    tPaymentRequest.setOrder       (tOrder       ); //设定支付请求的订单 （必要信息）

    tPaymentRequest.setProductType(tProductType); //设定商品种类 （必要信息）

                                   //PaymentRequest.PRD_TYPE_ONE：非实体商品，如服务、IP 卡、下载
MP3、...

                                   //PaymentRequest.PRD_TYPE_TWO：实体商品

    tPaymentRequest.setPaymentType(tPaymentType); //设定支付类型

                                   //PaymentRequest.PAY_TYPE_ABC：农行卡支付

                                   //PaymentRequest.PAY_TYPE_INT：国际卡支付

    tPaymentRequest.setNotifyType(tNotifyType);    //设定商户通知方式

                                                   //0：URL 页面通知

                                                   //1：服务器通知

    tPaymentRequest.setResultNotifyURL(tResultNotifyURL); //设定支付结果回传网址 （必要信息）

    tPaymentRequest.setMerchantRemarks(tMerchantRemarks); //设定商户备注信息

    tPaymentRequest.setPaymentLinkType(tPaymentLinkType);    //设定支付接入方式

                                       // PaymentRequest.PAY_LINK_TYPE_NET internet 网络接入

                                       // PaymentRequest.PAY_LINK_TYPE_MOBILE mobile 网络接入

                                       // PaymentRequest.PAY_LINK_TYPE_TV 数字电视网络接入

    //5、传送支付请求并取得支付网址

    TrxResponse tTrxResponse = tPaymentRequest.postRequest();

    if (tTrxResponse.isSuccess()) {

       //6、支付请求提交成功，将客户端导向支付页面

       response.sendRedirect(tTrxResponse.getValue("PaymentURL"));

    }

    else {

       //7、支付请求提交失败，商户自定后续动作
```

```
%>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-支付请求</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>支付请求<br>
<%
  out.println("ReturnCode   = [" + tTrxResponse.getReturnCode  () + "]<br>");
  out.println("ErrorMessage = [" + tTrxResponse.getErrorMessage() + "]<br>");
}
%>
<a href='Merchant.html'>回商户首页</a></CENTER>
</BODY></HTML>
```

B、

B、支付结果接收范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-支付结果接收</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>支付结果<br>
<%
//1、取得 MSG 参数，并利用此参数值生成支付结果对象
PaymentResult tResult = new PaymentResult(request.getParameter("MSG"));

//2、判断支付结果状态，进行后续操作
if (tResult.isSuccess()) {
  //3、支付成功
  out.println("TrxType         = [" + tResult.getValue("TrxType"        ) + "]<br>");
```

```
    out.println("OrderNo          = [" + tResult.getValue("OrderNo"         ) + "]<br>");
    out.println("Amount           = [" + tResult.getValue("Amount"          ) + "]<br>");
    out.println("BatchNo          = [" + tResult.getValue("BatchNo"         ) + "]<br>");
    out.println("VoucherNo        = [" + tResult.getValue("VoucherNo"       ) + "]<br>");
    out.println("HostDate         = [" + tResult.getValue("HostDate"        ) + "]<br>");
    out.println("HostTime         = [" + tResult.getValue("HostTime"        ) + "]<br>");
    out.println("MerchantRemarks = [" + tResult.getValue("MerchantRemarks") + "]<br>");
    out.println("PayType          = [" + tResult.getValue("PayType"         ) + "]<br>");
    out.println("NotifyType       = [" + tResult.getValue("NotifyType"      ) + "]<br>");
    out.println("TrnxNo           = [" + tResult.getValue("iRspRef"         ) + "]<br>");
  }
  else {
    //4、支付失败
    out.println("ReturnCode    = [" + tResult.getReturnCode  () + "]<br>");
    out.println("ErrorMessage = [" + tResult.getErrorMessage() + "]<br>");
  }
  %>
  <a href='Merchant.html'>回商户首页</a></CENTER>
  </BODY></HTML>
```

C、从服务器直接接收支付结果页面范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<%
//0、设定商户结果显示页面
String tMerchantPage = "";

//1、取得 MSG 参数，并利用此参数值生成支付结果对象
PaymentResult tResult = new PaymentResult(request.getParameter("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://your.server.name/yourSuccessMerchantPage.jsp?请传入必要参数";
}
else {
  //4、支付失败
  tMerchantPage = "http://your.server.name/yourOtherMerchantPage.jsp?请传入必要参数";
}
%>
<!--
<URL><%= tMerchantPage %></URL>
-->
<HTML>
<HEAD>
<meta http-equiv="refresh" content="0; url='<%= tMerchantPage %>'">
</HEAD>
</HTML>
```

D、退货交易范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-退货</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>退货<br>
<%
//1、取得退货所需要的信息
String tOrderNo      = request.getParameter("OrderNo"  );
String tNewOrderNo      = request.getParameter("NewOrderNo"  );
String tTrxAmountStr = request.getParameter("TrxAmount");
double  tTrxAmount   = Double.parseDouble(tTrxAmountStr);

//2、生成退货请求对象
RefundRequest tRequest = new RefundRequest();
tRequest.setOrderNo  (tOrderNo  );  //订单号    （必要信息）
tRequest.setNewOrderNo  (tNewOrderNo  );  //退款订单号    （可选）
tRequest.setTrxAmount(tTrxAmount);  //退货金额  （必要信息）

//3、传送退货请求并取得退货结果
TrxResponse tResponse = tRequest.postRequest();

//4、判断退货结果状态，进行后续操作
if (tResponse.isSuccess()) {
  //5、退货成功
  out.println("TrxType   = [" + tResponse.getValue("TrxType"  ) + "]<br>");
  out.println("OrderNo   = [" + tResponse.getValue("OrderNo"  ) + "]<br>");
  out.println("NewOrderNo   = [" + tResponse.getValue("NewOrderNo"  ) + "]<br>");
```

```
    out.println("TrxAmount = [" + tResponse.getValue("TrxAmount") + "]<br>");
    out.println("BatchNo   = [" + tResponse.getValue("BatchNo"  ) + "]<br>");
    out.println("VoucherNo = [" + tResponse.getValue("VoucherNo") + "]<br>");
    out.println("HostDate  = [" + tResponse.getValue("HostDate" ) + "]<br>");
    out.println("HostTime  = [" + tResponse.getValue("HostTime" ) + "]<br>");
    out.println("TrnxNo    = [" + tResponse.getValue("iRspRef"  ) + "]<br>");
  }
  else {
    //6、退货失败
    out.println("ReturnCode   = [" + tResponse.getReturnCode  () + "]<br>");
    out.println("ErrorMessage = [" + tResponse.getErrorMessage() + "]<br>");
  }
  %>
  <a href='Merchant.html'>回商户首页</a></CENTER>
  </BODY></HTML>
```

E、订单查询交易范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<%@ page import = "java.util.ArrayList" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-商户订单查询</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>商户订单查询<br>
<%
//1、取得商户订单查询所需要的信息
String tOrderNo    = request.getParameter("OrderNo"  );
String tQueryType = request.getParameter("QueryType");
boolean tEnableDetailQuery = false;
if (tQueryType.equals("1"))
    tEnableDetailQuery = true;

//2、生成商户订单查询请求对象
QueryOrderRequest tRequest = new QueryOrderRequest();
tRequest.setOrderNo       (tOrderNo          );  //订单号            （必要信息）
tRequest.enableDetailQuery(tEnableDetailQuery);  //是否查询详细信息  （必要信息）

//3、传送商户订单查询请求并取得订单查询结果
TrxResponse tResponse = tRequest.postRequest();

//4、判断商户订单查询结果状态，进行后续操作
if (tResponse.isSuccess()) {
  //5、生成订单对象
  Order tOrder = new Order(new XMLDocument(tResponse.getValue("Order")));
  out.println("OrderNo      = [" + tOrder.getOrderNo     () + "]<br>");
```

```
    out.println("OrderAmount  = [" + tOrder.getOrderAmount () + "]<br>");
    out.println("OrderDesc    = [" + tOrder.getOrderDesc   () + "]<br>");
    out.println("OrderDate    = [" + tOrder.getOrderDate   () + "]<br>");
    out.println("OrderTime    = [" + tOrder.getOrderTime   () + "]<br>");
    out.println("OrderURL     = [" + tOrder.getOrderURL    () + "]<br>");
    out.println("PayAmount    = [" + tOrder.getPayAmount   () + "]<br>");
    out.println("RefundAmount = [" + tOrder.getRefundAmount() + "]<br>");
    out.println("OrderStatus  = [" + tOrder.getOrderStatus () + "]<br>");

