**微信支付**

**Java版**

极学网

目录

[1、 创建一个工程 2](#_Toc531341156)

[1.1 创建一个Spring Boot工程 2](#_Toc531341157)

[1.2 引入JQuery与JQueryQRCode 2](#_Toc531341158)

[1.3 引用微信官方DEMO （Java）版的几个Class类 2](#_Toc531341159)

[1.4 编写HttpClientUtils类 2](#_Toc531341160)

[1.5 编写HttpCertClientUtils类 4](#_Toc531341161)

[1.6 编写ResponseUtils类 7](#_Toc531341162)

[2、 启动Ngrok 9](#_Toc531341163)

[3、 微信支付功能 10](#_Toc531341164)

[2.1 支付配置文件 10](#_Toc531341165)

[3.1 统一下单 11](#_Toc531341167)

[3.2 查询订单 11](#_Toc531341168)

[3.3 关闭订单 11](#_Toc531341169)

[3.4 申请退款 11](#_Toc531341170)

[3.5 查询退款 11](#_Toc531341171)

[3.6 支付结果通知 11](#_Toc531341172)

[3.7 退款结果通知 12](#_Toc531341173)

# 创建一个工程

## 创建一个Spring Boot工程

|  |
| --- |
| Pom.xml文件需要引入HttpClient包及解密包 |
| *<!-- HTTP Client -->* <**dependency**>  <**groupId**>org.apache.httpcomponents</**groupId**>  <**artifactId**>httpclient</**artifactId**>  <**version**>4.5.5</**version**> </**dependency**>  *<!-- 微信退款解密 -->* <**dependency**>  <**groupId**>org.bouncycastle</**groupId**>  <**artifactId**>bcprov-jdk15on</**artifactId**>  <**version**>1.47</**version**> </**dependency**> |

## 引入JQuery与JQueryQRCode

|  |
| --- |
| 引入JS文件 |
| JQuery 3.3.1  JQuery QRCode 0.14 |

## 引用微信官方DEMO （Java）版的几个Class类

|  |
| --- |
| 微信官方DEMO |
| * WXPayConstants.java * WXPayUtil.java * WXPayXmlUtil.java |

## 编写HttpClientUtils类

|  |
| --- |
| 创建一个HttpClientUtils类 |
| **public class** HttpClientUtils {   *//编码格式。发送编码格式统一用UTF-8* **private static final** String ***ENCODING*** = **"UTF-8"**;   **private final static** Logger ***logger*** = LoggerFactory.*getLogger*(HttpClientUtils.**class**);   */\*\*  \* Post JOSN  \** ***@param url*** *\** ***@param json*** *\** ***@return*** *\*/* **public static** String post(String url, String json) {  ***logger***.info(**"Post URL="** + url);  ***logger***.info(**"Post Data="** + json);   StringBuffer requestText = **new** StringBuffer();   CloseableHttpResponse response = **null**;  CloseableHttpClient client = **null**;  HttpPost httpPost = **new** HttpPost(url);  StringEntity entityParams = **null**;  **try** {  entityParams = **new** StringEntity(json, **"utf-8"**);  httpPost.setEntity(entityParams);  httpPost.setHeader(**"Content-Type"**, **"type/json;charset=ISO-8859-1"**);  client = HttpClients.*createDefault*();  response = client.execute(httpPost);  **byte**[] x = EntityUtils.*toByteArray*(response.getEntity());  requestText.append(**new** String(x, **"utf-8"**));  } **catch** (Exception e) {  ***logger***.error(e.getMessage(), e);  } **finally** {  ***logger***.info(**"Request="** + requestText.toString());  }  **return** requestText.toString();  }   */\*\*  \* 基于HttpClient 4.5的通用GET方法  \*  \** ***@param url*** *提交的URL  \** ***@return*** *提交响应  \*/* **public static** String get(String url) **throws** RuntimeException {  CloseableHttpClient client = HttpClients.*createDefault*();  String responseText = **""**;  CloseableHttpResponse response = **null**;  **try** {  HttpGet get = **new** HttpGet(url);  response = client.execute(get);  HttpEntity entity = response.getEntity();  **if** (entity != **null**) {  responseText = EntityUtils.*toString*(entity, ***ENCODING***);  }  } **catch** (Exception e) {  **throw new** RuntimeException(e);  } **finally** {  **try** {  **if** (response != **null**)  response.close();  } **catch** (Exception e) {  e.printStackTrace();  }  }  **return** responseText;  }  } |

