**电商系统-微服务支付系统**

# 互联网支付方式的分类

## 什么是在线支付

在线支付是指卖方与买方通过因特网上的电子商务网站进行交易时，银行为其提供网上资金结算服务的一种业务。它为企业和个人提供了一个安全、快捷、方便的电子商务应用环境和网上资金结算工具。在线支付不仅帮助企业实现了销售款项的快速归集，缩短收款周期，同时也为个人网上银行客户提供了网上消费支付结算方式，使客户真正做到足不出户，网上购物。

## 什么是聚合支付

|  |
| --- |
| 聚合支付：也称“融合支付”，是指只从事“支付、结算、清算”服务之外的“支付服务”，依托银行、非银机构或清算组织，借助银行、非银机构或清算组织的支付通道与清结算能力，利用自身的技术与服务集成能力，将一个以上的银行、非银机构或清算组织的支付服务，整合到一起，为商户提供包括但不限于“支付通道服务”、“集合对账服务”、“技术对接服务”、“差错处理服务”、“金融服务引导”、“会员账户服务”、“作业流程软件服务”、“运行维护服务”、“终端提供与维护”等服务内容，以此减少商户接入、维护支付结算服务时面临的成本支出，提高商户支付结算系统运行效率的，并收取增值收益的支付服务。 |

# 在线支付系统流程

回调方式：同步回调、异步回调

回调场景: 告诉商户支付通知结果

同步回调: 整个支付流程完毕，使用同步方式将参数重定向给商户平台，一般场景用于展示结果。

异步回调: 第三方支付接口发一个后台通知给商户平台，一般场景用户修改订单信息。

# 数据安全加密

JAVA中为我们提供了丰富的加密技术，可以基本的分为单向加密和非对称加密

## 单向加密

单向加密算法主要用来验证数据传输的过程中，是否被篡改过。

BASE64 严格地说，属于编码格式，而非加密算法

MD5(Message Digest algorithm 5，信息摘要算法)

SHA(Secure Hash Algorithm，安全散列算法)

HMAC(Hash Message Authentication Code，散列消息鉴别码

## 对称加密

你可以这么理解，一方通过密钥将信息加密后，把密文传给另一方，另一方通过这个相同的密钥将密文解密，转换成可以理解的明文。他们之间的关系如下：

明文 <-> 密钥 <-> 密文

常用对称加密方案 DES、AES、Base64

## 非对称加密

在通信双方，如果使用非对称加密，一般遵从这样的原则：公钥加密，私钥解密。同时，一般一个密钥加密，另一个密钥就可以解密。

因为公钥是公开的，如果用来解密，那么就很容易被不必要的人解密消息。因此，**私钥也可以认为是个人身份的证明。**

如果通信双方需要互发消息，那么应该建立两套非对称加密的机制（即两对公私钥密钥对），发消息的一方使用对方的公钥进行加密，接收消息的一方使用自己的私钥解密。

就从上面提到的这个对称加密的缺点开始，怎么做到即时一个人的密钥被盗窃了，最起码保证你给其他人发送密文不被破解。于是，人们就想出了这么个办法，首先，我们停止分享共同的密钥，因为上面的 bug 就是来源于共享一个密钥，那么怎么办呢？每个人生成一个“私钥-公钥”对，这个私钥需要每个人自行进行保护！公钥可以随便分享，后面详细说，同时，生成的这个“私钥-公钥”对还有个强大的功能就是，使用私钥加密的信息，只能由该私钥对应的公钥才能解密，使用公钥加密的信息，只能由该公钥对应的私钥才能解密！