    //6、取得订单明细
    ArrayList tOrderItems = tOrder.getOrderItems();
    for(int i = 0; i < tOrderItems.size(); i++) {
      OrderItem tOrderItem = (OrderItem) tOrderItems.get(i);
      out.println("ProductID   = [" + tOrderItem.getProductID  () + "]<br>");
      out.println("ProductName = [" + tOrderItem.getProductName() + "]<br>");
      out.println("UnitPrice   = [" + tOrderItem.getUnitPrice  () + "]<br>");
      out.println("Qty         = [" + tOrderItem.getQty        () + "]<br>");
    }
  }
  else {
    //7、商户订单查询失败
    out.println("ReturnCode   = [" + tResponse.getReturnCode  () + "]<br>");
    out.println("ErrorMessage = [" + tResponse.getErrorMessage() + "]<br>");
  }
  %>
  <a href='Merchant.html'>回商户首页</a></CENTER>
  </BODY></HTML>
```

F、交易对账单下载范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<%@ page import = "java.util.ArrayList" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-交易对账单下载</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>交易对账单下载<br>
<%
//1、取得商户对账单下载所需要的信息
String tSettleDate = request.getParameter("SettleDate");

//2、生成商户对账单下载请求对象
SettleRequest tRequest = new SettleRequest();
tRequest.setSettleDate(tSettleDate);                //对账日期 YYYY/MM/DD （必要信息）
tRequest.setSettleType(SettleFile.SETTLE_TYPE_TRX);//对账类型 （必要信息）
                    //SettleFile.SETTLE_TYPE_TRX: 交易对账单

//3、传送商户对账单下载请求并取得对账单
TrxResponse tResponse = tRequest.postRequest();

//4、判断商户对账单下载结果状态，进行后续操作
if (tResponse.isSuccess()) {
  //5、商户对账单下载成功，生成对账单对象
  SettleFile tSettleFile = new SettleFile(tResponse);
  out.println("SettleDate        = [" + tSettleFile.getSettleDate       () + "]<br>");
  out.println("SettleType        = [" + tSettleFile.getSettleType       () + "]<br>");
  out.println("NumOfPayments     = [" + tSettleFile.getNumOfPayments    () + "]<br>");
  out.println("SumOfPayAmount    = [" + tSettleFile.getSumOfPayAmount   () + "]<br>");
```

```
    out.println("NumOfRefunds      = [" + tSettleFile.getNumOfRefunds      () + "]<br>");
    out.println("SumOfRefundAmount = [" + tSettleFile.getSumOfRefundAmount() + "]<br>");

    //6、取得对账单明细
    String[] tRecords = tSettleFile.getDetailRecords();
    for(int i = 0; i < tRecords.length; i++) {
      out.println("Record-" + i + " = [" + tRecords[i] + "]<br>");
    }
  }
  else {
    //7、商户账单下载失败
    out.println("ReturnCode   = [" + tResponse.getReturnCode  () + "]<br>");
    out.println("ErrorMessage = [" + tResponse.getErrorMessage() + "]<br>");
  }
  %>
  <a href='Merchant.html'>回商户首页</a></CENTER>
  </BODY></HTML>
```

```
/BODY></HTML>
```

G、指定时间段交易对账单下载

```
<%@ page contentType="text/html; charset=gb2312" %>

<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>

<%@ page import = "com.hitrust.trustpay.client.*" %>

<%@ page import = "java.util.ArrayList" %>

<% response.setHeader("Cache-Control", "no-cache"); %>

<HTML>

<HEAD><TITLE>农行网上支付平台-商户接口范例-交易对账单下载</TITLE></HEAD>

<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>

<CENTER>交易对账单下载<br>

<%

//1、取得商户对账单下载所需要的信息

String tSettleDate = request.getParameter("SettleDate");

String tSettleStartHour = request.getParameter("SettleStartHour");

String tSettleEndHour = request.getParameter("SettleEndHour");


//2、生成商户对账单下载请求对象

SettleRequest tRequest = new SettleRequest();

tRequest.setSettleDate(tSettleDate);                //对账日期 YYYY/MM/DD （必要信息）

tRequest.setSettleType(SettleFile.SETTLE_TYPE_TRX);//对账类型 （必要信息）

                    //SettleFile.SETTLE_TYPE_TRX: 交易对账单

tRequest.setSettleStartHour(tSettleStartHour);          //对帐开始时间段（0-23）

tRequest.setSettleEndHour(tSettleEndHour);              //对帐截止时间段（0-23）


//3、传送商户对账单下载请求并取得对账单

TrxResponse tResponse = tRequest.postRequest();


//4、判断商户对账单下载结果状态，进行后续操作

if (tResponse.isSuccess()) {

  //5、商户对账单下载成功，生成对账单对象

  SettleFile tSettleFile = new SettleFile(tResponse);
```

```
    out.println("SettleDate        = [" + tSettleFile.getSettleDate       () + "]<br>");
    out.println("SettleType        = [" + tSettleFile.getSettleType       () + "]<br>");
    out.println("NumOfPayments     = [" + tSettleFile.getNumOfPayments    () + "]<br>");
    out.println("SumOfPayAmount    = [" + tSettleFile.getSumOfPayAmount   () + "]<br>");
    out.println("NumOfRefunds      = [" + tSettleFile.getNumOfRefunds     () + "]<br>");
    out.println("SumOfRefundAmount = [" + tSettleFile.getSumOfRefundAmount() + "]<br>");

    //6、取得对账单明细
    String[] tRecords = tSettleFile.getDetailRecords();
    for(int i = 0; i < tRecords.length; i++) {
      out.println("Record-" + i + " = [" + tRecords[i] + "]<br>");
    }
  }
  else {
    //7、商户账单下载失败
    out.println("ReturnCode   = [" + tResponse.getReturnCode  () + "]<br>");
    out.println("ErrorMessage = [" + tResponse.getErrorMessage() + "]<br>");
  }
  %>
  <a href='Merchant.html'>回商户首页</a></CENTER>
  </BODY></HTML>
```

H、卡验证请求范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<%
//1、取得支付请求所需要的信息
String tResultNotifyURL = request.getParameter("ResultNotifyURL");
String tCertificateType = request.getParameter("CertificateType");
String tCertificateNo   = request.getParameter("CertificateNo"  );
```

```
//2、生成卡身份验证请求对象
CardVerifyRequest tCardVerifyRequest = new CardVerifyRequest();
tCardVerifyRequest.setResultNotifyURL(tResultNotifyURL); //设定支付结果回传网址 （必要信息）
tCardVerifyRequest.setCertificateType(tCertificateType);//设定证件类型
tCardVerifyRequest.setCertificateNo(tCertificateNo);//设定证件号码

//5、传送卡身份验证请求并取得支付网址
TrxResponse tTrxResponse = tCardVerifyRequest.postRequest();
if (tTrxResponse.isSuccess()) {
   //6、卡身份验证请求提交成功，将客户端导向支付页面
   response.sendRedirect(tTrxResponse.getValue("VerifyURL"));
}
else {
   //7、卡身份验证请求提交失败，商户自定后续动作
%>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-支付请求</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>支付请求<br>
<%
  out.println("ReturnCode   = [" + tTrxResponse.getReturnCode   () + "]<br>");
  out.println("ErrorMessage = [" + tTrxResponse.getErrorMessage() + "]<br>");
}
%>
<a href='Merchant.html'>回商户首页</a></CENTER>
</BODY></HTML>
```

I、卡身份验证结果接收范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<HTML>
```

```
<HEAD><TITLE>农行网上支付平台-商户接口范例-支付结果接收</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>支付结果<br>
<%
//1、取得 MSG 参数，并利用此参数值生成支付结果对象
PaymentResult tResult = new PaymentResult(request.getParameter("MSG"));

//2、判断支付结果状态，进行后续操作
if (tResult.isSuccess()) {
  //3、支付成功
    out.println("卡验证成功”);
}
else {
  //4、支付失败
  out.println("ReturnCode   = [" + tResult.getReturnCode  () + "]<br>");
  out.println("ErrorMessage = [" + tResult.getErrorMessage() + "]<br>");
}
%>
<a href='Merchant.html'>回商户首页</a></CENTER>
</BODY></HTML>
```

J、身份验证请求范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-身份验证请求</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>身份验证请求<br>
<%
//1、取得身份验证请求所需要的信息
```

```
String tCertificateType  = request.getParameter("CertificateType");
String tCertificateNo  = request.getParameter("CertificateNo");
String tBankCardNo = request.getParameter("BankCardNo");
String tResultNotifyURL  = request.getParameter("ResultNotifyURL");
String tRequestDate       = request.getParameter("RequestDate"      );
String tRequestTime       = request.getParameter("RequestTime"      );
//2、生成身份验证请求对象
IdentityVerifyRequest tRequest = new IdentityVerifyRequest();
tRequest.setBankCardNo(tBankCardNo);       //银行帐号         （必要信息）
tRequest.setCertificateNo(tCertificateNo);    //证件号码         （必要信息）
tRequest.setCertificateType(tCertificateType); //证件类型         （必要信息）
tRequest.setResultNotifyURL(tResultNotifyURL); //身份验证回传网址（必要信息）
tRequest.setOrderDate  (tRequestDate  ); //验证请求日期 （必要信息 - YYYY/MM/DD）
tRequest.setOrderTime  (tRequestTime  ); //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
TrxResponse tTrxResponse = tRequest.postRequest();
if (tResult.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.sendRedirect(tTrxResponse.getValue("VerifyURL"));}
else {
//5、身份验证请求提交失败，商户自定后续动作
  out.println("ReturnCode   = [" + tResult.getReturnCode  () + "]<br>");
  out.println("ErrorMessage = [" + tResult.getErrorMessage() + "]<br>");
}
%>
<a href='Merchant.html'>回商户首页</a></CENTER>
</BODY></HTML>
```

K、身份验证结果接收范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
```

```
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-身份验证结果接收</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>身份验证结果<br>
<%
//1、取得 MSG 参数，并利用此参数值生成验证结果对象
PaymentResult tResult = new PaymentResult(request.getParameter("MSG"));
//2、判断验证结果状态，进行后续操作
if (tResult.isSuccess()) {
//3、验证成功
  out.println("户名          = [" + tResult.getValue("AccountName"        ) + "]<br>");
}
else {
//4、验证失败
  out.println("ReturnCode    = [" + tResult.getReturnCode  () + "]<br>");
  out.println("ErrorMessage = [" + tResult.getErrorMessage() + "]<br>");
}
%>
<a href='Merchant.html'>回商户首页</a></CENTER>
</BODY></HTML>
```

L、批量退款发送请求范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<%@ page import = "java.util.ArrayList"%>
<% response.setHeader("Cache-Control", "no-cache"); %>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-批量退款</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>批量退款<br>
<%
//1、取得批量退款所需要的信息
```

```
String tTotalCount = request.getParameter("TotalCount");
String tTotalAmount = request.getParameter("TotalAmount");
String tRemark = request.getParameter("Remark");



String orderno_arr[] = null ;
String orderamount_arr[] = null ;
int iTotalCount    = Integer.parseInt(tTotalCount);
double  iTotalAmount   = Double.parseDouble(tTotalAmount);


