**品优购电商系统开发**

**第18章**

**微信扫码支付**

传智播客.黑马程序员

# 课程目标

目标1：二维码生成插件qrious的使用

目标2：完成微信支付开发的整体思路

目标3：完成微信支付接口（统一下单）生成支付二维码

目标4：完成微信支付接口（查询订单）查询支付状态

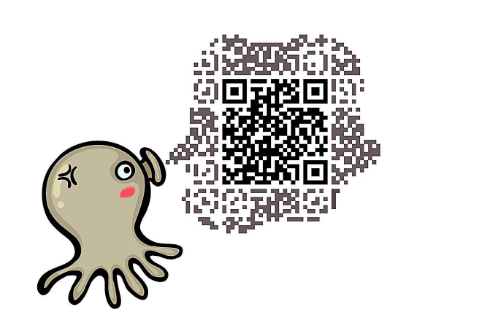
目标5：实现支付日志的生成与订单状态的修改

# 1.二维码

## 1.1什么是二维码

二维码又称QR Code，QR全称Quick Response，是一个近几年来移动设备上超流行的一种编码方式，它比传统的Bar Code条形码能存更多的信息，也能表示更多的数据类型。

二维条码/二维码（2-dimensional bar code）是用某种特定的几何图形按一定规律在平面（二维方向上）分布的黑白相间的图形记录数据符号信息的；在代码编制上巧妙地利用构成计算机内部逻辑基础的“0”、“1”比特流的概念，使用若干个与二进制相对应的几何形体来表示文字数值信息，通过图象输入设备或光电扫描设备自动识读以实现信息自动处理：它具有条码技术的一些共性：每种码制有其特定的字符集；每个字符占有一定的宽度；具有一定的校验功能等。同时还具有对不同行的信息自动识别功能、及处理图形旋转变化点。



## 1.2二维码优势

* 信息容量大, 可以容纳多达1850个大写字母或2710个数字或500多个汉字
* 应用范围广, 支持文字,声音,图片,指纹等等...
* 容错能力强, 即使图片出现部分破损也能使用
* 成本低, 容易制作

## 1.3二维码容错级别

L级（低） 7％的码字可以被恢复。

M级（中） 的码字的15％可以被恢复。

Q级（四分）的码字的25％可以被恢复。

H级（高） 的码字的30％可以被恢复。

## 1.4二维码生成插件qrious

qrious是一款基于HTML5 Canvas的纯JS二维码生成插件。通过qrious.js可以快速生成各种二维码，你可以控制二维码的尺寸颜色，还可以将生成的二维码进行Base64编码。

qrious.js二维码插件的可用配置参数如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 参数 | 类型 | 默认值 | 描述 |
| element | dom | 无 | <img/>显示二维码容器 |
| background | String | "white" | 二维码的背景颜色。 |
| foreground | String | "black" | 二维码的前景颜色。 |
| level | String | "L" | 二维码的容错级别(L, M, Q, H)。 |
| mime | String | "image/png" | 二维码输出图片时的MIME类型。 |
| size | Number | 100 | 二维码的尺寸，单位像素。 |
| value | String | 无 | 二维码的值(请求URL) |

新建index.html页面，在页面引入【资料\生成二维码\qrious框架\qrious.min.js】js文件，下面的代码即可生成一张二维码：

|  |
| --- |
| <html>  <head>  <title> 二维码 </title>  <meta http-equiv="Content-Type"  content="text/html;charset=UTF-8"/>  <script src="qrious.min.js"></script>  </head>  <body>  <img id="qrious"/>  <script type="text/javascript">  var qr = new QRious({  element : document.getElementById("qrious"),  size : 200,  level : 'H',  value : 'https://www.baidu.com'  });  </script>  </body>  </html> |

运行效果：



大家掏出手机，扫一下看看是否会看到百度呢？

# 2.微信扫码支付简介

## 2.1微信扫码支付申请

微信扫码支付是商户系统按微信支付协议生成支付二维码，用户再用微信“扫一扫”完成支付的模式。该模式适用于PC网站支付、实体店单品或订单支付、媒体广告支付等场景。

申请步骤：（了解）

**第一步：注册公众号（类型须为：服务号）**

**请根据营业执照类型选择以下主体注册：[个体工商户](http://kf.qq.com/faq/120911VrYVrA151009JB3i2Q.html" \t "http://kf.qq.com/faq/_blank)| [企业/公司](http://kf.qq.com/faq/120911VrYVrA151013MfYvYV.html" \t "http://kf.qq.com/faq/_blank)| [政府](http://kf.qq.com/faq/161220eaAJjE161220IJn6zU.html" \t "http://kf.qq.com/faq/_blank)| [媒体](http://kf.qq.com/faq/161220IFBJFv161220YnqAbQ.html" \t "http://kf.qq.com/faq/_blank)| [其他类型](http://kf.qq.com/faq/120911VrYVrA151013nYFZ7Z.html" \t "http://kf.qq.com/faq/_blank)。**