常用非对称加密 RSA 一般用于数据安全加密特别，支付领域。

# RSA加密案例

## 生成RSA秘钥对

http://web.chacuo.net/netrsakeypair

# 银联在线支付对接

银联接口官方文档

<https://open.unionpay.com/ajweb/index>

测试账号

用户名称: itmayiedu 密码 15527339672w

# 银联官方demo分析

# 支付系统环境搭建

## 数据库表的设计

## 创建项目工程

imayiedu-shopp-pay-api ----支付服务api

imayiedu-shopp-pay ----支付服务实现

imayiedu-shopp-pay-web ----支付web

## imayiedu-shopp-pay-api

### YML配置

|  |
| --- |
| **server:**  **port: 8767**  **# context-path: /member**  **eureka:**  **client:**  **serviceUrl:**  **defaultZone: http://localhost:8761/eureka/**  **spring:**  **application:**  **name: pay-service**  **redis:**  **host: 106.15.185.133**  **password: 123456**  **port: 6379**  **pool:**  **max-idle: 100**  **min-idle: 1**  **max-active: 1000**  **max-wait: -1**  **#数据库连接信息**  **datasource:**  **name: test**  **url: jdbc:mysql://127.0.0.1:3306/itmayiedu-pay**  **username: root**  **password: root**  **# 使用druid数据源**  **type: com.alibaba.druid.pool.DruidDataSource**  **driver-class-name: com.mysql.jdbc.Driver**  **filters: stat**  **maxActive: 20**  **initialSize: 1**  **maxWait: 60000**  **minIdle: 1**  **timeBetweenEvictionRunsMillis: 60000**  **minEvictableIdleTimeMillis: 300000**  **validationQuery: select 'x'**  **testWhileIdle: true**  **testOnBorrow: false**  **testOnReturn: false**  **poolPreparedStatements: true**  **maxOpenPreparedStatements: 20**  **##activemq连接信息**  **activemq:**  **broker-url: tcp://localhost:61616**  **in-memory: true**  **pool:**  **enabled: false** |

### Maven依赖

|  |
| --- |
| **<dependencies>**  **<dependency>**  **<groupId>com.itmayiedu</groupId>**  **<artifactId>imayiedu-shopp-pay-api</artifactId>**  **<version>0.0.1-SNAPSHOT</version>**  **</dependency>**  **<!-- springboot整合mybatis -->**  **<dependency>**  **<groupId>org.mybatis.spring.boot</groupId>**  **<artifactId>mybatis-spring-boot-starter</artifactId>**  **<version>1.3.0</version>**  **</dependency>**  **</dependencies>** |

### PaymentInfoService

|  |
| --- |
| **/\*\***  **\* 支付接口**  **\***  **\* @author Administrator**  **\***  **\*/**  **@RequestMapping("/pay")**  **public interface PaymentInfoService {**  **/\*\***  **\* 换取支付令牌**  **\***  **\* @return**  **\*/**  **@RequestMapping("/addPayInfoToken")**  **public Map<String, Object> addPayInfoToken(@RequestBody PaymentInfo paymentInfo);**  **/\*\***  **\* 使用token查找支付信息**  **\***  **\* @param paymentInfo**  **\* @return**  **\*/**  **@RequestMapping("/getPayInfoToken")**  **public Map<String, Object> getPayInfoToken(@RequestParam("token") String token);**  **/\*\***  **\* 使用token查找支付信息**  **\***  **\* @param paymentInfo**  **\* @return**  **\*/**  **@RequestMapping("/getByOrderIdPayInfo")**  **public Map<String, Object> getByOrderIdPayInfo(@RequestParam("orderId") String orderId);**    **/\*\***  **\* 使用token查找支付信息**  **\***  **\* @param paymentInfo**  **\* @return**  **\*/**  **@RequestMapping("/updatePayInfo")**  **public Map<String, Object> updatePayInfo(@RequestBody PaymentInfo paymentInfo);**  **}** |

### PaymentTypeService

|  |
| --- |
| **@RequestMapping("/pay")**  **public interface PaymentTypeService {**  **@RequestMapping("/getPaymentType")**  **public Map<String, Object> getPaymentType(@RequestParam("id") Long id);**  **}** |