System.out.println("TotalCount="+iTotalCount);
System.out.println("iTotalAmount="+iTotalAmount);
System.out.println("tRemark="+tRemark);


if(iTotalCount == 1) {
   String orderno = request.getParameter("orderno") ;
   String orderamount = request.getParameter("orderamount") ;
   orderno_arr = new String[]{orderno} ;
   orderamount_arr = new String[]{orderamount} ;
}
else
{
   orderno_arr = request.getParameterValues("orderno") ;
   orderamount_arr = request.getParameterValues("orderamount") ;
 }
  ArrayList tOrderList = new ArrayList() ;
  for(int i=0;i<orderno_arr.length;i++)
  {
     String [] torder = new String[2];
     torder[0] = orderno_arr[i];
     System.out.println("orderno_arr["+i+"]="+orderno_arr[i]);
     torder[1] = orderamount_arr[i];
      System.out.println("orderamount_arr["+i+"]="+orderamount_arr[i]);
```

```
        tOrderList.add(torder);


    }
 //2、生成批量退款请求对象
 OverdueRefundRequest tOverdueRequest = new OverdueRefundRequest();
 tOverdueRequest.setTotalCount  (iTotalCount  );  //总笔数  （必要信息）
 tOverdueRequest.setTotalAmount(iTotalAmount);  //总金额 （必要信息）
 tOverdueRequest.setRemark(tRemark);//备注
 tOverdueRequest.setOrderDital(tOrderList);


//3、传送批量退款请求并取得结果
 TrxResponse tResponse = tOverdueRequest.postRequest();


//4、判断批量退款结果状态，进行后续操作
if (tResponse.isSuccess()) {


//5、批量退款成功
  out.println("TrxType   = [" + tResponse.getValue("TrxType"  ) + "]<br>");
  out.println("TotalCount  = [" + tResponse.getValue("TotalCount"  ) + "]<br>");
  out.println("TotalAmount = [" + tResponse.getValue("TotalAmount") + "]<br>");
  out.println("SerialNumber   = [" + tResponse.getValue("SerialNumber"  ) + "]<br>");
  out.println("HostDate  = [" + tResponse.getValue("HostDate" ) + "]<br>");
  out.println("HostTime  = [" + tResponse.getValue("HostTime" ) + "]<br>");
  out.println("ResultMessage   = [" + tResponse.getErrorMessage() + "]<br>");
}
else {
  //6 批量退款失败
  out.println("ReturnCode   = [" + tResponse.getReturnCode  () + "]<br>");
  out.println("ErrorMessage = [" + tResponse.getErrorMessage() + "]<br>");
}
%>
<a href='Merchant.html'>回商户首页</a></CENTER>
```

```
    </BODY></HTML>
```

M、批量退款交易结果查询请求范例

```
    <%@ page contentType="text/html; charset=gb2312" %>

    <%@ page import = "com.hitrust.trustpay.client.b2c.*" %>

    <%@ page import = "com.hitrust.trustpay.client.*" %>

    <%@ page import = "java.util.ArrayList" %>

    <% response.setHeader("Cache-Control", "no-cache"); %>

    <HTML>

    <HEAD><TITLE>农行网上支付平台-商户接口范例-批量退款结果查询</TITLE></HEAD>

    <BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>

    <CENTER>批量退款结果查询<br>

    <%

    //1、取得批量退款结果查询请求所需要的信息

    String tSerialNumber         = request.getParameter("SerialNumber");


    //2、生成批量退款结果查询请求对象

    QueryOverdueRefundRequest tQueryBatchRequest = new QueryOverdueRefundRequest();

    tQueryBatchRequest.setSerialNumber      (tSerialNumber); //设定批量退款结果查询请求的流水号（必要信息）


    //3、传送批量退款结果查询请求并取得结果

    TrxResponse tResponse = tQueryBatchRequest.postRequest();

    //4、判断批量退款结果查询状态，进行后续操作

    if (tResponse.isSuccess()) {

     //5、生成批量对象

      OverdueBatch tBatch = new OverdueBatch(new XMLDocument(tResponse.getValue("QueryOverdueRefund")));

      out.println("SerialNumber  = [" + tBatch.getSerialNumber      () + "]<br>");

      out.println("RefundAmount  = [" + tBatch.getRefundAmount () + "]<br>");

      out.println("RefundCount   = [" + tBatch.getRefundCount    () + "]<br>");

      out.println("BatchStatus   = [" + tBatch.getStatus    () + "]<br>");


      //6、取得订单明细
```

```
    ArrayList tOrders = tBatch.getOrder();
    for(int i = 0; i < tOrders.size(); i++) {
      Order tOrder = (Order) tOrders.get(i);
      out.println("OrderNo    = [" + tOrder.getOrderNo  () + "]<br>");
      out.println("RefundAmount = [" + tOrder.getRefundAmount() + "]<br>");
      out.println("OrderStatus  = [" + tOrder.getOrderStatus () + "]<br>");
      out.println("OrderDesc  = [" + tOrder.getOrderDesc () + "]<br>");
    }
  }
  else {
    //7、批量退款结果查询失败
    out.println("ReturnCode    = [" + tResponse.getReturnCode  () + "]<br>");
    out.println("ErrorMessage = [" + tResponse.getErrorMessage() + "]<br>");
  }
  %>
  <a href='Merchant.html'>回商户首页</a></CENTER>
  </BODY></HTML>
```

N、委托扣款签约交易请求范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-委托扣款签约请求</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>委托扣款签约请求<br>
<%
//1、取得委托扣款签约请求所需要的信息
String tOrderNo          = request.getParameter("OrderNo");
String tCertificateType          = request.getParameter("CertificateType");
String tCertificateNo          = request.getParameter("CertificateNo");
String tResultNotifyURL          = request.getParameter("ResultNotifyURL");
```

```
String tRequestDate          = request.getParameter("RequestDate");
String tRequestTime          = request.getParameter("RequestTime");
String tNotifyType          = request.getParameter("NotifyType");

//2、生成委托扣款签约请求对象
B2CAgentSignContractRequest tRequest = new B2CAgentSignContractRequest();
tRequest.setIOrderNo(tOrderNo);//订单编号（必要信息）
tRequest.setICertificateNo(tCertificateNo);    //证件号码          （必要信息）
tRequest.setICertificateType(tCertificateType); //证件类型          （必要信息）
tRequest.setIResultNotifyURL(tResultNotifyURL); //身份验证回传网址（必要信息）
tRequest.setIRequestDate   (tRequestDate   ); //验证请求日期 （必要信息 - YYYY/MM/DD）
tRequest.setIRequestTime   (tRequestTime   ); //验证请求时间 （必要信息 - HH:MM:SS）
tRequest.setINotifyType(tNotifyType);

//3、传送委托扣款签约请求并取得签约网址
TrxResponse tTrxResponse= tRequest.postRequest();
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.sendRedirect(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
  out.println("ReturnCode   = [" + tTrxResponse.getReturnCode   () + "]<br>");
  out.println("ErrorMessage = [" + tTrxResponse.getErrorMessage() + "]<br>");
}
%>
<a href='Merchant.html'>回商户首页</a></CENTER>
</BODY></HTML>
```

0、委托扣款解约交易请求范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
```

```
<% response.setHeader("Cache-Control", "no-cache"); %>

<HTML>

<HEAD><TITLE>农行网上支付平台-商户接口范例-委托扣款解约请求</TITLE></HEAD>

<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>

<CENTER>委托扣款解约请求<br>

<%

//1、取得委托扣款解约请求所需要的信息

String tOrderNo = request.getParameter("OrderNo");

String tCertificateType  = request.getParameter("CertificateType");

String tCertificateNo  = request.getParameter("CertificateNo");

String tResultNotifyURL  = request.getParameter("ResultNotifyURL");

String tRequestDate       = request.getParameter("RequestDate");

String tRequestTime       = request.getParameter("RequestTime");

String tNotifyType           = request.getParameter("NotifyType");

String tAgentSignNo          = request.getParameter("AgentSignNo");


//2、生成委托扣款解约请求对象

B2CAgentUnsignContractRequest tRequest = new B2CAgentUnsignContractRequest();

tRequest.setIOrderNo(tOrderNo);//订单编号（必要信息）

tRequest.setICertificateNo(tCertificateNo);    //证件号码        （必要信息）

tRequest.setICertificateType(tCertificateType); //证件类型        （必要信息）

tRequest.setIResultNotifyURL(tResultNotifyURL); //身份验证回传网址（必要信息）

tRequest.setIRequestDate  (tRequestDate  ); //验证请求日期 （必要信息 - YYYY/MM/DD）

tRequest.setIRequestTime  (tRequestTime  ); //验证请求时间 （必要信息 - HH:MM:SS）

tRequest.setINotifyType(tNotifyType);

tRequest.setIAgentSignNo(tAgentSignNo);