## 编写HttpCertClientUtils类

|  |
| --- |
| 创建一个HttpCertClientUtils类 |
| **public class** HttpCertClientUtils {   **private final static** Logger ***logger*** = LoggerFactory.*getLogger*(HttpCertClientUtils.**class**);   *//连接超时时间，默认10秒* **private int socketTimeout** = 10000;   *//传输超时时间，默认30秒* **private int connectTimeout** = 30000;   *//请求器的配置* **private** RequestConfig **requestConfig**;   *//HTTP请求器* **private** CloseableHttpClient **httpClient**;   */\*\*  \* 加载证书  \** ***@param certPath*** *\** ***@throws*** *IOException  \** ***@throws*** *KeyStoreException  \** ***@throws*** *UnrecoverableKeyException  \** ***@throws*** *NoSuchAlgorithmException  \** ***@throws*** *KeyManagementException  \*/* **private void** initCert(String certPath, String password) **throws** IOException, KeyStoreException, UnrecoverableKeyException, NoSuchAlgorithmException, KeyManagementException {  *//拼接证书的路径* KeyStore keyStore = KeyStore.*getInstance*(**"PKCS12"**);   *//加载本地的证书进行https加密传输* FileInputStream instream = **new** FileInputStream(**new** File(certPath));  **try** {  keyStore.load(instream, password.toCharArray()); *//加载证书密码，默认为商户ID* } **catch** (NoSuchAlgorithmException e) {  e.printStackTrace();  } **catch** (CertificateException e) {  e.printStackTrace();  } **finally** {  instream.close();  }   *// Trust own CA and all self-signed certs* SSLContext sslcontext = SSLContexts.*custom*()  .loadKeyMaterial(keyStore, password.toCharArray()) *//加载证书密码，默认为商户ID* .build();  *// Allow TLSv1 protocol only* SSLConnectionSocketFactory sslsf = **new** SSLConnectionSocketFactory(  sslcontext,  **new** String[]{**"TLSv1"**},  **null**,  SSLConnectionSocketFactory.***BROWSER\_COMPATIBLE\_HOSTNAME\_VERIFIER***);   **httpClient** = HttpClients.*custom*()  .setSSLSocketFactory(sslsf)  .build();   *//根据默认超时限制初始化requestConfig* **requestConfig** = RequestConfig.*custom*().setSocketTimeout(**socketTimeout**).setConnectTimeout(**connectTimeout**).build();   }    */\*\*  \* 通过Https往API post xml数据  \** ***@param url*** *API地址  \** ***@param xmlObj*** *要提交的XML数据对象  \** ***@param cert*** *当前目录，用于加载证书  \** ***@param password*** *证书密码 (即微信商户ID)  \** ***@return*** *\** ***@throws*** *IOException  \** ***@throws*** *KeyStoreException  \** ***@throws*** *UnrecoverableKeyException  \** ***@throws*** *NoSuchAlgorithmException  \** ***@throws*** *KeyManagementException  \*/* **public** String post(String url, String xmlObj, String cert, String password) **throws** IOException, KeyStoreException, UnrecoverableKeyException, NoSuchAlgorithmException, KeyManagementException {  *//加载证书* initCert(cert, password);  String result = **null**;  HttpPost httpPost = **new** HttpPost(url);   *//得指明使用UTF-8编码，否则到API服务器XML的中文不能被成功识别* StringEntity postEntity = **new** StringEntity(xmlObj, **"UTF-8"**);  httpPost.addHeader(**"Content-Type"**, **"text/xml"**);  httpPost.setEntity(postEntity);   *//设置请求器的配置* httpPost.setConfig(**requestConfig**);   **try** {  HttpResponse response = **httpClient**.execute(httpPost);  HttpEntity entity = response.getEntity();  result = EntityUtils.*toString*(entity, **"UTF-8"**);  } **catch** (ConnectionPoolTimeoutException e) {  ***logger***.error(**"http get throw ConnectionPoolTimeoutException(wait time out)"**);  } **catch** (SocketTimeoutException e) {  ***logger***.error(**"http get throw SocketTimeoutException"**);  } **catch** (Exception e) {  ***logger***.error(**"http get throw Exception"**);  } **finally** {  httpPost.abort();  }  **return** result;  }   } |

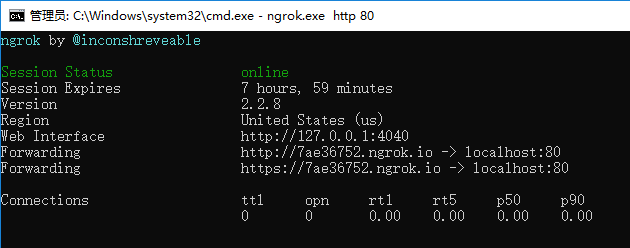
## 编写ResponseUtils类

|  |
| --- |
| 创建一个ResponseUtils类 |
| **public class** ResponseUtils {   **private final static** Logger ***logger*** = LoggerFactory.*getLogger*(ResponseUtils.**class**);   */\*\*  \* Response  \** ***@param response*** *\** ***@param responseMsg*** *\*/* **public static void** response(HttpServletResponse response, String responseMsg) {  **if** (responseMsg == **null**)  **return**;   **try** {  **byte**[] msgs = responseMsg.getBytes(**"UTF-8"**);   response.setContentType(**"text/html;charset=UTF-8"**);  response.setHeader(**"Cache-Control"**, **"no-store, max-age=0, no-cache, must-revalidate"**);  response.addHeader(**"Cache-Control"**, **"post-check=0, pre-check=0"**);  response.setHeader(**"Pragma"**, **"no-cache"**);  response.setDateHeader(**"Expires"**, 0);  response.setContentLength(msgs.**length**);  response.getOutputStream().write(msgs);  response.getOutputStream().flush();  response.getOutputStream().close();  } **catch** (Exception e) {  e.printStackTrace();  }  }   **public static** String getRequestBody(HttpServletRequest request) {  String retText = **null**;  BufferedReader reader = **null**;  **try** {  reader = **new** BufferedReader(**new** InputStreamReader(request.getInputStream(), **"ISO-8859-1"**));  String tmp = **null**;  StringBuffer htmlRet = **new** StringBuffer();  **while** ((tmp = reader.readLine()) != **null**) {  htmlRet.append(tmp).append(**"\n"**);  }  retText = **new** String(htmlRet.toString().trim().getBytes(**"ISO-8859-1"**), **"UTF-8"**);  } **catch** (Exception e) {  e.printStackTrace();  } **finally** {  **if** (reader != **null**) {  **try** {  reader.close();  } **catch** (IOException e) {  e.printStackTrace();  }  }  }  **return** retText;  }  } |