### PaymentInfoServiceImpl

|  |
| --- |
| **@RestController**  **@RequestMapping("/pay")**  **public class PaymentInfoServiceImpl extends BaseApiService implements PaymentInfoService {**  **@Autowired**  **private PaymentInfoDao paymentInfoDao;**  **@Autowired**  **private BaseRedisService baseRedisService;**  **@Autowired**  **private TokenUtils tokenUtils;**  **@Override**  **public Map<String, Object> addPayInfoToken(@RequestBody PaymentInfo paymentInfo) {**  **paymentInfo.setCreated(DateUtils.*getTimestamp*());**  **paymentInfo.setUpdated(DateUtils.*getTimestamp*());**  **paymentInfoDao.savePaymentType(paymentInfo);**  **Long paymentInfoId = paymentInfo.getId();**  **if (paymentInfoId == null) {**  **return setResutError("系统错误,为获取到支付id");**  **}**  **String token = tokenUtils.getPayToken();**  **baseRedisService.setString(token, paymentInfoId + "");**  **return setResutSuccessData(token);**  **}**  **@Override**  **public Map<String, Object> getPayInfoToken(@RequestParam("token") String token) {**  **if (StringUtils.*isEmpty*(token)) {**  **return setResutError("token不能为空!");**  **}**  **String payInfoId = baseRedisService.get(token);**  **Long newPayInfoId = Long.*parseLong*(payInfoId);**  **PaymentInfo paymentInfo = paymentInfoDao.getPaymentInfo(newPayInfoId);**  **return setResutSuccessData(paymentInfo);**  **}**  **@Override**  **public Map<String, Object> getByOrderIdPayInfo(@RequestParam("orderId")String orderId) {**  **PaymentInfo paymentInfo = paymentInfoDao.getByOrderIdPayInfo(orderId);**  **if(paymentInfo==null){**  **return setResutError("未查询到支付信息");**  **}**  **return setResutSuccessData(paymentInfo);**  **}**  **@Override**  **public Map<String, Object> updatePayInfo(@RequestBody PaymentInfo paymentInfo) {**  **paymentInfoDao.updatePayInfo(paymentInfo);**  **return setResutSuccess();**  **}**  **}** |

### PaymentTypeServiceImpl

|  |
| --- |
| @Service  @RestController  @RequestMapping("/pay")  **public** **class** PaymentTypeServiceImpl **extends** BaseApiService **implements** PaymentTypeService {  @Autowired  **private** PaymentTypeDao paymentTypeDao;  @RequestMapping("/getPaymentType")  **public** Map<String, Object> getPaymentType(@RequestParam("id") Long id) {  PaymentType paymentType = paymentTypeDao.getPaymentType(id);  **if** (paymentType == **null**) {  **return** setResutError("未查找到类型");  }  **return** setResutSuccessData(paymentType);  }  } |

### PaymentInfoDao

|  |
| --- |
| **@Mapper**  **public interface PaymentInfoDao extends BaseDao {**  **@Select("select \* from payment\_info where id=#{id}")**  **public PaymentInfo getPaymentInfo(@Param("id") Long id);**  **@Insert("insert into payment\_info ( id,typeid,orderid,platformorderid,price,source,state,created,updated) value(null,#{typeId},#{orderId},#{platformorderId},#{price},#{source},#{state},#{created},#{updated})")**  **@Options(useGeneratedKeys = true, keyProperty = "id") // 添加该行，product中的id将被自动添加**  **public Integer savePaymentType(PaymentInfo paymentInfo);**  **@Select("select \* from payment\_info where orderId=#{orderId}")**  **public PaymentInfo getByOrderIdPayInfo(@Param("orderId") String orderId);**  **@Update("update payment\_info set state =#{state},payMessage=#{payMessage},platformorderId=#{platformorderId},updated=#{updated} where orderId=#{orderId} ")**  **public void updatePayInfo(PaymentInfo paymentInfo);**  **}** |