//3、传送委托扣款解约请求并取得解约网址

TrxResponse tTrxResponse = tRequest.postRequest();

if (tTrxResponse.isSuccess()) {

//4、委托扣款解约请求提交成功，将客户端导向解约页面

   response.sendRedirect(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
  out.println("ReturnCode   = [" + tTrxResponse.getReturnCode  () + "]<br>");
  out.println("ErrorMessage = [" + tTrxResponse.getErrorMessage() + "]<br>");
}
%>
<a href='Merchant.html'>回商户首页</a></CENTER>
</BODY></HTML>
```

P、委托扣款单笔代扣交易请求范例

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-委托扣款单笔代扣请求</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>委托扣款单笔代扣请求<br>
<%
//1、取得委托扣款单笔代扣请求所需要的信息
String tOrderNo = request.getParameter("OrderNo");
String tRequestDate       = request.getParameter("RequestDate");
String tRequestTime       = request.getParameter("RequestTime");
String tCurrency          = request.getParameter("Currency");
String tAmount            = request.getParameter("Amount");
String tProductId         = request.getParameter("ProductId");
String tProductName       = request.getParameter("ProductName");
String tQuantity          = request.getParameter("Quantity");
String tCertificateNo     = request.getParameter("CertificateNo");
String tAgentSignNo         = request.getParameter("AgentSignNo");

//2、生成委托扣款单笔代扣请求对象
```

```
B2CAgentPaymentRequest tRequest = new B2CAgentPaymentRequest();
tRequest.setIOrderNo(tOrderNo);//订单编号（必要信息）
tRequest.setIRequestDate  (tRequestDate  ); //验证请求日期 （必要信息 - YYYY/MM/DD）
tRequest.setIRequestTime  (tRequestTime  ); //验证请求时间 （必要信息 - HH:MM:SS）
tRequest.setICurrency(tCurrency);
tRequest.setIAmount(tAmount);
tRequest.setIProductId(tProductId);
tRequest.setIProductName(tProductName);
tRequest.setIQuantity(tQuantity);
tRequest.setICertificateNo(tCertificateNo);
tRequest.setIAgentSignNo(tAgentSignNo);

//3、传送委托扣款单笔代扣请求
TrxResponse tTrxResponse = tRequest.postRequest();
if (tTrxResponse.isSuccess()) {
//4、委托扣款单笔代扣请求提交成功
   out.println("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
  out.println("ReturnCode   = [" + tTrxResponse.getReturnCode  () + "]<br>");
  out.println("ErrorMessage = [" + tTrxResponse.getErrorMessage() + "]<br>");
}
%>
<a href='Merchant.html'>回商户首页</a></CENTER>
</BODY></HTML>
```

Q、委托扣款批量交易请求范例

```
<%@ page contentType="text/html; charset=gb2312"%>
<%@ page import="java.math.BigDecimal"%>
<%@ page import="com.hitrust.trustpay.client.b2c.*"%>
<%@ page import="com.hitrust.trustpay.client.*"%>
<%
```

```
        request.setCharacterEncoding("GBK");
        response.setHeader("Cache-Control", "no-cache");
    %>
    <HTML>
    <HEAD><TITLE>农行网上支付平台-商户接口范例-委托扣款批量</TITLE></HEAD>
    <BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
    <CENTER>委托扣款批量<br>
    <%
    //1、取得委托扣款批量需要的信息
        String batchNo = request.getParameter("BatchNo");
        String batchDate = request.getParameter("BatchDate");
        String agentCount = request.getParameter("AgentCount");
        String agentAmount = request.getParameter("AgentAmount");
        String orderno_arr[] = null;
        String orderamount_arr[] = null;
        String certificateno_arr[] = null;
        String contractid_arr[] = null;
        String productid_arr[] = null;
        String productname_arr[] = null;
        String productnum_arr[] = null;

        int iBatchSize = Integer.parseInt(agentCount);
        if (iBatchSize == 1) {
                String orderno = request.getParameter("orderno");
                String orderamount = request.getParameter("orderamount");
                String certificateno = request.getParameter("certificateno");
                String contractid = request.getParameter("contractid");
                String productid = request.getParameter("productid");
                String productname = request.getParameter("productname");
                String productnum = request.getParameter("productnum");

                orderno_arr = new String[] { orderno };
                orderamount_arr = new String[] { orderamount };
```

```
            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.getParameterValues("orderno");
            orderamount_arr = request.getParameterValues("orderamount");
            certificateno_arr = request.getParameterValues("certificateno");
            contractid_arr = request.getParameterValues("contractid");
            productid_arr = request.getParameterValues("productid");
            productname_arr = request.getParameterValues("productname");
            productnum_arr = request.getParameterValues("productnum");

    }

//2、生成委托扣款批量请求对象
    AgentBatch iAgentBatch = new AgentBatch();
    iAgentBatch.setBatchNo(batchNo);
    iAgentBatch.setBatchDate(batchDate);
    iAgentBatch.setAgentAmount(new Double(agentAmount).doubleValue());
    iAgentBatch.setAgentCount(Integer.parseInt(agentCount));

    B2CAgentBatchRequest aB2CAgentBatchRequest = new B2CAgentBatchRequest();
    aB2CAgentBatchRequest.setAgentBatch(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++) {

            AgentBatchDetail aAgentBatchDetail = new AgentBatchDetail();
            aAgentBatchDetail.setOrderNo(orderno_arr[i]);
            aAgentBatchDetail.setOrderAmount(new Double(orderamount_arr[i]).doubleValue());
            aAgentBatchDetail.setCertificateNo(certificateno_arr[i]);
```

```
            aAgentBatchDetail.setContractID(contractid_arr[i]);
            aAgentBatchDetail.setProductID(productid_arr[i]);
            aAgentBatchDetail.setProductName(productname_arr[i]);
            aAgentBatchDetail.setProductNum(new Integer(productnum_arr[i]).intValue());



            aB2CAgentBatchRequest.addAgentBatchDetail(aAgentBatchDetail);
    }
//3、传送委托扣款批量请求
    TrxResponse tResponse = aB2CAgentBatchRequest.postRequest();

    if (tResponse.isSuccess()) {
//4、委托扣款批量请求提交成功
            out.print("批量受理成功!");
    } else {
//5、委托扣款批量请求提交失败，商户自定后续动作
            out.println("ReturnCode   = [" + tResponse.getReturnCode() + "]<br>");
            out.println("ErrorMessage = [" + tResponse.getErrorMessage() + "]<br>");
    }%>
<a href='Merchant.html'>回商户首页</a></CENTER>
</BODY></HTML>
```

R、委托扣款批量结果查询交易请求范例

```
<%@ page contentType="text/html; charset=gb2312"%>
<%@ page import="java.math.BigDecimal"%>
<%@ page import="com.hitrust.trustpay.client.b2c.*"%>
<%@ page import="com.hitrust.trustpay.client.*"%>
<%
    request.setCharacterEncoding("GBK");
    response.setHeader("Cache-Control", "no-cache");
%>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-委托扣款批量处理结果</TITLE></HEAD>
```

```
    <BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>

    <CENTER>委托扣款批量处理结果<br>

    <%

    //1、取得委托扣款批量结果查询需要的信息

        String tBatchNo = request.getParameter("BatchNo");

        String tBatchDate = request.getParameter("BatchDate");


    //2、生成委托扣款批量结果查询请求对象

        B2CAgentBatchQueryRequest tRequest = new B2CAgentBatchQueryRequest();

        tRequest.setBatchNo(tBatchNo); //请求批次号         (必要信息)

        tRequest.setBatchDate(tBatchDate); //请求日期       YYYY/MM/DD


    //3、传送委托扣款批量结果查询请求并取得结果

        TrxResponse tResponse = tRequest.postRequest();


        if (tResponse.isSuccess()) {

    //4、委托扣款批量结果查询请求提交成功

        AgentBatch tAgentBatch = new AgentBatch(new
XMLDocument(tResponse.getValue("AgentBatch")));

        out.println("BatchNo  = [" + tAgentBatch.getBatchNo() + "]<br>");

        out.println("BatchDate    = [" + tAgentBatch.getBatchDate() + "]<br>");

        out.println("BatchTime    = [" + tAgentBatch.getBatchTime() + "]<br>");

        out.println("AgentAmount  = [" + tAgentBatch.getAgentAmount() + "]<br>");

        out.println("AgentCount    = [" + tAgentBatch.getAgentCount() + "]<br>");

        out.println("BatchStatus    = [" + tAgentBatch.getBatchStatus() + "]<br>");

        out.println("BatchStatusZH    = [" +
        tAgentBatch.getBatchSatusChinese(tAgentBatch.getBatchStatus()) + "]<br>");

        //6、取得订单明细

        ArrayList tAgentBatchDetails = tAgentBatch.getAgentBatchDetail();

        for (int i = 0; i < tAgentBatchDetails.size(); i++) {

                AgentBatchDetail tAgentBatchDetail = (AgentBatchDetail) tAgentBatchDetails.get(i);

                out.println("SeqNo    = [" + (i + 1) + "],");

                out.println("OrderNo    = [" + tAgentBatchDetail.getOrderNo() + "],");
```

```
            out.println("OrderAmount = [" + tAgentBatchDetail.getOrderAmount() + "],");

            out.println("CertificateNo  = [" + tAgentBatchDetail.getCertificateNo() + "],");

            out.println("ContractID = [" + tAgentBatchDetail.getContractID() + "],");

            out.println("ProductID   = [" + tAgentBatchDetail.getProductID() + "],");

            out.println("ProductName = [" + tAgentBatchDetail.getProductName() + "],");

            out.println("ProductNum  = [" + tAgentBatchDetail.getProductNum() + "],");

            out.println("OrderStatus  = [" + tAgentBatchDetail.getOrderStatus() + "],");

            out.println("OrderStatusZH  = [" +
    tAgentBatchDetail.getStatusChinese(tAgentBatchDetail.getOrderStatus()) + "]<br>");

    }

    } else {

//5、委托扣款批量结果查询请求提交失败，商户自定后续动作

            out.println("ReturnCode   = [" + tResponse.getReturnCode() + "]<br>");

            out.println("ErrorMessage = [" + tResponse.getErrorMessage() + "]<br>");

    }%>

<a href='Merchant.html'>回商户首页</a></CENTER>

</BODY></HTML>
```