# 启动Ngrok

在DOS下，支持 **ngork http 80**即可启动将我们的电脑映射到公网中访问，用于微信支付回调。

可访问<http://127.0.0.1:4040>检查



# 微信支付功能

## 支付配置文件

|  |
| --- |
| 创建一个WXPayConfig类，用于配置微信支付环境 |
| **public class** WXPayConfig {   */\*\*  \* 获取 App ID  \*  \** ***@return*** *App ID  \*/* **public static** String getAppID() {  **return "XXXXXXXXXX"**;  }   */\*\*  \* 获取 Mch ID (安装证书的商户ID)  \*  \** ***@return*** *Mch ID  \*/* **public static** String getMchID() {  **return "XXXXXXXXXX"**;  }   */\*\*  \* 获取 API 密钥  \*  \** ***@return*** *API密钥  \*/* **public static** String getKey() {  **return "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"**;  }   */\*\*  \* 微信证书  \** ***@return*** *\*/* **public static** String getCert() {  String separator = System.*getProperty*(**"file.separator"**);  **return** WXPayConfig.*getClassPath*() + separator + **"static"** + separator + **"cert"** + separator + **"apiclient\_cert.p12"**;  }   */\*\*  \* 获取系统路径  \** ***@return*** *\*/* **public static** String getClassPath() {  String path = Thread.*currentThread*().getContextClassLoader().getResource(**""**).toString();  String temp = path.replaceFirst(**"file:/"**, **""**);  String separator = System.*getProperty*(**"file.separator"**);  String resultPath = temp.replaceAll(**"/"**, separator + separator);  **return** resultPath;  }  } |
| **application.yml** |
| *# Spring Tomcat Server* **server:  port:** 80   **pays:  http:** http://216e2635.ngrok.io |



## 统一下单

* + 1. 后台Controller

|  |
| --- |
| 创建一个WXPayController类 |
| */\*\*  \* 微信下单  \*/* @Controller **public class** WXPayController {   @Value(**"${pays.http}"**)  **private** String **payHttp**;   *// 订单回调* **private final** String **callBack** = **"/pay/callback"**;   */\*\*  \* 进入订单页面  \** ***@return*** *\*/* @GetMapping(**"/pay"**)  **public** String pay() {  **return "/pay"**;  }   */\*\*  \* 生成支付二维码  \** ***@param map*** *\** ***@param orderNo*** *\** ***@param body*** *\** ***@param price*** *注意: 在生产环境中, 价格最从好从数据库中查询商品价格, 不能通过参数转入价格, 否则容易被人篡改价格  \** ***@return*** *\*/* @PostMapping(**"/paycode"**)  **public** String paycode(ModelMap map, String orderNo, String body, Integer price) **throws** Exception {  WXPayUtils wxPayUtils = **new** WXPayUtils();   String callback = **payHttp** + **callBack**;  Map<String, String> rs = wxPayUtils.wxPay(callback, orderNo, String.*valueOf*(price),**"192.168.1.1"**, body,**"123456"**);   System.***out***.println(**"===========已获取获取订单信息==========="**);  System.***out***.println(rs);   map.put(**"code\_url"**, rs.get(**"code\_url"**));  **return "/paycode"**;  }   */\*\*  \* 支付结果通知  \** ***@return*** *\** ***@throws*** *Exception  \*/* @PostMapping(**callBack**)  **public void** notifyOrder(HttpServletRequest request, HttpServletResponse response) **throws** Exception {  *// 获取request返回的值* String retText = ResponseUtils.*getRequestBody*(request);  Map<String, String> map = WXPayUtil.*xmlToMap*(retText);   *// 检查签名信息* **boolean** sign = WXPayUtil.*isSignatureValid*(map, WXPayConfig.*getKey*());  **if** (sign) {  String wxouttradeno = map.get(**"out\_trade\_no"**);  String wxappid = map.get(**"appid"**);  String wxmchid = map.get(**"mch\_id"**);  String wxbanktype = map.get(**"bank\_type"**);  String wxcashfee = map.get(**"cash\_fee"**);  String wxtotalfee = map.get(**"total\_fee"**);  String wxissubscribe = map.get(**"is\_subscribe"**);  String wxopenid = map.get(**"openid"**);  String wxtimeend = map.get(**"time\_end"**);  String wxtradetype = map.get(**"trade\_type"**);  String wxtransactionid = map.get(**"transaction\_id"**);   *// 写入数据库* System.***out***.println(**"===========支付回调==========="**);  System.***out***.println(**"商户订单号:"** + wxouttradeno);  System.***out***.println(**"APPID:"** + wxappid);  System.***out***.println(**"商户ID:"** + wxmchid);  System.***out***.println(**"付款银行:"** + wxbanktype);  System.***out***.println(**"现金支付金额:"** + wxcashfee);  System.***out***.println(**"订单金额:"** + wxtotalfee);  System.***out***.println(**"是否关注公众账号:"** + wxissubscribe);  System.***out***.println(**"用户标识:"** + wxopenid);  System.***out***.println(**"支付完成时间:"** + wxtimeend);  System.***out***.println(**"交易类型:"** + wxtradetype); *// JSAPI、NATIVE、APP* System.***out***.println(**"微信支付订单号:"** + wxtransactionid);   *// 响应完成到微信* StringBuilder sb = **new** StringBuilder();  sb.append(**"<xml>"**);  sb.append(**"<return\_code><![CDATA[SUCCESS]]></return\_code>"**);  sb.append(**"<return\_msg><![CDATA[OK]]></return\_msg>"**);  sb.append(**"</xml>"**);   ResponseUtils.*response*(response, sb.toString());  }  }  } |