### PaymentTypeDao

|  |
| --- |
| **@Mapper**  **public interface PaymentTypeDao {**  **@Select("select \* from payment\_type where id = #{id}")**  **public PaymentType getPaymentType(@Param("id") Long id);**  **}** |

## imayiedu-shopp-pay-web

### yml配置

|  |
| --- |
| **server:**  **port: 8768**  **context-path: /pay-web**  **eureka:**  **client:**  **serviceUrl:**  **defaultZone: http://localhost:8761/eureka/**  **spring:**  **application:**  **name: pay-web** |

### Maven依赖

|  |
| --- |
| **<dependencies>**  **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-freemarker</artifactId>**  **</dependency>**  **<!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->**  **<dependency>**  **<groupId>javax.servlet</groupId>**  **<artifactId>javax.servlet-api</artifactId>**  **<scope>provided</scope>**  **</dependency>**  **<dependency>**  **<groupId>com.itmayiedu</groupId>**  **<artifactId>itmayiedu-shopp-common</artifactId>**  **<version>0.0.1-SNAPSHOT</version>**  **</dependency>**  **<dependency>**  **<groupId>com.itmayiedu</groupId>**  **<artifactId>imayiedu-shopp-pay-api</artifactId>**  **<version>0.0.1-SNAPSHOT</version>**  **</dependency>**  **<dependency>**  **<groupId>com.sdk4j</groupId>**  **<artifactId>sdk4j</artifactId>**  **<version>1.0</version>**  **</dependency>**  **</dependencies>** |

## 将银联支付文件

acp\_sdk.properties和log4j.properties放入项目根目录

## 将银联sdk拷到项目中

## 启动加载配置文件

|  |
| --- |
| **@Component**  **public class AutoLoadRunner implements CommandLineRunner {**  **@Override**  **public void run(String... args) throws Exception {**  **SDKConfig.*getConfig*().loadPropertiesFromSrc();**  **System.*out*.println(">>>>>>>>>>>>>>>服务启动执行，执行加载数据等操作<<<<<<<<<<<<<");**  **}**  **}** |

### PayController

|  |
| --- |
| **@Controller**  **public class PayController extends BaseController {**  **@Autowired**  **private PaymentInfoFeign paymentInfoFeign;**  **private static final String *PAYGATEWAY* = "payGateway";**  **@Autowired**  **private PayImplService payService;**  **@RequestMapping("/payGateway")**  **public void payGateway(HttpServletRequest request, String token,HttpServletResponse resp) throws IOException {**  **resp.setContentType("text/html;charset=utf-8");**  **Map<String, Object> payInfoToken = paymentInfoFeign.getPayInfoToken(token);**  **Map<String, Object> resultMap = (Map<String, Object>) ResultUtils.*getResultMap*(payInfoToken);**  **PrintWriter out=resp.getWriter();**  **if (resultMap == null) {**  **resp.getWriter().write("系统错误!");**  **return ;**  **}**  **String json = new JSONObject().*toJSONString*(resultMap);**  **PaymentInfo paymentInfo = new JSONObject().*parseObject*(json, PaymentInfo.class);**  **String html = payService.pay(paymentInfo);**  **out.println(html);**  **out.close();**  **}**  **}** |

#### PayService

|  |
| --- |
| **public interface PayService {**  **public String pay(PaymentInfo paymentInfo);**  **}** |