S、贷记卡对账单下载交易请求范例

```
<%@ page contentType="text/html; charset=gb2312" %>

<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>

<%@ page import = "com.hitrust.trustpay.client.*" %>

<%@ page import = "java.util.ArrayList" %>

<% response.setHeader("Cache-Control", "no-cache"); %>

<HTML>

<HEAD><TITLE>农行网上支付平台-商户接口范例-交易对账单下载</TITLE></HEAD>

<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>

<CENTER>交易对账单下载<br>

<%

//1、取得商户对账单下载所需要的信息

String tSettleDate = request.getParameter("SettleDate");


//2、生成商户对账单下载请求对象
```

```
SettleRequest tRequest = new SettleRequest();
tRequest.setSettleDate(tSettleDate);                    //对账日期 YYYY/MM/DD （必要信息）
tRequest.setSettleType(SettleFile.SETTLE_TYPE_CREDIT_TRX);//对账类型 （必要信息）
                    //SettleFile.SETTLE_TYPE_TRX: 交易对账单

//3、传送商户对账单下载请求并取得对账单
TrxResponse tResponse = tRequest.postRequest();

//4、判断商户对账单下载结果状态，进行后续操作
if (tResponse.isSuccess()) {
  //5、商户对账单下载成功，生成对账单对象
  SettleFile tSettleFile = new SettleFile(tResponse);
  out.println("SettleDate        = [" + tSettleFile.getSettleDate       () + "]<br>");
  out.println("SettleType        = [" + tSettleFile.getSettleType       () + "]<br>");
  out.println("NumOfPayments     = [" + tSettleFile.getNumOfPayments    () + "]<br>");
  out.println("SumOfPayAmount    = [" + tSettleFile.getSumOfPayAmount   () + "]<br>");
  out.println("NumOfRefunds      = [" + tSettleFile.getNumOfRefunds     () + "]<br>");
  out.println("SumOfRefundAmount = [" + tSettleFile.getSumOfRefundAmount() + "]<br>");

  //6、取得对账单明细
  String[] tRecords = tSettleFile.getDetailRecords();
  for(int i = 0; i < tRecords.length; i++) {
    out.println("Record-" + i + " = [" + tRecords[i] + "]<br>");
  }
}
else {
  //7、商户账单下载失败
  out.println("ReturnCode   = [" + tResponse.getReturnCode  () + "]<br>");
  out.println("ErrorMessage = [" + tResponse.getErrorMessage() + "]<br>");
}
%>
<a href='Merchant.html'>回商户首页</a></CENTER>
```

```
</BODY></HTML>
```

T、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

```
<%@ page pageEncoding="gb2312" %>

<%@ page contentType="text/html; charset=gb2312" %>

<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>

<%@ page import = "com.hitrust.trustpay.client.*" %>

<% response.setHeader("Cache-Control", "no-cache"); %>

<% request.setCharacterEncoding("gb2312"); %>

<%

//1、取得支付请求所需要的信息

String tOrderNo         = request.getParameter("OrderNo"        );

String tOrderDesc       = request.getParameter("OrderDesc"      );

String tOrderDate       = request.getParameter("OrderDate"      );

String tOrderTime       = request.getParameter("OrderTime"      );

String tOrderAmountStr  = request.getParameter("OrderAmount"    );

String tOrderURL        = request.getParameter("OrderURL"       );

String tProductType     = request.getParameter("ProductType"    );

String tPaymentType     = request.getParameter("PaymentType"    );

String tNotifyType      = request.getParameter("NotifyType"     );

String tResultNotifyURL = request.getParameter("ResultNotifyURL");

String tMerchantRemarks = request.getParameter("MerchantRemarks");

double  tOrderAmount    = Double.parseDouble(tOrderAmountStr);

String tPaymentLinkType = request.getParameter("PaymentLinkType");

String tSignature="";

//2、生成订单对象
```

```
    Order tOrder = new Order();

    tOrder.setOrderNo     (tOrderNo     ); //设定订单编号 （必要信息）

    tOrder.setOrderDesc   (tOrderDesc   ); //设定订单说明

    tOrder.setOrderDate   (tOrderDate   ); //设定订单日期 （必要信息 - YYYY/MM/DD）

    tOrder.setOrderTime   (tOrderTime   ); //设定订单时间 （必要信息 - HH:MM:SS）

    tOrder.setOrderAmount(tOrderAmount); //设定订单金额 （必要信息）

    tOrder.setOrderURL    (tOrderURL    ); //设定订单网址



    //3、生成定单订单对象，并将订单明细加入定单中（可选信息）

    tOrder.addOrderItem(new OrderItem("IP000001", "中国移动 IP 卡", 100.00f, 1));

    tOrder.addOrderItem(new OrderItem("IP000002", "网通 IP 卡"     ,  90.00f, 2));

    MerchantConfig tMerchantConfig=MerchantConfig.getUniqueInstance();;
    String sTrustPayIETrxURL=tMerchantConfig.getTrustPayIETrxURL();
    String sErrorUrl=tMerchantConfig.getMerchantErrorURL();



    //4、生成支付请求对象

    PaymentRequest tPaymentRequest = new PaymentRequest();

    tPaymentRequest.setOrder       (tOrder       ); //设定支付请求的订单 （必要信息）

    tPaymentRequest.setProductType(tProductType); //设定商品种类 （必要信息）

                                                  //PaymentRequest.PRD_TYPE_ONE：非实体商品，如服务、
IP 卡、下载 MP3、...

                                                  //PaymentRequest.PRD_TYPE_TWO：实体商品

    tPaymentRequest.setPaymentType(tPaymentType); //设定支付类型

                                                  //PaymentRequest.PAY_TYPE_ABC：农行卡支付

                                                  //PaymentRequest.PAY_TYPE_INT：国际卡支付

                                                  // PaymentRequest.PAY_TYPE_CREDIT：贷记卡支付
```

```
    tPaymentRequest.setNotifyType(tNotifyType);    //设定商户通知方式

                                                   //0：URL 页面通知

                                                   //1：服务器通知

    tPaymentRequest.setResultNotifyURL(tResultNotifyURL); //设定支付结果回传网址 （必要信息）

    tPaymentRequest.setMerchantRemarks(tMerchantRemarks); //设定商户备注信息

    tPaymentRequest.setPaymentLinkType(tPaymentLinkType);//设定支付接入方式

    //5、生成带商户签名的报文,它是表单提交时的一个参数，在此过程中会进行客户端验证，如订单号是否为空，长度是否符合
    //规定等，在此过程中产生的错误会在 ErrorPageInternal.jsp 页面进行展示

    try{

     tSignature = tPaymentRequest.genSignature(1);

    }catch (TrxException e){

            request.setAttribute("tReturnCode", e.getCode());

            request.setAttribute("tErrorMsg", e.getMessage());

            request.getRequestDispatcher("/ErrorPageInternal.jsp").forward(request, response);

            return;

    }

    %>

    <HTML>

    <HEAD><TITLE>农行网上支付平台-商户接口范例-支付请求</TITLE></HEAD>

<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>

<CENTER>支付请求<br>

<form name="form1" method="post" action="<%= sTrustPayIETrxURL%>">

<input type="hidden" name="Signature" value="<%=tSignature%>">

<input type="hidden" name="errorPage" value="<%= sErrorUrl%>">

<TR><TD colspan=2><INPUT type=submit value="提交">

</form>

//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
```

U、商户本地返回错误页面范例

```
<%@ page pageEncoding="gb2312" %>
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<% request.setCharacterEncoding("gb2312"); %>
<%
String tReturnCode=(String)request.getAttribute("tReturnCode");
String tErrorMsg=(String)request.getAttribute("tErrorMsg");
%>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-支付请求</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>支付请求<br>
<%
  out.println("ReturnCode   = [" +tReturnCode+ "]<br>");
  out.println("ErrorMessage = [" + tErrorMsg+ "]<br>");
%>
<a href='MerchantPaymentIE.html'>回商户首页</a></CENTER>
</BODY></HTML>
```

V、支付平台返回错误页面范例

```
<%@ page pageEncoding="gb2312" %>
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<% request.setCharacterEncoding("gb2312"); %>
<%
String tReturnCode=request.getParameter("ReturnCode");
String tErrorMsg=request.getParameter("ErrorMessage");
%>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-支付请求</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>支付请求<br>
<%
  out.println("ReturnCode   = [" +tReturnCode+ "]<br>");
  out.println("ErrorMessage = [" + tErrorMsg+ "]<br>");
%>
<a href='MerchantPaymentIE.html'>回商户首页</a></CENTER>
</BODY></HTML>
```