* + 1. 支付接口调用

|  |
| --- |
| 微信接口调用 |
| */\*\*  \* 微信统一下单接口  \** ***@param notify\_url*** *回调地址  \** ***@param out\_trade\_no*** *商户订单号  \** ***@param total\_fee*** *订单总金额  \** ***@param ip*** *IP  \** ***@param body*** *商品内容描述  \** ***@param openid*** *微信用户 OPENID  \** ***@return*** *\** ***@throws*** *Exception  \*/* **public** Map<String, String> wxPay(String notify\_url, String out\_trade\_no, String total\_fee, String ip, String body, String openid) **throws** Exception {  Map<String, String> mp = **new** HashMap();  mp.put(**"appid"**, WXPayConfig.*getAppID*()); *// 公众账号ID* mp.put(**"mch\_id"**, WXPayConfig.*getMchID*()); *// 商户号* mp.put(**"nonce\_str"**, WXPayUtil.*generateNonceStr*()); *// 随机字符串* mp.put(**"body"**, body); *// 商品描述* mp.put(**"out\_trade\_no"**, out\_trade\_no); *// 商户订单号* mp.put(**"total\_fee"**, total\_fee); *// 订单总金额, 单位为分* mp.put(**"spbill\_create\_ip"**, ip); *// 终端IP* mp.put(**"notify\_url"**, notify\_url); *// 异步接收微信支付结果通知的回调地址, 通知url必须为外网可访问的url, 不能携带参数* mp.put(**"trade\_type"**, **"NATIVE"**); *// 交易类型(JSAPI公众号支付/NATIVE扫码支付/APP支付)* mp.put(**"product\_id"**, out\_trade\_no); *// trade\_type=NATIVE时(即扫码支付), 此参数必传. 此参数为二维码中包含的商品ID, 商户自行定义。* String sign = WXPayUtil.*generateSignature*(mp, WXPayConfig.*getKey*());   StringBuilder sb = **new** StringBuilder();  sb.append(**"<xml>"**);  sb.append(**"<appid>"** + mp.get(**"appid"**) + **"</appid>"**);  sb.append(**"<mch\_id><![CDATA["** + mp.get(**"mch\_id"**) + **"]]></mch\_id>"**);  sb.append(**"<nonce\_str><![CDATA["** + mp.get(**"nonce\_str"**) + **"]]></nonce\_str>"**);  sb.append(**"<body><![CDATA["** + mp.get(**"body"**) + **"]]></body>"**);  sb.append(**"<out\_trade\_no><![CDATA["** + mp.get(**"out\_trade\_no"**) + **"]]></out\_trade\_no>"**);  sb.append(**"<total\_fee>"** + mp.get(**"total\_fee"**) + **"</total\_fee>"**);  sb.append(**"<spbill\_create\_ip><![CDATA["** + mp.get(**"spbill\_create\_ip"**) + **"]]></spbill\_create\_ip>"**);  sb.append(**"<notify\_url><![CDATA["** + mp.get(**"notify\_url"**) + **"]]></notify\_url>"**);  sb.append(**"<trade\_type><![CDATA["** + mp.get(**"trade\_type"**) + **"]]></trade\_type>"**);  sb.append(**"<product\_id><![CDATA["** + mp.get(**"product\_id"**) + **"]]></product\_id>"**);  sb.append(**"<sign><![CDATA["** + sign + **"]]></sign>"**);  sb.append(**"</xml>"**);   String url = **"https://api.mch.weixin.qq.com/pay/unifiedorder"**;  String str = HttpClientUtils.*post*(url, sb.toString());   Map<String, String> retmap = **new** HashMap();  **try** {  retmap = WXPayUtil.*xmlToMap*(str);  } **catch** (Exception e) {  e.printStackTrace();  }  **return** retmap; } |