**第二步：认证公众号**

公众号认证后才可申请微信支付，认证费：300元/次。

**第三步：提交资料申请微信支付**

登录公众平台，点击左侧菜单【微信支付】，开始填写资料等待审核，审核时间为1-5个工作日内。

**第四步：开户成功，登录商户平台进行验证**

资料审核通过后，请登录联系人邮箱查收商户号和密码，并登录商户平台填写财付通备付金打的小额资金数额，完成账户验证。

**第五步：在线签署协议**

本协议为线上电子协议，签署后方可进行交易及资金结算，签署完立即生效。

本课程已经提供好“传智播客”的微信支付账号，学员无需申请。

## 2.2微信扫码支付开发文档

**微信支付接口调用的整体思路：**

**按API要求组装参数，以XML方式发送（POST）给微信支付接口（URL）,微信支付接口也是以XML方式给予响应。程序根据返回的结果（其中包括支付URL）生成二维码或判断订单状态。**

在线微信支付开发文档：

<https://pay.weixin.qq.com/wiki/doc/api/index.html>

我们在本章课程中会用到”**统一下单**”和”**订单查询**”两组API

1. **appid**：微信公众账号或开放平台APP的唯一标识【微信接口方提供】
2. **mch\_id**：商户号(配置文件中的partner)【微信接口方提供】
3. **partnerkey**：商户密钥【微信接口方提供】
4. **sign**: 数字签名, 根据微信官方提供的密钥和一套算法生成的一个加密信息, 就是为了保证交易的安全性【partnerkey + 请求参数】

## 2.3微信扫码支付SDK

微信支付提供了SDK(Software Development Kit), 大家下载后打开源码，install到本地仓库。



【资料\微信扫码支付】WxPayAPI\_JAVA\_v3.zip压缩包，解压后进入资料\微信扫码支付\java-sdk-v3目录，在该目录进行cmd，输入出以下安装命令：

mvn install -Dmaven.test.skip=true

**说明：课程配套的本地仓库已经提供jar包，所以安装SDK步骤可跳过。**

使用微信支付SDK，在maven工程中引入依赖

|  |
| --- |
| <dependency>  <groupId>com.github.wxpay</groupId>  <artifactId>wxpay-sdk</artifactId>  <version>0.0.3</version>  </dependency> |

我们主要会用到微信支付SDK的**WXPayUtil.java**工具类的以下功能：

1. 生成随机字符串

|  |
| --- |
| WXPayUtil.*generateNonceStr*() |

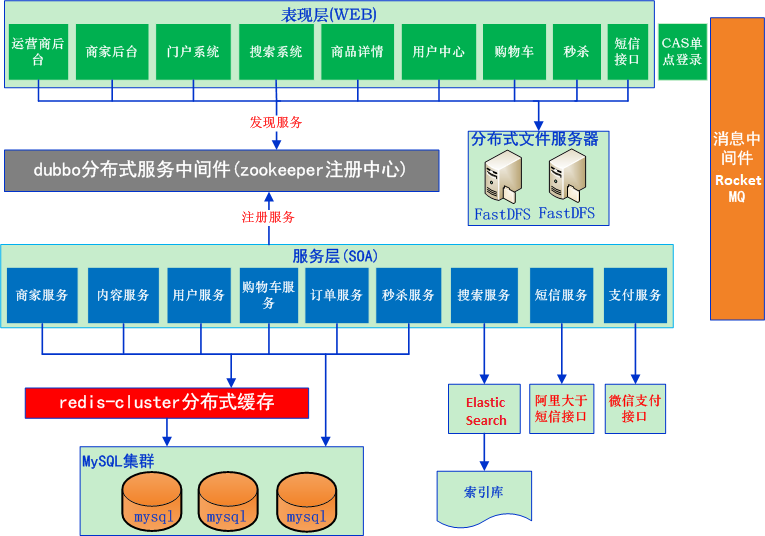
1. Map集合转换为XML字符串（自动添加签名）

|  |
| --- |
| WXPayUtil.*generateSignedXml*(param, partnerkey) |

1. XML字符串转换为Map集合

|  |
| --- |
| WXPayUtil.*xmlToMap*(result) |

# 3.搭建支付服务工程



## 3.1 pinyougou-pay-service(war)

与pinyougou-sms-service服务层的构建方式一样，拷贝pinyougou-sms-service模块中的web.xml、applicationContext-service.xml、log4j.properties。