#### PayImplService

|  |
| --- |
| @Slf4j  @Service  **public** **class** PayImplService **implements** PayService {  @Autowired  **private** PaymentTypeFeign paymentTypeFeign;  @Autowired  **private** YinLianPay yinLianPay;  @Override  **public** String pay(PaymentInfo paymentInfo) {  Long typeId = paymentInfo.getTypeId();  Map<String, Object> paymentTypeMap = paymentTypeFeign.getPaymentType(typeId);  **if** (paymentTypeMap == **null**) {  ***log***.error("####pay() typeId:{},paymentTypeMap is null");  **return** **null**;  }  Map<String, Object> data = (Map<String, Object>) ResultUtils.*getResultMap*(paymentTypeMap);  String json = **new** JSONObject().*toJSONString*(data);  PaymentType paymentType = **new** JSONObject().*parseObject*(json, PaymentType.**class**);  **if** (paymentType == **null**) {  **return** **null**;  }  String typeName = paymentType.getTypeName();  PayAdaptService payAdaptService=**null**;  **switch** (typeName) {  **case** "yinlianPay":  payAdaptService=yinLianPay;  **break**;  **default**:  **break**;  }  **return** payAdaptService.pay(paymentInfo, paymentType);  }  } |

#### PayAdaptService

|  |
| --- |
| **public** **interface** PayAdaptService {  **public** String pay(PaymentInfo paymentInfo,PaymentType paymentType );  } |

#### YinLianPay

|  |
| --- |
| **@Slf4j**  **@Service**  **public class YinLianPay implements PayAdaptService {**  **@Override**  **public String pay(PaymentInfo paymentInfo, PaymentType paymentType) {**  **return setPay(paymentType, paymentInfo);**  **}**  **public String setPay(PaymentType paymentType, PaymentInfo paymentInfo) {**  **Map<String, String> requestData = new HashMap<String, String>();**  **/\*\*\* 银联全渠道系统，产品参数，除了encoding自行选择外其他不需修改 \*\*\*/**  **requestData.put("version", DemoBase.*version*); // 版本号，全渠道默认值**  **requestData.put("encoding", DemoBase.*encoding\_UTF8*); // 字符集编码，可以使用UTF-8,GBK两种方式**  **requestData.put("signMethod", "01"); // 签名方法，只支持 01：RSA方式证书加密**  **requestData.put("txnType", "01"); // 交易类型 ，01：消费**  **requestData.put("txnSubType", "01"); // 交易子类型， 01：自助消费**  **requestData.put("bizType", "000201"); // 业务类型，B2C网关支付，手机wap支付**  **requestData.put("channelType", "07"); // 渠道类型，这个字段区分B2C网关支付和手机wap支付；07：PC,平板**  **// 08：手机**  **/\*\*\* 商户接入参数 \*\*\*/**  **requestData.put("merId", paymentType.getMerchantId()); // 商户号码，请改成自己申请的正式商户号或者open上注册得来的777测试商户号**  **requestData.put("accessType", "0"); // 接入类型，0：直连商户**  **requestData.put("orderId", paymentInfo.getOrderId()); // 商户订单号，8-40位数字字母，不能含“-”或“\_”，可以自行定制规则**  **requestData.put("txnTime", DemoBase.*getCurrentTime*()); // 订单发送时间，取系统时间，格式为YYYYMMDDhhmmss，必须取当前时间，否则会报txnTime无效**  **requestData.put("currencyCode", "156"); // 交易币种（境内商户一般是156 人民币）**  **requestData.put("txnAmt", paymentInfo.getPrice().toString()); // 交易金额，单位分，不要带小数点**  **// requestData.put("reqReserved", "透传字段");**  **// //请求方保留域，如需使用请启用即可；透传字段（可以实现商户自定义参数的追踪）本交易的后台通知,对本交易的交易状态查询交易、对账文件中均会原样返回，商户可以按需上传，长度为1-1024个字节**  **// 前台通知地址 （需设置为外网能访问 http https均可），支付成功后的页面 点击“返回商户”按钮的时候将异步通知报文post到该地址**  **// 如果想要实现过几秒中自动跳转回商户页面权限，需联系银联业务申请开通自动返回商户权限**  **// 异步通知参数详见open.unionpay.com帮助中心 下载 产品接口规范 网关支付产品接口规范 消费交易 商户通知**  **requestData.put("frontUrl", paymentType.getFrontUrl());**  **// 后台通知地址（需设置为【外网】能访问 http**  **// https均可），支付成功后银联会自动将异步通知报文post到商户上送的该地址，失败的交易银联不会发送后台通知**  **// 后台通知参数详见open.unionpay.com帮助中心 下载 产品接口规范 网关支付产品接口规范 消费交易 商户通知**  **// 注意:1.需设置为外网能访问，否则收不到通知 2.http https均可 3.收单后台通知后需要10秒内返回http200或302状态码**  **// 4.如果银联通知服务器发送通知后10秒内未收到返回状态码或者应答码非http200，那么银联会间隔一段时间再次发送。总共发送5次，每次的间隔时间为0,1,2,4分钟。**  **// 5.后台通知地址如果上送了带有？的参数，例如：http://abc/web?a=b&c=d**  **// 在后台通知处理程序验证签名之前需要编写逻辑将这些字段去掉再验签，否则将会验签失败**  **requestData.put("backUrl", paymentType.getBackUrl());**  **//////////////////////////////////////////////////**  **//**  **// 报文中特殊用法请查看 PCwap网关跳转支付特殊用法.txt**  **//**  **//////////////////////////////////////////////////**  **/\*\* 请求参数设置完毕，以下对请求参数进行签名并生成html表单，将表单写入浏览器跳转打开银联页面 \*\*/**  **Map<String, String> submitFromData = AcpService.*sign*(requestData, DemoBase.*encoding\_UTF8*); // 报文中certId,signature的值是在signData方法中获取并自动赋值的，只要证书配置正确即可。**  **String requestFrontUrl = SDKConfig.*getConfig*().getFrontRequestUrl(); // 获取请求银联的前台地址：对应属性文件acp\_sdk.properties文件中的acpsdk.frontTransUrl**  **String html = AcpService.*createAutoFormHtml*(requestFrontUrl, submitFromData, DemoBase.*encoding\_UTF8*); // 生成自动跳转的Html表单**  **LogUtil.*writeLog*("打印请求HTML，此为请求报文，为联调排查问题的依据：" + html);**  **// 将生成的html写到浏览器中完成自动跳转打开银联支付页面；这里调用signData之后，将html写到浏览器跳转到银联页面之前均不能对html中的表单项的名称和值进行修改，如果修改会导致验签不通过**  **return html;**  **}**  **}** |