* + 1. 前端

|  |
| --- |
| 前端pay.ftl |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"** />  <**title**>支付订单</**title**> </**head**> <**body**>  <**h2**>支付订单</**h2**>  <**form action="/paycode" method="post"**>  <**div**>  <**label**>商户单号</**label**>  <**input name="orderNo" value=""** />  </**div**>  <**div**>  <**label**>商品描述</**label**>  <**input name="body" value=""** />  </**div**>  <**div**>  <**label**>商品价格</**label**>  <**input name="price" value=""** />  <**span style="color**: **red**;**"**>在生产环境中, 商品价格建议通过商品ID在后查询出, 而不是通过http请求将价格做为参数传入到后台</**span**>  </**div**>  <**input type="submit"** />  </**form**>    <**div align="center" style="margin-top**: 200**px"**>  <**a href="/index"**>返回首页</**a**>  </**div**> </**body**> </**html**> |
| 前端paycode.ftl |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"** />  <**title**>微信支付</**title**>  <**script type="text/javascript" src="/jquery/jquery-3.3.1.js"**></**script**>  <**script type="text/javascript" src="/jquery-qrcode-0.14.0/jquery-qrcode-0.14.0.js"**></**script**>    <**script**>  **$**(**function** (){  **var** recommend\_code = **"${code\_url**?**if\_exists}"**; *//微信扫码关注* **$**(**"#qrcode\_div"**).qrcode({  **render** : **"canvas"**, *//设置渲染方式，有table和canvas，使用canvas方式渲染性能相对来说比较好* **text** : recommend\_code, *//扫描了二维码后的内容显示,在这里也可以直接填一个网址，扫描二维码后* **size** : 150, *//大小* **background** : **"#ffffff"** *//二维码的后景色* });  });  </**script**> </**head**> <**body**>  <**h2**>微信支付二维码</**h2**>  <**span**>**${code\_url**?**if\_exists}**</**span**>   <**div id="qrcode\_div"**>  </**div**>    <**div align="center" style="margin-top**: 200**px"**>  <**a href="/index"**>返回首页</**a**>  </**div**>  </**body**> </**html**> |

## 查询订单

* + 1. 后台Controller

|  |
| --- |
| 创建一个WXPayQueryController类 |
| */\*\*  \* 微信订单查询  \*/* @Controller **public class** WXPayQueryController {   */\*\*  \* 查询订单  \** ***@return*** *\*/* @GetMapping(**"/payquery"**)  **public** String payquery() {  **return "/payquery"**;  }   */\*\*  \* 查询订单  \** ***@param map*** *\** ***@param orderNo*** *\** ***@return*** *\*/* @PostMapping(**"/payquery"**)  **public** String payquery(ModelMap map, String orderNo) **throws** Exception {  WXPayUtils wxPayUtils = **new** WXPayUtils();  Map<String, String> rs = wxPayUtils.wxOrderQuery(orderNo);  map.put(**"payqueryMap"**, rs.toString());   System.***out***.println(**"===========订单查询结果==========="**);  System.***out***.println(rs);   **return "/payquery"**;  }  } |

* + 1. 支付接口调用

|  |
| --- |
| 微信接口调用 |
| */\*\*  \* 微信查询  \** ***@param out\_trade\_no*** *商户订单号  \** ***@return*** *\** ***@throws*** *Exception  \*/* **public** Map<String, String> wxOrderQuery(String out\_trade\_no) **throws** Exception {  Map<String, String> mp = **new** HashMap<>();  mp.put(**"appid"**, WXPayConfig.*getAppID*()); *// 公众账号ID* mp.put(**"mch\_id"**, WXPayConfig.*getMchID*()); *// 商户号  // mp.put("transaction\_id", transaction\_id); // 微信订单号 (2 S 1)* mp.put(**"out\_trade\_no"**, out\_trade\_no); *// 商户订单号 (2 S 1)* mp.put(**"nonce\_str"**, WXPayUtil.*generateNonceStr*()); *// 随机字符串* String sign = WXPayUtil.*generateSignature*(mp, WXPayConfig.*getKey*());  StringBuilder sb = **new** StringBuilder();   sb.append(**"<xml>"**);  sb.append(**"<appid>"** + mp.get(**"appid"**) + **"</appid>"**);  sb.append(**"<mch\_id><![CDATA["** + mp.get(**"mch\_id"**) + **"]]></mch\_id>"**);  sb.append(**"<out\_trade\_no><![CDATA["** + mp.get(**"out\_trade\_no"**) + **"]]></out\_trade\_no>"**);  sb.append(**"<nonce\_str>"** + mp.get(**"nonce\_str"**) + **"</nonce\_str>"**);  sb.append(**"<sign><![CDATA["** + sign + **"]]></sign>"**);  sb.append(**"</xml>"**);   String url = **"https://api.mch.weixin.qq.com/pay/orderquery"**;  String str = HttpClientUtils.*post*(url, sb.toString());   Map<String, String> retmap = **new** HashMap<>();  **try** {  retmap = WXPayUtil.*xmlToMap*(str);  } **catch** (Exception e) {  e.printStackTrace();  }  **return** retmap; } |

* + 1. 前端

|  |
| --- |
| 前端pay.js |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"** />  <**title**>查询订单</**title**> </**head**> <**body**>  <**h2**>查询订单</**h2**>  <**form action="/payquery" method="post"**>  <**div**>  <**label**>商户单号</**label**>  <**input name="orderNo" value=""** />  </**div**>  <**input type="submit"** />  </**form**>   <**h3**>查询结果</**h3**>  <**span**>**${payqueryMap**?**if\_exists}**</**span**>   <**div align="center" style="margin-top**: 200**px"**>  <**a href="/index"**>返回首页</**a**>  </**div**> </**body**> </**html**> |

## 关闭订单

* + 1. 后台Controller

|  |
| --- |
| 创建一个WXPayCloseController类 |
| */\*\*  \* 微信关闭订单  \*/* @Controller **public class** WXPayCloseController {   */\*\*  \* 打开关闭订单页面  \** ***@return*** *\*/* @GetMapping(**"/payclose"**)  **public** String payclose() {  **return "/payclose"**;  }   */\*\*  \* 执行关闭订单  \** ***@param map*** *\** ***@param orderNo*** *\** ***@return*** *\*/* @PostMapping(**"/payclose"**)  **public** String payclose(ModelMap map, String orderNo) **throws** Exception {  WXPayUtils wxPayUtils = **new** WXPayUtils();  Map<String, String> rs = wxPayUtils.wxCloseOrder(orderNo);  map.put(**"paycloseMap"**, rs.toString());   System.***out***.println(**"===========关闭订单结果==========="**);  System.***out***.println(rs);    **return "/payclose"**;  }  } |