pinyougou-pay-service/pom.xml：

|  |
| --- |
| <**properties**>  *<!-- 指定端口号 -->* <**port**>9008</**port**> </**properties**> <**dependencies**>  *<!-- wxpay-sdk -->* <**dependency**>  <**groupId**>com.github.wxpay</**groupId**>  <**artifactId**>wxpay-sdk</**artifactId**>  <**version**>0.0.3</**version**>  </**dependency**> </**dependencies**> |

pinyougou-pay-service/src/main/resources/applicationContext-service.xml：

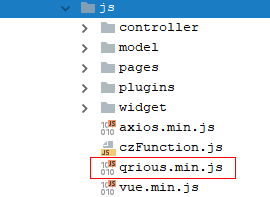
|  |
| --- |
| *<?***xml version="1.0" encoding="utf-8"***?>* <**beans xmlns="http://www.springframework.org/schema/beans"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"  xmlns:context="http://www.springframework.org/schema/context"  xsi:schemaLocation="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://code.alibabatech.com/schema/dubbo  http://code.alibabatech.com/schema/dubbo/dubbo.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd"**>  *<!-- 配置加载属性文件 -->* <**context:property-placeholder  location="classpath\*:props/\*.properties"**/>  *<!-- ############## 配置dubbo服务提供者 ############## -->  <!-- 配置当前应用的名称 -->* <**dubbo:application name="pinyougou-pay-service"**/>  *<!-- 配置zookeeper作为注册中心，注册服务地址 -->* <**dubbo:registry protocol="zookeeper" address="192.168.12.131:2181"**/>  *<!-- 用dubbo协议在20887端口暴露服务 -->* <**dubbo:protocol name="dubbo" port="20887"**/>  *<!-- 配置采用包扫描来暴露服务 -->* <**dubbo:annotation package="com.pinyougou.pay.service.impl"**/> </**beans**> |

pinyougou-pay-service/src/main/resources/props/目录下，创建weixinpay.properties

|  |
| --- |
| *# 微信公众账号或开放平台APP的唯一标识* **appid**=**wx8397f8696b538317** *# 商户账号* **partner**=**1473426802** *# 商户密钥* **partnerkey**=**T6m9iK73b0kn9g5v426MKfHQH7X8rKwb**  *# 统一下单请求地址* **unifiedorder**=**https://api.mch.weixin.qq.com/pay/unifiedorder** *# 订单查询请求地址* **orderquery**=**https://api.mch.weixin.qq.com/pay/orderquery** |

拷贝【资料\生成二维码\qrious框架\qrious.min.js】 到pinyougou-cart-web/

src/main/webapp/js/目录下。



# 4.品优购【支付二维码生成】

## 4.1需求分析&实现思路

### 4.1.1需求分析

* 在支付页面上生成支付二维码，并显示订单号和金额。
* 用户拿出手机，打开微信扫码页面上的二维码，然后在微信中完成支付。



### 4.1.2实现思路

* 我们通过HttpClientUtils工具类实现对微信支付接口的调用。
* 接口URL：https://api.mch.weixin.qq.com/pay/unifiedorder
* 具体参数参见“统一下单”API, 封装请求参数发送给统一下单的URL，返回的信息中有支付url，根据url生成支付二维码，在页面上显示订单号和支付金额。

## 4.2前端代码

### 4.2.1页面

修改pinyougou-cart-web/src/main/webapp/order/pay.html，引入js：

|  |
| --- |
| <**script src="/js/vue.min.js"**></**script**> <**script src="/js/axios.min.js"**></**script**> <**script src="/js/controller/payController.js"**></**script**> <**script src="/js/qrious.min.js"**></**script**> |

pay.html显示登录用户名(21行)：

|  |
| --- |
| <**ul class="fl"**>  <**li class="f-item"**>  {{**loginName**}}，品优购欢迎您！**&nbsp;** <**a :href="'http://sso.pinyougou.com/logout?service=' + redirectUrl"**>退出</**a**>  </**li**> </**ul**> |

pay.html设置二维码图片的ID(66行)：

|  |
| --- |
| <**div class="fl sao"**>  <**div class="fl code"**>  <**img id="qrious"**/>  <**div class="saosao"**>  <**p**>请使用微信扫一扫</**p**>  <**p**>扫描二维码支付</**p**>  </**div**>  </**div**>  <**div class="fl phone"**></**div**> </**div**> |

pay.html显示支付交易号(55行)：

|  |
| --- |
| <**h4 class="fl tit-txt"**>  <**span class="success-icon"**></**span**>  <**span class="success-info"**>订单提交成功，请您及时付款！订单号：{{**outTradeNo**}}</**span**> </**h4**> |

pay.html显示金额(58行)：

|  |
| --- |
| <**span class="fr"**>  <**em class="sui-lead"**>应付金额：</**em**>  <**em class="orange money"**>￥{{**money**}}</**em**>元 </**span**> |