### YinLianCallbackController

|  |
| --- |
| **@RequestMapping("/pay/callback")**  **@Controller**  **public class YinLianCallbackController {**  **@Autowired**  **private CallbackService callbackService;**  **private static final String *PAY\_SUCCESS* = "pay\_success";**  **private static final String *PAY\_FAIL* = "pay\_fail";**  **/\*\***  **\* 功能说明:同步回调**  **\***  **\* @return**  **\*/**  **@RequestMapping("/syn")**  **public String syn(HttpServletRequest request) {**  **Map<String, String> synResultMap = callbackService.syn(request);**  **String encoding = request.getParameter(SDKConstants.*param\_encoding*);**  **if (!AcpService.*validate*(synResultMap, encoding)) {**  **LogUtil.*writeLog*("验证签名结果[失败].");**  **return *PAY\_FAIL*;**  **}**  **request.setAttribute("txnAmt", Double.*parseDouble*(synResultMap.get("txnAmt")) / 100);**  **request.setAttribute("orderId", Long.*parseLong*(synResultMap.get("orderId")));**  **LogUtil.*writeLog*("验证签名结果[成功].");**  **return *PAY\_SUCCESS*;**  **}**  **/\*\***  **\* 功能说明:异步回调**  **\***  **\* @return**  **\*/**  **@RequestMapping("/asyn")**  **public String asyn(HttpServletRequest request) {**  **return callbackService.asyn(request);**  **}**  **}** |