* + 1. 支付接口调用

|  |
| --- |
| 微信接口调用 |
| */\*\*  \* 关闭订单  \** ***@param out\_trade\_no*** *商户订单号  \** ***@return*** *\** ***@throws*** *Exception  \*/* **public** Map<String, String> wxCloseOrder(String out\_trade\_no) **throws** Exception {  Map<String, String> mp = **new** HashMap<>();  mp.put(**"appid"**, WXPayConfig.*getAppID*()); *// 公众账号ID* mp.put(**"mch\_id"**, WXPayConfig.*getMchID*()); *// 商户号* mp.put(**"out\_trade\_no"**, out\_trade\_no); *// 商户订单号 (2 S 1)* mp.put(**"nonce\_str"**, WXPayUtil.*generateNonceStr*()); *// 随机字符串* String sign = WXPayUtil.*generateSignature*(mp, WXPayConfig.*getKey*());  StringBuilder sb = **new** StringBuilder();   sb.append(**"<xml>"**);  sb.append(**"<appid>"** + mp.get(**"appid"**) + **"</appid>"**);  sb.append(**"<mch\_id><![CDATA["** + mp.get(**"mch\_id"**) + **"]]></mch\_id>"**);  sb.append(**"<out\_trade\_no><![CDATA["** + mp.get(**"out\_trade\_no"**) + **"]]></out\_trade\_no>"**);  sb.append(**"<nonce\_str>"** + mp.get(**"nonce\_str"**) + **"</nonce\_str>"**);  sb.append(**"<sign><![CDATA["** + sign + **"]]></sign>"**);  sb.append(**"</xml>"**);   String url = **"https://api.mch.weixin.qq.com/pay/closeorder"**;  String str = HttpClientUtils.*post*(url, sb.toString());   Map<String, String> retmap = **new** HashMap<>();  **try** {  retmap = WXPayUtil.*xmlToMap*(str);  } **catch** (Exception e) {  e.printStackTrace();  }  **return** retmap; } |

* + 1. 前端

|  |
| --- |
| 前端pay.js |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"** />  <**title**>关闭订单</**title**> </**head**> <**body**>  <**h2**>关闭订单</**h2**>  <**form action="/payclose" method="post"**>  <**div**>  <**label**>商户单号</**label**>  <**input name="orderNo" value=""** />  </**div**>  <**input type="submit"** />  </**form**>   <**h3**>关闭结果</**h3**>  <**span**>**${paycloseMap**?**if\_exists}**</**span**>   <**div align="center" style="margin-top**: 200**px"**>  <**a href="/index"**>返回首页</**a**>  </**div**> </**body**> </**html**> |

## 申请退款

* + 1. 后台Controller

|  |
| --- |
| 创建一个WXRefundController类 |
| */\*\*  \* 微信 Controller  \*/* @Controller **public class** WXRefundController {   @Value(**"${pays.http}"**)  **private** String **payHttp**;   *// 退款回调* **private final** String **callBack** = **"/refund/callback"**;   */\*\*  \* 申请退款  \** ***@return*** *\*/* @GetMapping(**"/refund"**)  **public** String refund() {  **return "/refund"**;  }   */\*\*  \* 申请退款  \** ***@param map*** *\** ***@param orderNo*** *\** ***@param refundNo*** *商户退款单号  \** ***@param price*** *订单金额  \** ***@return*** *\*/* @PostMapping(**"/refund"**)  **public** String refund(ModelMap map, String orderNo, String refundNo, String price) **throws** Exception {  WXPayUtils wxPayUtils = **new** WXPayUtils();   String callback = **payHttp** + **callBack**;  Map<String, String> rs = wxPayUtils.wxRefund(callback, orderNo, refundNo, price, price);  map.put(**"refundMap"**, rs.toString());   System.***out***.println(**"===========申请退款结果==========="**);  System.***out***.println(rs);    **return "/refund"**;  }   */\*\*  \* 退款结果通知  \** ***@return*** *\** ***@throws*** *Exception  \*/* @PostMapping(**callBack**)  **public void** notifyRefund(HttpServletRequest request, HttpServletResponse response) **throws** Exception {  String retText = ResponseUtils.*getRequestBody*(request);  Map<String, String> rsmap = WXPayUtil.*xmlToMap*(retText);   String return\_code = rsmap.get(**"return\_code"**);  **if** (WXPayConstants.***SUCCESS***.equals(return\_code)) {  String req\_info = rsmap.get(**"req\_info"**);   String xmlStr = WXRefundUtils.*getRefundDecrypt*(req\_info, WXPayConfig.*getKey*());  Map<String, String> map = WXPayUtil.*xmlToMap*(xmlStr);   String out\_trade\_no = map.get(**"out\_trade\_no"**);  String out\_refund\_no = map.get(**"out\_refund\_no"**);   String refund\_id = map.get(**"refund\_id"**);  String refund\_fee = map.get(**"refund\_fee"**);  String settlement\_refund\_fee = map.get(**"settlement\_refund\_fee"**);  String refund\_status = map.get(**"refund\_status"**);  String success\_time = map.get(**"success\_time"**);  String refund\_recv\_accout = map.get(**"refund\_recv\_accout"**);   **if** (out\_refund\_no != **null**) {   *// 写入数据库* System.***out***.println(**"===========退款回调==========="**);  System.***out***.println(**"商户订单号:"** + out\_trade\_no);  System.***out***.println(**"商户退款单号:"** + out\_refund\_no);  System.***out***.println(**"微信退款单号:"** + refund\_id);  System.***out***.println(**"申请退款金额:"** + refund\_fee);  System.***out***.println(**"退款金额:"** + settlement\_refund\_fee); *// 退款金额=申请退款金额-非充值代金券退款金额，退款金额<=申请退款金额* System.***out***.println(**"退款状态:"** + refund\_status); *// SUCCESS-退款成功 CHANGE-退款异常 REFUNDCLOSE—退款关闭* System.***out***.println(**"退款成功时间:"** + success\_time); *// 资金退款至用户帐号的时间，格式2017-12-15 09:46:01* System.***out***.println(**"退款入账账户"** + refund\_recv\_accout);   *// 响应完成到微信* StringBuilder sb = **new** StringBuilder();  sb.append(**"<xml>"**);  sb.append(**"<return\_code><![CDATA[SUCCESS]]></return\_code>"**);  sb.append(**"<return\_msg><![CDATA[OK]]></return\_msg>"**);  sb.append(**"</xml>"**);  ResponseUtils.*response*(response, sb.toString());  }   }  }  } |