W、保险直销支付请求范例

```
<%@ page pageEncoding="gb2312" %>
```

```
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<% request.setCharacterEncoding("gb2312"); %>
<%
//1、取得支付请求所需要的信息
String tOrderNo         = request.getParameter("OrderNo"        );
String tOrderDesc       = request.getParameter("OrderDesc"      );
String tOrderDate       = request.getParameter("OrderDate"      );
String tOrderTime       = request.getParameter("OrderTime"      );
String tOrderAmountStr  = request.getParameter("OrderAmount"    );
String tOrderURL        = request.getParameter("OrderURL"       );
String tProductType     = request.getParameter("ProductType"    );
String tPaymentType     = request.getParameter("PaymentType"    );
String tNotifyType      = request.getParameter("NotifyType"     );
String tResultNotifyURL = request.getParameter("ResultNotifyURL");
String tMerchantRemarks = request.getParameter("MerchantRemarks");
double  tOrderAmount    = Double.parseDouble(tOrderAmountStr);
String tPaymentLinkType = request.getParameter("PaymentLinkType");
String tBuyIP = request.getParameter("BuyIP");

//Insure
String tInsureType           = request.getParameter("InsureType"       );
String tFurl                 = request.getParameter("Furl"             );
String tOrderCategory        = request.getParameter("OrderCategory"    );
String tOrderCode            = request.getParameter("OrderCode"        );
String tOrderMode            = request.getParameter("OrderMode"        );
String tOrderName            = request.getParameter("OrderName"        );
String tInsureOrderAmountStr = request.getParameter("InsureOrderAmount");
double tInsureOrderAmount    = Double.parseDouble(tInsureOrderAmountStr);
String tUserCardNo           = request.getParameter("UserCardNo"       );
String tUserCertificateNo    = request.getParameter("CertificateNo"    );
String tUserCertificateType  = request.getParameter("CertificateType"  );
String tUserName             = request.getParameter("UserName"         );

String tSignature="";
//2、生成订单对象
Order tOrder = new Order();
tOrder.setOrderNo    (tOrderNo    ); //设定订单编号 （必要信息）
tOrder.setOrderDesc  (tOrderDesc  ); //设定订单说明
tOrder.setOrderDate  (tOrderDate  ); //设定订单日期 （必要信息 - YYYY/MM/DD）
tOrder.setOrderTime  (tOrderTime  ); //设定订单时间 （必要信息 - HH:MM:SS）
tOrder.setOrderAmount(tOrderAmount); //设定订单金额 （必要信息）
```

```
tOrder.setOrderURL    (tOrderURL    ); //设定订单网址
tOrder.setBuyIP    (tBuyIP    );        //设定订单IP
//3、生成定单订单对象，并将订单明细加入定单中（可选信息）
tOrder.addOrderItem(new OrderItem("IP000001", "中国移动IP卡", 100.00f, 1));
tOrder.addOrderItem(new OrderItem("IP000002", "网通IP卡"     ,  90.00f, 2));

MerchantConfig tMerchantConfig=MerchantConfig.getUniqueInstance();;
String sTrustPayIETrxURL=tMerchantConfig.getTrustPayIETrxURL();
String sErrorUrl=tMerchantConfig.getMerchantErrorURL();

//4、生成支付请求对象
PaymentRequest tPaymentRequest = new PaymentRequest();
tPaymentRequest.setOrder       (tOrder       ); //设定支付请求的订单 （必要信息）
tPaymentRequest.setProductType(tProductType); //设定商品种类 （必要信息）
                                              //PaymentRequest.PRD_TYPE_ONE：非实体商品，如服务、IP
卡、下载MP3、...
                                              //PaymentRequest.PRD_TYPE_TWO：实体商品
tPaymentRequest.setPaymentType(tPaymentType); //设定支付类型
                                              //PaymentRequest.PAY_TYPE_ABC：农行卡支付
                                              //PaymentRequest.PAY_TYPE_INT：国际卡支付
tPaymentRequest.setNotifyType(tNotifyType);   //设定商户通知方式
                                              //0：URL页面通知
                                              //1：服务器通知
tPaymentRequest.setResultNotifyURL(tResultNotifyURL); //设定支付结果回传网址 （必要信息）
tPaymentRequest.setMerchantRemarks(tMerchantRemarks); //设定商户备注信息
tPaymentRequest.setPaymentLinkType(tPaymentLinkType);//设定支付接入方式



        //保险对象
        Insure tInsure = new Insure();
        tInsure.setType(tInsureType);
        if (tInsure.getType().equals(Insure.INSURE_TYPE_FINANCING))
            tInsure.setFurl(tFurl);

        InsureOrder tInsureOrder = new InsureOrder();

        InsureOrderItem tInsureOrderItem = new InsureOrderItem();
        tInsureOrderItem.setCategory(tOrderCategory);
        tInsureOrderItem.setCode(tOrderCode);
        tInsureOrderItem.setMode(tOrderMode);
        tInsureOrderItem.setName(tOrderName);
        tInsureOrderItem.setAmount(tInsureOrderAmount);
        tInsureOrder.addOrderItem(tInsureOrderItem);
```

```
        tInsure.setOrder(tInsureOrder);

        if (tInsureType.equals(Insure.INSURE_TYPE_FINANCING) ||
tInsureType.equals(Insure.INSURE_TYPE_APPOINTED))
        {
            InsureUser tInsureUser = new InsureUser();
            tInsureUser.setCardNo(tUserCardNo);
            tInsureUser.setCertificateNo(tUserCertificateNo);
            tInsureUser.setCertificateType(tUserCertificateType);
            tInsureUser.setName(tUserName);

            tInsure.setUser(tInsureUser);
        }

        tPaymentRequest.setInsure(tInsure);


TrxResponse tTrxResponse = tPaymentRequest.extendPostRequest(1);
if (tTrxResponse.isSuccess()) {
   System.out.println("PaymentURL-->"+tTrxResponse.getValue("PaymentURL"));
   response.sendRedirect(tTrxResponse.getValue("PaymentURL"));
}
else {

%>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-保险支付请求</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>保险支付请求<br>
<%
  out.println("ReturnCode   = [" + tTrxResponse.getReturnCode  () + "]<br>");
  out.println("ErrorMessage = [" + tTrxResponse.getErrorMessage() + "]<br>");
}
%>
<a href='Merchant.html'>回商户首页</a></CENTER>
</BODY></HTML>
```

Y、商户通过页面传参数提交表单方式进行保险支付请求范例

```
<%@ page pageEncoding="gb2312" %>
<%@ page contentType="text/html; charset=gb2312" %>
<%@ page import = "com.hitrust.trustpay.client.b2c.*" %>
<%@ page import = "com.hitrust.trustpay.client.*" %>
<% response.setHeader("Cache-Control", "no-cache"); %>
<% request.setCharacterEncoding("gb2312"); %>
<%
```

```
//1、取得支付请求所需要的信息
String tOrderNo         = request.getParameter("OrderNo"        );
String tOrderDesc       = request.getParameter("OrderDesc"      );
String tOrderDate       = request.getParameter("OrderDate"      );
String tOrderTime       = request.getParameter("OrderTime"      );
String tOrderAmountStr  = request.getParameter("OrderAmount"    );
String tOrderURL        = request.getParameter("OrderURL"       );
String tProductType     = request.getParameter("ProductType"    );
String tPaymentType     = request.getParameter("PaymentType"    );
String tNotifyType      = request.getParameter("NotifyType"     );
String tResultNotifyURL = request.getParameter("ResultNotifyURL");
String tMerchantRemarks = request.getParameter("MerchantRemarks");
double  tOrderAmount    = Double.parseDouble(tOrderAmountStr);
String tPaymentLinkType = request.getParameter("PaymentLinkType");
String tBuyIP = request.getParameter("BuyIP");

//Insure
String tInsureType           = request.getParameter("InsureType"       );
String tFurl                 = request.getParameter("Furl"             );
String tOrderCategory        = request.getParameter("OrderCategory"    );
String tOrderCode            = request.getParameter("OrderCode"        );
String tOrderMode            = request.getParameter("OrderMode"        );
String tOrderName            = request.getParameter("OrderName"        );
String tInsureOrderAmountStr = request.getParameter("InsureOrderAmount");
double tInsureOrderAmount    = Double.parseDouble(tInsureOrderAmountStr);
String tUserCardNo           = request.getParameter("UserCardNo"       );
String tUserCertificateNo    = request.getParameter("CertificateNo"    );
String tUserCertificateType  = request.getParameter("CertificateType"  );
String tUserName             = request.getParameter("UserName"         );

String tSignature="";
//2、生成订单对象
Order tOrder = new Order();
tOrder.setOrderNo    (tOrderNo    ); //设定订单编号 （必要信息）
tOrder.setOrderDesc  (tOrderDesc  ); //设定订单说明
tOrder.setOrderDate  (tOrderDate  ); //设定订单日期 （必要信息 - YYYY/MM/DD）
tOrder.setOrderTime  (tOrderTime  ); //设定订单时间 （必要信息 - HH:MM:SS）
tOrder.setOrderAmount(tOrderAmount); //设定订单金额 （必要信息）
tOrder.setOrderURL   (tOrderURL   ); //设定订单网址
tOrder.setBuyIP    (tBuyIP    );       //设定订单IP
//3、生成定单订单对象，并将订单明细加入定单中（可选信息）
tOrder.addOrderItem(new OrderItem("IP000001", "中国移动IP卡", 100.00f, 1));
tOrder.addOrderItem(new OrderItem("IP000002", "网通IP卡"    ,  90.00f, 2));
```

```
MerchantConfig tMerchantConfig=MerchantConfig.getUniqueInstance();;
String sTrustPayIETrxURL=tMerchantConfig.getTrustPayIETrxURL();
String sErrorUrl=tMerchantConfig.getMerchantErrorURL();

//4、生成支付请求对象
PaymentRequest tPaymentRequest = new PaymentRequest();
tPaymentRequest.setOrder       (tOrder       ); //设定支付请求的订单 （必要信息）
tPaymentRequest.setProductType(tProductType); //设定商品种类 （必要信息）
                                              //PaymentRequest.PRD_TYPE_ONE：非实体商品，如服务、IP
卡、下载MP3、...
                                              //PaymentRequest.PRD_TYPE_TWO：实体商品
tPaymentRequest.setPaymentType(tPaymentType); //设定支付类型
                                              //PaymentRequest.PAY_TYPE_ABC：农行卡支付
                                              //PaymentRequest.PAY_TYPE_INT：国际卡支付
tPaymentRequest.setNotifyType(tNotifyType);   //设定商户通知方式
                                              //0：URL页面通知
                                              //1：服务器通知
tPaymentRequest.setResultNotifyURL(tResultNotifyURL); //设定支付结果回传网址 （必要信息）
tPaymentRequest.setMerchantRemarks(tMerchantRemarks); //设定商户备注信息
tPaymentRequest.setPaymentLinkType(tPaymentLinkType);//设定支付接入方式



        //
        Insure tInsure = new Insure();
        tInsure.setType(tInsureType);
        if (tInsure.getType().equals(Insure.INSURE_TYPE_FINANCING))
            tInsure.setFurl(tFurl);

        InsureOrder tInsureOrder = new InsureOrder();

        InsureOrderItem tInsureOrderItem = new InsureOrderItem();
        tInsureOrderItem.setCategory(tOrderCategory);
        tInsureOrderItem.setCode(tOrderCode);
        tInsureOrderItem.setMode(tOrderMode);
        tInsureOrderItem.setName(tOrderName);
        tInsureOrderItem.setAmount(tInsureOrderAmount);
        tInsureOrder.addOrderItem(tInsureOrderItem);

        tInsure.setOrder(tInsureOrder);

        if (tInsureType.equals(Insure.INSURE_TYPE_FINANCING) ||
tInsureType.equals(Insure.INSURE_TYPE_APPOINTED))
        {
            InsureUser tInsureUser = new InsureUser();
```

```
            tInsureUser.setCardNo(tUserCardNo);
            tInsureUser.setCertificateNo(tUserCertificateNo);
            tInsureUser.setCertificateType(tUserCertificateType);
            tInsureUser.setName(tUserName);

            tInsure.setUser(tInsureUser);
        }

        tPaymentRequest.setInsure(tInsure);

try{
 tSignature = tPaymentRequest.genSignature(1);
}catch (TrxException e){
       request.setAttribute("tReturnCode", e.getCode());
       request.setAttribute("tErrorMsg", e.getMessage());
       request.getRequestDispatcher("/ErrorPageInternal.jsp").forward(request, response);
       return;
}
%>
<HTML>
<HEAD><TITLE>农行网上支付平台-商户接口范例-保险支付请求</TITLE></HEAD>
<BODY BGCOLOR='#FFFFFF' TEXT='#000000' LINK='#0000FF' VLINK='#0000FF' ALINK='#FF0000'>
<CENTER>保险支付请求<br>
<form name="form1" method="post" action="<%=sTrustPayIETrxURL%>">
<input type="hidden" name="Signature" value="<%=tSignature%>">
<input type="hidden" name="errorPage" value="<%=sErrorUrl%>">
<TR><TD colspan=2><INPUT type=submit value="提交">

</form>
</BODY></HTML>
```