#### CallbackService

|  |
| --- |
| **public interface CallbackService {**  **/\*\***  **\* 同步回调**  **\***  **\* @return**  **\*/**  **public Map<String, String> syn(HttpServletRequest request);**  **/\*\***  **\* 异步回调**  **\***  **\* @return**  **\*/**  **public String asyn(HttpServletRequest request);**  **}** |

#### YinLianCallbackService

|  |
| --- |
| **@Slf4j**  **@Service**  **public class YinLianCallbackService implements CallbackService {**  **@Autowired**  **private PaymentInfoFeign paymentInfoFeign;**  **@Override**  **public Map<String, String> syn(HttpServletRequest req) {**  **LogUtil.*writeLog*("FrontRcvResponse前台接收报文返回开始");**  **try {**  **String encoding = req.getParameter(SDKConstants.*param\_encoding*);**  **Map<String, String> valideData = *valideData*(req, encoding);**  **return valideData;**  **} catch (Exception e) {**  ***log*.error("#####YinLianCallbackService##syn() ERROR:", e);**  **return null;**  **} finally {**  **LogUtil.*writeLog*("FrontRcvResponse前台接收报文返回结束");**  **}**  **}**  **@SuppressWarnings({ "static-access", "unchecked" })**  **@Override**  **public String asyn(HttpServletRequest req) {**  **LogUtil.*writeLog*("BackRcvResponse接收后台通知开始");**  **String encoding = req.getParameter(SDKConstants.*param\_encoding*);**  **Map<String, String> valideData = *valideData*(req, encoding);**  **// 重要！验证签名前不要修改reqParam中的键值对的内容，否则会验签不过**  **if (!AcpService.*validate*(valideData, encoding)) {**  **LogUtil.*writeLog*("验证签名结果[失败].");**  **// 验签失败，需解决验签问题**  **return "fail";**  **}**  **LogUtil.*writeLog*("验证签名结果[成功].");**  **// 【注：为了安全验签成功才应该写商户的成功处理逻辑】交易成功，更新商户订单状态**  **String orderId = valideData.get("orderId"); // 获取后台通知的数据，其他字段也可用类似方式获取**  **Map<String, Object> byOrderIdPayInfoResult = paymentInfoFeign.getByOrderIdPayInfo(orderId);**  **if (byOrderIdPayInfoResult != null) {**  **Map<String, Object> resultMap = (Map<String, Object>) ResultUtils.*getResultMap*(byOrderIdPayInfoResult);**  **if (resultMap == null) {**  **return Constants.*PAY\_FAIL*;**  **}**  **String json = new JSONObject().*toJSONString*(resultMap);**  **PaymentInfo paymentInfo = new JSONObject().*parseObject*(json, PaymentInfo.class);**  **Integer state = paymentInfo.getState();**  **if (state.equals(1)) {**  ***log*.error("###asyn() 异步通知,订单id:{},状态已经为已支付,不能在修改为已支付.",orderId);**  **return Constants.*PAY\_SUCCES*;**  **}**  **paymentInfo.setPlatformorderId(valideData.get("queryId"));**  **paymentInfo.setUpdated(DateUtils.*getTimestamp*());**  **paymentInfo.setPayMessage(new JSONObject().*toJSONString*(valideData));**  **paymentInfo.setState(1);**  **paymentInfoFeign.updatePayInfo(paymentInfo);**  ***log*.error("###asyn() 异步通知,订单id:{},状态已经为已支付,修改成功。",orderId);**  **}**  **// 返回给银联服务器http 200 状态码**  **return Constants.*PAY\_SUCCES*;**  **}**  **public static Map<String, String> valideData(final HttpServletRequest req,String encoding){**  **// 获取银联通知服务器发送的后台通知参数**  **Map<String, String> reqParam = *getAllRequestParam*(req);**  **LogUtil.*printRequestLog*(reqParam);**  **Map<String, String> valideData = null;**  **if (null != reqParam && !reqParam.isEmpty()) {**  **Iterator<Entry<String, String>> it = reqParam.entrySet().iterator();**  **valideData = new HashMap<String, String>(reqParam.size());**  **while (it.hasNext()) {**  **Entry<String, String> e = it.next();**  **String key = (String) e.getKey();**  **String value = (String) e.getValue();**  **try {**  **value = new String(value.getBytes(encoding), encoding);**  **} catch (Exception e2) {**  **// TODO: handle exception**  **}**  **valideData.put(key, value);**  **}**  **}**  **return valideData;**  **}**  **/\*\***  **\* 获取请求参数中所有的信息**  **\***  **\* @param request**  **\* @return**  **\*/**  **public static Map<String, String> getAllRequestParam(final HttpServletRequest request) {**  **Map<String, String> res = new HashMap<String, String>();**  **Enumeration<?> temp = request.getParameterNames();**  **if (null != temp) {**  **while (temp.hasMoreElements()) {**  **String en = (String) temp.nextElement();**  **String value = request.getParameter(en);**  **res.put(en, value);**  **// 在报文上送时，如果字段的值为空，则不上送<下面的处理为在获取所有参数数据时，判断若值为空，则删除这个字段>**  **if (res.get(en) == null || "".equals(res.get(en))) {**  **// System.out.println("======为空的字段名===="+en);**  **res.remove(en);**  **}**  **}**  **}**  **return res;**  **}**  **}** |