* + 1. 支付接口调用

|  |
| --- |
| 微信接口调用 |
| */\*\*  \* 申请退款  \** ***@param notify\_url*** *退款结果通知url  \** ***@param out\_trade\_no*** *商户订单号  \** ***@param out\_refund\_no*** *商户退款单号  \** ***@param total\_fee*** *订单金额  \** ***@param refund\_fee*** *退款金额  \** ***@return*** *\** ***@throws*** *Exception  \*/* **public** Map<String, String> wxRefund(String notify\_url, String out\_trade\_no, String out\_refund\_no, String total\_fee, String refund\_fee) **throws** Exception {  Map<String, String> mp = **new** HashMap<>();  mp.put(**"appid"**, WXPayConfig.*getAppID*()); *// 公众账号ID* mp.put(**"mch\_id"**, WXPayConfig.*getMchID*()); *// 商户号* mp.put(**"nonce\_str"**, WXPayUtil.*generateNonceStr*()); *// 随机字符串  // mp.put("transaction\_id", transaction\_id); // 微信订单号 (2 S 1)* mp.put(**"out\_trade\_no"**, out\_trade\_no); *// 商户订单号 (2 S 1)* mp.put(**"out\_refund\_no"**, out\_refund\_no); *// 商户退款单号* mp.put(**"total\_fee"**, total\_fee); *// 订单金额* mp.put(**"refund\_fee"**, refund\_fee); *// 退款金额* **if** (!StringUtils.*isEmpty*(notify\_url))  mp.put(**"notify\_url"**, notify\_url); *// 退款结果通知url* String sign = WXPayUtil.*generateSignature*(mp, WXPayConfig.*getKey*());  StringBuilder sb = **new** StringBuilder();  sb.append(**"<xml>"**);  sb.append(**"<appid>"** + mp.get(**"appid"**) + **"</appid>"**);  sb.append(**"<mch\_id><![CDATA["** + mp.get(**"mch\_id"**) + **"]]></mch\_id>"**);  sb.append(**"<nonce\_str>"** + mp.get(**"nonce\_str"**) + **"</nonce\_str>"**);  sb.append(**"<out\_trade\_no><![CDATA["** + mp.get(**"out\_trade\_no"**) + **"]]></out\_trade\_no>"**);  sb.append(**"<out\_refund\_no><![CDATA["** + mp.get(**"out\_refund\_no"**) + **"]]></out\_refund\_no>"**);  sb.append(**"<total\_fee><![CDATA["** + mp.get(**"total\_fee"**) + **"]]></total\_fee>"**);  sb.append(**"<refund\_fee><![CDATA["** + mp.get(**"refund\_fee"**) + **"]]></refund\_fee>"**);  **if** (!StringUtils.*isEmpty*(notify\_url))  sb.append(**"<notify\_url><![CDATA["** + mp.get(**"notify\_url"**) + **"]]></notify\_url>"**);  sb.append(**"<sign><![CDATA["** + sign + **"]]></sign>"**);  sb.append(**"</xml>"**);   String url = **"https://api.mch.weixin.qq.com/secapi/pay/refund"**;  String str = **new** HttpCertClientUtils().post(url, sb.toString(), WXPayConfig.*getCert*(), WXPayConfig.*getMchID*());   Map<String, String> retmap = **new** HashMap<>();  **try** {  retmap = WXPayUtil.*xmlToMap*(str);  } **catch** (Exception e) {  e.printStackTrace();  }  **return** retmap; } |
| **WXRefundUtils** |
| */\*\*  \* 微信退款加密字段解密  \*/* **public class** WXRefundUtils {   **public static** String getRefundDecrypt(String reqInfoSecret, String key) {  String result = **""**;  **try** {  Security.*addProvider*(**new** BouncyCastleProvider());  sun.misc.BASE64Decoder decoder = **new** sun.misc.BASE64Decoder();  **byte**[] bt = decoder.decodeBuffer(reqInfoSecret);  String b = **new** String(bt);  String md5key = WXPayUtil.*MD5*(key).toLowerCase();  SecretKey secretKey = **new** SecretKeySpec(md5key.getBytes(), **"AES"**);  Cipher cipher = Cipher.*getInstance*(**"AES/ECB/PKCS7Padding"**, **"BC"**);  cipher.init(Cipher.***DECRYPT\_MODE***, secretKey);  **byte**[] resultbt = cipher.doFinal(bt);  result = **new** String(resultbt);  } **catch** (Exception e) {  e.printStackTrace();  }  **return** result;  }  } |