### 4.2.2控制器层

pinyougou-cart-web\src\main\webapp\js\controller\payController.js

|  |
| --- |
| *// 窗口加载完* ***window***.onload = **function** () {  **var** vue = **new Vue**({  **el** : **'#app'**, *// 元素绑定* **data** : { *// 数据模型* **loginName** : **''**, *// 登录用户名* **redirectUrl** : **''**, *// 重定向URL* **outTradeNo** : **''**, *// 订单交易号* **money** : 0 *// 交易总金额* },  **methods** : { *// 操作方法  // 加载用户* loadUsername : **function** () {  *// 定义重定向URL* **this**.**redirectUrl** = ***window***.*encodeURIComponent*(***location***.href);  *// 获取登录用户名* **axios**.get(**"/user/showName"**).then(**function**(response){  vue.**loginName** = response.**data**.**loginName**;  });  },  *// 生成微信支付二维码* genPayCode : **function** () {  **axios**.get(**"/order/genPayCode"**).then(**function**(response){  */\*\* 获取金额(转化成元) \*/* vue.**money** = (response.**data**.totalFee / 100).toFixed(2);  */\*\* 获取订单交易号 \*/* vue.**outTradeNo**= response.**data**.**outTradeNo**;  */\*\* 生成二维码 \*/* **var** qr = **new QRious**({  **element** : ***document***.getElementById(**'qrious'**),  **size** : 250,  **level** : **'H'**,  **value** : response.**data**.codeUrl  });  });  }  },  created : **function** () { *// 创建生命周期* **this**.loadUsername();  **this**.genPayCode();  }  }); }; |

## 4.3后端代码

### 4.3.1控制器层

pinyougou-cart-web/src/main/java/com/pinyougou/cart/controller/OrderController.java，定义方法：

|  |
| --- |
| */\*\* 微信支付服务接口 \*/* @Reference(timeout=10000) **private** WeixinPayService **weixinPayService**;  */\*\* 生成微信支付二维码 \*/* @GetMapping(**"/genPayCode"**) **public** Map<String, String> genPayCode(){  */\*\* 创建分布式id生成器 \*/* IdWorker idworker = **new** IdWorker();  */\*\* 调用生成微信支付二维码服务方法 \*/* **return weixinPayService**.genPayCode(  String.*valueOf*(idworker.nextId()), **"1"**); } |

说明：这里我们订单号通过分布式ID生成器生成，金额暂时写死，后续开发我们再对接业务系统得到订单号和金额。

### 4.3.2服务接口层

pinyougou-interface\src\main\java\com\pinyougou\service\包下，创建微信支付服务接口WeixinPayService.java

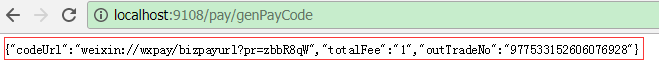
|  |
| --- |
| */\*\* 微信支付服务接口 \*/* **public interface** WeixinPayService {  */\*\*  \* 生成微信支付二维码  \** ***@param outTradeNo*** *订单交易号  \** ***@param totalFee*** *金额(分)  \** ***@return Map****集合**\*/* Map<String, String> genPayCode(String outTradeNo,  String totalFee); } |