## 开发流程

其他系统与支付系统对接时，为了能够保证整个传输的安全性。

使用token令牌方式传输。

步骤:

1. 其他系统生成支付令牌
2. 其他系统使用支付令牌调用支付系统支付接口
3. 支付系统接口根据令牌的判断，跳转到具体的第三方支付网关接口
4. 使用异步通知,根据实际支付结果,修改订单状态。

# 报文格式

|  |
| --- |
| {  "typeId":"1",  "orderId":"123456",  "platformorderId":"123456789",  "price":"10",  "source":"1",  "state":"0"  } |

# 外网测试工具

Ngrok、花生壳。

## Ngrok用法

●windows用户:

1,下载windows版本的客户端，解压到你喜欢的目录  
2,在命令行下进入到path/to/windows\_386/下  
3,执行 ngrok -config=ngrok.cfg -subdomain xxx 80 //(xxx 是你自定义的域名前缀)

4,如果开启成功 你就可以使用 xxx.tunnel.qydev.com 来访问你本机的 127.0.0.1:80 的服务啦  
5,如果你自己有顶级域名，想通过自己的域名来访问本机的项目，那么先将自己的顶级域名解析到123.57.165.240(域名需要已备案哦)，然后执行./ngrok -config=ngrok.cfg -hostname xxx.xxx.xxx 80 //(xxx.xxx.xxx是你自定义的顶级域名)  
6，如果开启成功 你就可以使用你的顶级域名来访问你本机的 127.0.0.1:80 的服务啦  
7,如果失败 就加下交流群 反馈下问题 本屌会看看什么原因....吧

# 常见测试问题

## 查询不到测试订单信息

查询测试交易信息，必须有订单号和商户号一起查询。

如果还是查询不到,官方客服说,会有10分时间延迟。

## 测试交易卡号信息

https://open.unionpay.com/ajweb/help/faq/list?id=4&level=0&from=0&keyword=%E6%B5%8B%E8%AF%95%E5%8D%A1

生成token

<http://itmayiedu.tunnel.qydev.com/pay-web/payGateway?token=pay-791d86d7-e545-4ac9-8705-8d7223efa03>

## 常见支付面试题

### 怎么解决支付重试回调幂等性？

① 在网络延迟的情况下， 先查找支付信息是否已经支付成功，如果已经支付成功，直接返回给ok给支付接口，防止一直方式。

②如果是有延迟的时候，使用mq推送。

### 支付系统之间怎么防止金额篡改？

### 微服务系统间怎么保证支付系统的传输安全性？