Z、网上付款信息发送请求范例

```
<%
//1、取得网上付款所需要的信息
        String tSerialNumber = request.getParameter("SerialNumber");
        String tTotalCount = request.getParameter("TotalCount");
        String tTotalAmount = request.getParameter("TotalAmount");
        String tCheckType = request.getParameter("CheckType");
        String tRemark = request.getParameter("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 = Integer.parseInt(tTotalCount);
        }
        catch(Exception ex)
```

```
        {
            out.println("ReturnCode   = [1101]<br/>");
            out.println("ErrorMessage = [付款总笔数不合法: " + tTotalCount + "]<br/>");
            return;
        }

        try
        {
            iTotalAmount = Double.parseDouble(tTotalAmount);
        }
        catch(Exception ex)
        {
            out.println("ReturnCode   = [1101]<br/>");
            out.println("ErrorMessage = [付款总金额不合法" + tTotalAmount + "]<br/>");
            return;
        }

        if (iTotalCount == 1)
        {
            String NO = request.getParameter("NO");
            String CardNo = request.getParameter("CardNo");
            String CardName = request.getParameter("CardName");
            String RemitAmount = request.getParameter("RemitAmount");
            String Purpose = request.getParameter("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.getParameter("NO").split(",");
            CardNo_arr = request.getParameter("CardNo").split(",");
            CardName_arr = request.getParameter("CardName").split(",");
            RemitAmount_arr = request.getParameter("RemitAmount").split(",");
            Purpose_arr = request.getParameter("Purpose").split(",");
        }
        java.util.ArrayList tRemitList = new java.util.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.setTotalCount(iTotalCount);//总笔数  （必要信息）
        tOnlineRemitRequest.setTotalAmount(iTotalAmount); //总金额  （必要信息）
        tOnlineRemitRequest.setSerialNumber(tSerialNumber);
        //tOnlineRemitRequest.setCheckType(tCheckType);
        tOnlineRemitRequest.setRemark(tRemark);//备注
        tOnlineRemitRequest.setRemitList(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
            out.println("TrxType    = [" + tResponse.getValue("TrxType") + "]<br/>");
            out.println("TotalCount  = [" + tResponse.getValue("TotalCount") + "]<br/>");
            out.println("TotalAmount = [" + tResponse.getValue("TotalAmount") + "]<br/>");
            out.println("SerialNumber  = [" + tResponse.getValue("SerialNumber") + "]<br/>");
            out.println("ResultMessage   = [" + tResponse.getErrorMessage() + "]<br/>");
        }
```

```
        else
        {
            //6、网上付款发送失败
            out.println("ReturnCode    = [" + tResponse.getReturnCode() + "]<br/>");
            out.println("ErrorMessage = [" + tResponse.getErrorMessage() + "]<br/>");
        }
%>
```

AA、网上付款交易结果查询请求范例

```
<%
        StringBuffer strMessage = new StringBuffer("");

        String tSerialNumber = request.getParameter("SerialNumber");
        String tReceiveAccount = request.getParameter("ReceiveAccount");

        //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.setSerialNumber(tSerialNumber);
        //tRequest.setPayAccount(request.getParameter("PayAccount"));
        tRequest.setReceiveAccount(tReceiveAccount);
        tRequest.setStartTime(request.getParameter("StartTime"));
        tRequest.setEndTime(request.getParameter("EndTime"));
        tRequest.setRemark(request.getParameter("Remark"));

        //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 (tSerialNumber != "" && 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
                java.util.ArrayList tQueryResults = tBatch.getQueryResults();
                for (int i = 0; i < tQueryResults.size(); i++)
                {
                    com.hitrust.trustpay.client.b2c.QueryResult tQueryResult =
(com.hitrust.trustpay.client.b2c.QueryResult)tQueryResults.get(i);
                    strMessage.append("No                    = [" + tQueryResult.getNo() +
"]<br/>");
                    strMessage.append("PayAccountNo          = [" +
tQueryResult.getPayAccountNo() + "]<br/>");
                    strMessage.append("PayAccountName        = [" +
tQueryResult.getPayAccountName() + "]<br/>");
                    strMessage.append("ReceiveAccountNo      = [" +
tQueryResult.getReceiveAccountNo() + "]<br/>");
                    strMessage.append("ReceiveAccountName    = [" +
tQueryResult.getReceiveAccountName() + "]<br/>");
```

```
                    strMessage.append("Purpose              = [" + tQueryResult.getPurpose()
+ "]<br/>");
                    strMessage.append("PayAmount            = [" +
tQueryResult.getPayAmount() + "]<br/>");
                    strMessage.append("Status               = [" + tQueryResult.getStatus() +
"]<br/>");
                    strMessage.append("FailReason           = [" +
tQueryResult.getFailReason() + "]<br/>");
                }
            }
            else if (tSerialNumber == "" && tReceiveAccount !="")
            {
                //5、取得查询结果明细
                java.util.ArrayList tQueryResults = tBatch.getQueryResults();
                for (int i = 0; i < tQueryResults.size(); i++)
                {
                    com.hitrust.trustpay.client.b2c.QueryResult tQueryResult =
(com.hitrust.trustpay.client.b2c.QueryResult)tQueryResults.get(i);
                    strMessage.append("No                   = [" + tQueryResult.getNo() +
"]<br/>");
                    strMessage.append("SerialNumber         = [" +
tQueryResult.getSerialNumber() + "]<br/>");
                    strMessage.append("TrnxTime             = [" + tQueryResult.getTrnxTime()
+ "]<br/>");
                    strMessage.append("PayAccountNo         = [" +
tQueryResult.getPayAccountNo() + "]<br/>");
                    strMessage.append("PayAccountName       = [" +
tQueryResult.getPayAccountName()+ "]<br/>");
                    strMessage.append("ReceiveAccountNo     = [" +
tQueryResult.getReceiveAccountNo() + "]<br/>");
                    strMessage.append("ReceiveAccountName   = [" +
tQueryResult.getReceiveAccountName() + "]<br/>");
                    strMessage.append("Purpose              = [" + tQueryResult.getPurpose()
+ "]<br/>");
                    strMessage.append("PayAmount            = [" +
tQueryResult.getPayAmount() + "]<br/>");
                    strMessage.append("Status               = [" + tQueryResult.getStatus() +
"]<br/>");
                    strMessage.append("FailReason           = [" +
tQueryResult.getFailReason() + "]<br/>");
                }
            }
            else if (tSerialNumber != "" && tReceiveAccount != "")
            {
                //5、取得查询结果明细
                java.util.ArrayList tQueryResults = tBatch.getQueryResults();
                for (int i = 0; i < tQueryResults.size(); i++)
                {
                    com.hitrust.trustpay.client.b2c.QueryResult tQueryResult =
(com.hitrust.trustpay.client.b2c.QueryResult)tQueryResults.get(i);
                    strMessage.append("No                   = [" + tQueryResult.getNo() +
"]<br/>");
                    strMessage.append("SerialNumber         = [" +
tQueryResult.getSerialNumber() + "]<br/>");
                    strMessage.append("TrnxTime             = [" + tQueryResult.getTrnxTime()
+ "]<br/>");
                    strMessage.append("PayAccountNo         = [" +
tQueryResult.getPayAccountNo() + "]<br/>");
                    strMessage.append("PayAccountName       = [" +
tQueryResult.getPayAccountName() + "]<br/>");
                    strMessage.append("ReceiveAccountNo     = [" +
tQueryResult.getReceiveAccountNo() + "]<br/>");
                    strMessage.append("ReceiveAccountName   = [" +
tQueryResult.getReceiveAccountName() + "]<br/>");
                    strMessage.append("Purpose              = [" + tQueryResult.getPurpose()
+ "]<br/>");
                    strMessage.append("PayAmount            = [" +
tQueryResult.getPayAmount() + "]<br/>");
                    strMessage.append("Status               = [" + tQueryResult.getStatus() +
"]<br/>");
                    strMessage.append("FailReason           = [" +
tQueryResult.getFailReason() + "]<br/>");
                }
            }
        }
```

```
        else
        {

            strMessage.append("ReturnCode   = [" + tTrxResponse.getReturnCode() + "]<br/>");
            strMessage.append("ErrorMessage = [" + tTrxResponse.getErrorMessage() +
"]<br/>");
        }
       out.println(strMessage.toString());

 %>
```