* + 1. 前端

|  |
| --- |
| 前端pay.js |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"** />  <**title**>申请退款</**title**> </**head**> <**body**>  <**h2**>申请退款</**h2**>  <**form action="/refund" method="post"**>  <**div**>  <**label**>商户订单号</**label**>  <**input name="orderNo" value=""** />  </**div**>  <**div**>  <**label**>商户退单号</**label**>  <**input name="refundNo" value=""** />  <**span**>商户系统需要生成一个退货单号</**span**>  </**div**>  <**div**>  <**label**>退款金额</**label**>  <**input name="price" value=""** />  <**span**>可分批退</**span**>  </**div**>  <**input type="submit"** />  </**form**>   <**h3**>退款结果</**h3**>  <**span**>**${refundMap**?**if\_exists}**</**span**>   <**div align="center" style="margin-top**: 200**px"**>  <**a href="/index"**>返回首页</**a**>  </**div**> </**body**> </**html**> |

## 查询退款

* + 1. 后台Controller

|  |
| --- |
| 创建一个WXRefundQueryController类 |
| */\*\*  \* 微信 Controller  \*/* @Controller **public class** WXRefundQueryController {   */\*\*  \* 查询订单  \** ***@return*** *\*/* @GetMapping(**"/refundquery"**)  **public** String refundquery() {  **return "/refundquery"**;  }   */\*\*  \* 查询订单  \** ***@param map*** *\** ***@param orderNo*** *\** ***@return*** *\*/* @PostMapping(**"/refundquery"**)  **public** String refundquery(ModelMap map, String orderNo) **throws** Exception {  WXPayUtils x = **new** WXPayUtils();  Map<String, String> rs = x.wxRefundQuery(orderNo);  map.put(**"refundqueryMap"**, rs.toString());   System.***out***.println(**"===========退款查询结果==========="**);  System.***out***.println(rs);   **return "/refundquery"**;  }  } |

* + 1. 支付接口调用

|  |
| --- |
| 微信接口调用 |
| */\*\*  \* 查询退款  \** ***@param out\_trade\_no*** *商户订单号  \** ***@return*** *\** ***@throws*** *Exception  \*/* **public** Map<String, String> wxRefundQuery(String out\_trade\_no) **throws** Exception {  Map<String, String> mp = **new** HashMap<>();  mp.put(**"appid"**, WXPayConfig.*getAppID*()); *// 公众账号ID* mp.put(**"mch\_id"**, WXPayConfig.*getMchID*()); *// 商户号* mp.put(**"nonce\_str"**, WXPayUtil.*generateNonceStr*()); *// 随机字符串  // mp.put("transaction\_id", transaction\_id); // 微信订单号 (4 S 1)  // mp.put("out\_refund\_no", out\_refund\_no); // 商户退款单号 (4 S 1)  // mp.put("refund\_id", refund\_id); // 微信退款单号 (4 S 1)* mp.put(**"out\_trade\_no"**, out\_trade\_no); *// 商户订单号 (4 S 1)* String sign = WXPayUtil.*generateSignature*(mp, WXPayConfig.*getKey*());  StringBuilder sb = **new** StringBuilder();  sb.append(**"<xml>"**);  sb.append(**"<appid>"** + mp.get(**"appid"**) + **"</appid>"**);  sb.append(**"<mch\_id><![CDATA["** + mp.get(**"mch\_id"**) + **"]]></mch\_id>"**);  sb.append(**"<nonce\_str>"** + mp.get(**"nonce\_str"**) + **"</nonce\_str>"**);  sb.append(**"<out\_trade\_no><![CDATA["** + mp.get(**"out\_trade\_no"**) + **"]]></out\_trade\_no>"**);  sb.append(**"<sign><![CDATA["** + sign + **"]]></sign>"**);  sb.append(**"</xml>"**);   String url = **"https://api.mch.weixin.qq.com/pay/refundquery"**;  String str = HttpClientUtils.*post*(url, sb.toString());   Map<String, String> retmap = **new** HashMap<>();  **try** {  retmap = WXPayUtil.*xmlToMap*(str);  } **catch** (Exception e) {  e.printStackTrace();  }  **return** retmap; } |

* + 1. 前端

|  |
| --- |
| 前端pay.js |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="UTF-8"** />  <**title**>查询退款</**title**> </**head**> <**body**>  <**h2**>查询退款</**h2**>  <**form action="/refundquery" method="post"**>  <**div**>  <**label**>商户单号</**label**>  <**input name="orderNo" value=""** />  </**div**>  <**input type="submit"** />  </**form**>   <**h3**>退款查询结果</**h3**>  <**span**>**${refundqueryMap**?**if\_exists}**</**span**>   <**div align="center" style="margin-top**: 200**px"**>  <**a href="/index"**>返回首页</**a**>  </**div**> </**body**> </**html**> |