AB、网上付款银行卡状态验证请求范例

```
<%
                //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.OnlineRmtCardVerifyRequest tRequest = new
com.hitrust.trustpay.client.b2c.OnlineRmtCardVerifyRequest();
        String iBankCardNo = request.getParameter("txtBankCardNo");
        String iAccountName = new
String(request.getParameter("txtAccountName").getBytes("8859_1"),"gb2312");

        tRequest.setBankCardNo(iBankCardNo);  //银行帐号        （必要信息）
               tRequest.setAccountName(iAccountName);  //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();
        StringBuffer strMessage = new StringBuffer("");
        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.getReturnCode() + "]<br/>");
            strMessage.append("ErrorMessage = [" + tTrxResponse.getErrorMessage() +
"]<br/>");
        }

        out.println(strMessage.toString());

 %>
```

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

附录三、TrustPay Client API

本章中所有对象的属性皆可用 set*Attribute*() 来设定相对应的属性，并用 get*Attribute*()来取得属性的值。如要取得 TrxResponse 对象的 ReturnCode 属性，可以使用 getReturnCode()来取得；如要设定 TrxResponse 对象的 ReturnCode 属性，可以使用 setReturnCode()。

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
ReturnCode	响应码	String	响应码请参考《附录二、响应码一览表》的说明
ErrorMessage	错误信息	String	交易错误原因的本文说明

■ **方法**

◆ public TrxResponse(XMLDocument aXMLDocument)

TrxResponse 的构造函数，使用 XMLDocument 初始对象的属性。

参数：aXMLDocument – 网上支付平台响应的 XML 报文。

◆ 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(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	String 最大长度 50 个字符	支付请求时必须设定
OrderDesc	订单说明	String 最大长度 100 个字符	支付请求时可设定
OrderDate	订单日期	String YYYY/MM/DD	支付请求时必须设定
OrderTime	订单时间	String HH:MM:SS	支付请求时必须设定
OrderAmount	订单金额	float 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	String 最大长度 200 个字符	支付请求时必须设定
BuyIP	买方客户端 IP 地址信息	String	建议设定
PayAmount	支付金额	float 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
RefundAmount	退货金额	float 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	
OrderStatus	订单状态	String 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()

Order 的构造函数，生成一个空的订单对象。

◆ public Order(XMLDocument aXMLDocument)

构造 Order 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 Order addOrderItem(OrderItem aOrderItem)

新增一笔订单明细。

参数：aOrderItem – 订单明细对象

◆ public Order clearOrderItems()

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	String 最大长度 20 个字符	支付请求时必须设定
ProductName	产品名称	String 最大长度 50 个字符	支付请求时必须设定
UnitPrice	产品单价	float 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	int	支付请求时必须设定
Amount	金额小计	float 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 UnitPrice * Qty

■ **方法**

◆ public OrderItem()

OrderItem 的构造函数，生成一个空的订单明细对象。

◆ public OrderItem(String aProductID, String aProductName, float aUnitPrice, int aQty)

构造 OrderItem 对象，并使用参数初始对象的属性。

参数：aProductID - 产品代码

aProductName - 产品名称

aUnitPrice - 产品单价

aQty - 购买数量

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	String 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。
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。
MerchantRemarks	商户 备注信息	String 最大长度 200 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。
Insure	保单对象	Insure 类型	只有在进行保险直销支付请求的时候必须设定。

■ **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	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。

■ **属性**

■ **方法**

◆ public PaymentResult(String aMessage)

构造 PaymentResult 对象，并使用参数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.getParameter("MSG") 来取得。

回传值：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(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	PAY01：银行网点注册客户支付 PAY02：网上注册客户支付 PAY03：威士(VISA)国际卡支付

		PAY04：万事达(MasterCard)国际卡支付 PAY09：贷记卡网上支付
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	String 最大长度 50 个字符	必须设定
EnableDetailQuery	是否查询详细信息	boolean	必须设定 true：查询详细信息 false：不查询详细信息

■ **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	Query：订单状态查询
Order	订单信息	以字符串表示的订单信息。 可用来初始订单对象。

com.hitrust.trustpay.client.b2c.VoidPaymentRequest

■ **说明**

商户端接口软件包业务处理类，负责商户提交取消支付交易的处理。

■ **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

■ **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	VoidPay：取消支付
OrderNo	订单号	
PayAmount	支付金额	
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.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	String 最大长度 50 个字符	必须设定，指原交易订单号
NewOrderNo	退款订单编号	String 最大长度 50 个字符	可选，为标识退款订单，商户可自行设定
TrxAmount	退货金额	float 单位为人民币元 小数点后最多两位	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	Refund：退货请求
OrderNo	订单号	
NewOrderNo	退款订单号	
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	String YYYY/MM/DD	必须设定
SettleType	对账类型	String 代码	必须设定 TRX：交易对账单下载

■ **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。

■ **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(TrxResponse aTrxResponse)

构造 SettleFile 对象，并使用参数初始对账单对象的属性。

参数：aTrxResponse – 对账单下载回传的交易响应对象。

◆ 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
iResultNotifyURL	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
iBankCardNo	银行卡号	String 最大长度 20 个字符	必须设定
iCertificateType	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
iCertificateNo	证件号码	String 最大长度 30 个字符	必须设定
iRequestDate	请求日期	String YYYY/MM/DD 的日期格式	必须设定
iRequestTime	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
iSerialNumber	退款批量流水号	String	退款批量查询请求时必须设定
iRefundAmount	退款总金额	double	不需设定
iRefundCount	退款总笔数	int	不需设定
iStatus	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。
iOrder	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，代表批量退款批量对象。

■ **属性**

属性	属性名称	类型	说明
iSerialNumber	批量退款批量流水号	String	批量退款查询请求时必须设定
iRefundAmount	批量退款总金额	double	不需设定
iRefundCount	批量退款总笔数	int	不需设定
iStatus	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。
iOrder	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
iSerialNumber	退款批量流水号	String 最大长度 30 个字符	必须设定
iOperatorNo	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
iSerialNumber	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批量请求交易的处理。

■ **属性**

属性	属性名称	类型	说明
iTotalAmount	批量退款订单总金额	double	必须设定
iTotalCount	批量退款订单总笔数	int	必须设定

sRemark	批量退款备注信息	String	最长 30 个字符
iOrderlist	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
iSerialNumber	批量退款流水号	String	必须设定

		最大长度 30 个字符	

■ **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。 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。
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： 0：未处理；1：处理成功；2：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

iOrderNo	订单编号	String 最大长度 50 个字符	必须设定
iRequestDate	请求日期	String YYYY/MM/DD 的日期格式	必须设定
iRequestTime	请求时间	String HH:MM:SS 的时间格式	必须设定
iCertificateType	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
iCertificateNo	证件号码	当前系统必须设置为公民身份证	必须设定
iNotifyType	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。
iResultNotifyURL	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。

■ **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
iOrderNo	订单编号	String 最大长度 50 个字符	必须设定
iRequestDate	请求日期	String YYYY/MM/DD 的日期格式	必须设定
iRequestTime	请求时间	String HH:MM:SS 的时间格式	必须设定
iCertificateType	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
iCertificateNo	证件号码	当前系统必须设置为公民身份证	必须设定
iNotifyType	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。
iResultNotifyURL	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。
iAgentSignNo	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
iOrderNo	订单编号	String 最大长度 50 个字符	必须设定
iRequestDate	请求日期	String YYYY/MMDD 的日期格式	必须设定
iRequestTime	请求时间	String HH:MM:SS 的时间格式	必须设定
iCertificateNo	证件号码	当前系统必须设置为公民身份证	必须设定
iAgentSignNo	签约协议号	String	必须设定
iCurrency	账单币种	String	必须设定,设置为：“RMB”
iAmount	账单金额	String 小数点后最多两位	必须设定
iProductId	商品编号	String	必须设定
iProductName	商品名称	String	必须设定
iQuantity	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
iMerchantNo	商户编号	String	不需设定 网上支付平台响应
iBatchNo	批量编号	String	构建委托扣款批量请求时必须设置
iBatchDate	批量日期	String	构建委托扣款批量请求时必须设置
iBatchTime	批量时间	long	不需设定 网上支付平台响应
iAgentAmount	委托扣款总金额	double	构建委托扣款批量请求时必须设置
iAgentCount	委托扣款总笔数	int	构建委托扣款批量请求时必须设置
iTrnxTypeNo	交易类型编码	String	
iBatchStatus	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
iAgentBatchDetail	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.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
iMerchantNo	商户编号	String	不需设定 网上支付平台响应
iBatchNo	批量编号	String	
iBatchDate	批量日期	String	
iOrderNo	订单编号	String	构建委托扣款批量请求时必须设置
iOrderAmount	订单金额	double	构建委托扣款批量请求时必须设置
iCurrency	订单币种	String	
iCertificateNo	证件号码	String	构建委托扣款批量请求时必须设置
iContractID	签约协议号	String	构建委托扣款批量请求时必须设置
iProductID	商品编号	String	构建委托扣款批量请求时必须设置

iProductName	商品名称	String	构建委托扣款批量请求时必须设置
iProductNum	商品数量	int	构建委托扣款批量请求时必须设置
iOrderStatus	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
iAgentBatch	委托扣款批量对象	AgentBatch	必须设定
iAgentBatchDetailLi	批次明细对象集	Vector	必须设定

st	合	AgentBatchDetail 对象的 Vector 集合	

■ 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
iBatchNo	批次号	String	必须设定
iBatchDate	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：一般支付、理财支付、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。

■ **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

■ 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

■ 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

■ **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 void checkRequest()

网上付款请求信息是否合法

protected 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 void checkRequest()

支付请求信息是否合法

protected 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 void checkRequest()

网上付款交易结果查询信息是否合法

protected 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 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	
ReceiveAccountNa	收款方户名	string	

me			
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。