### 4.3.3服务实现层

pinyougou-pay-service\src\main\java\com\pinyougou\pay\service\impl\包下，创建WeixinPayServiceImpl.java微信支付服务接口实现类

|  |
| --- |
| */\*\* 微信支付服务接口实现 \*/* @Service **public class** WeixinPayServiceImpl **implements** WeixinPayService {  */\*\* 微信公众号 \*/* @Value(**"${appid}"**)  **private** String **appid**;  */\*\* 商户账号 \*/* @Value(**"${partner}"**)  **private** String **partner**;  */\*\* 商户密钥 \*/* @Value(**"${partnerkey}"**)  **private** String **partnerkey**;  */\*\* 统一下单请求地址 \*/* @Value(**"${unifiedorder}"**)  **private** String **unifiedorder**;  */\*\*  \* 生成微信支付二维码  \** ***@param outTradeNo*** *订单交易号  \** ***@param totalFee*** *金额(分)  \** ***@return*** *集合  \*/* **public** Map<String, String> genPayCode(String outTradeNo,  String totalFee){  */\*\* 创建Map集合封装请求参数 \*/* Map<String, String> param = **new** HashMap<>();  */\*\* 公众号 \*/* param.put(**"appid"**, **appid**);  */\*\* 商户号 \*/* param.put(**"mch\_id"**, **partner**);  */\*\* 随机字符串 \*/* param.put(**"nonce\_str"**, WXPayUtil.*generateNonceStr*());  */\*\* 商品描述 \*/* param.put(**"body"**, **"品优购"**);  */\*\* 商户订单交易号 \*/* param.put(**"out\_trade\_no"**, outTradeNo);  */\*\* 总金额（分） \*/* param.put(**"total\_fee"**,totalFee);  */\*\* IP地址 \*/* param.put(**"spbill\_create\_ip"**, **"127.0.0.1"**);  */\*\* 回调地址(随意写) \*/* param.put(**"notify\_url"**, **"http://test.itcast.cn"**);  */\*\* 交易类型 \*/* param.put(**"trade\_type"**, **"NATIVE"**);  **try** {  */\*\* 根据商户密钥签名生成XML格式请求参数 \*/* String xmlParam = WXPayUtil.*generateSignedXml*(param, **partnerkey**);  System.***out***.println(**"请求参数："** + xmlParam);  */\*\* 创建HttpClientUtils对象发送请求 \*/* HttpClientUtils client = **new** HttpClientUtils(**true**);  */\*\* 发送请求，得到响应数据 \*/* String result = client.sendPost(**unifiedorder**, xmlParam);  System.***out***.println(**"响应数据："** + result);  */\*\* 将响应数据XML格式转化成Map集合 \*/* Map<String,String> resultMap = WXPayUtil.*xmlToMap*(result);  */\*\* 创建Map集合封装返回数据 \*/* Map<String,String> data = **new** HashMap<>();  */\*\* 支付地址(二维码中的URL) \*/* data.put(**"codeUrl"**, resultMap.get(**"code\_url"**));  */\*\* 总金额 \*/* data.put(**"totalFee"**, totalFee);  */\*\* 订单交易号 \*/* data.put(**"outTradeNo"**, outTradeNo);  **return** data;  } **catch** (Exception ex) {  **throw new** RuntimeException(ex);  }  } } |

浏览器测试：



# 5.品优购【检测支付状态】

## 5.1需求分析&实现思路

### 5.1.1需求分析

* 当用户支付成功后跳转到成功页面



* 当支付出现异常时跳转到错误页面



### 5.1.2实现思路

* 我们通过HttpClientUtils工具类实现对微信订单查询接口的调用。
* 接口URL：https://api.mch.weixin.qq.com/pay/orderquery
* 具体参数参见“订单查询”API, 我们在前端页面采用js定时器间隔3秒发送请求查询订单，当后台返回状态为1时，前端自动跳转到支付成功页面。当后台返回状态为3时，前端自动跳转到支付失败页面。
* 需用到js的定时器setInterval()方法。

## 5.2前端代码

### 5.2.1控制器层

pinyougou-cart-web\src\main\webapp\js\controller\payController.js，修改methods的genPayCode方法，开启定时器：

|  |
| --- |
| *// 生成微信支付二维码* genPayCode : **function** () {  **axios**.get(**"/order/genPayCode"**).then(**function**(response){  */\*\* 获取金额(转化成元) \*/* vue.**money** = (response.**data**.totalFee / 100).toFixed(2);  */\*\* 获取订单交易号 \*/* vue.**outTradeNo**= response.**data**.**outTradeNo**;  */\*\* 生成二维码 \*/* **var** qr = **new QRious**({  **element** : ***document***.getElementById(**'qrious'**),  **size** : 250,  **level** : **'H'**,  **value** : response.**data**.codeUrl  });   */\*\*  \* 开启定时器  \* 第一个参数：调用的函数  \* 第二个参数：时间毫秒数(3000毫秒也就是3秒)  \* \*/* **var** timer = *setInterval*(**function**() {  */\*\* 发送请求，查询支付状态 \*/* **axios**.get(**"/order/queryPayStatus?outTradeNo="** + vue.**outTradeNo**).then(**function**(response){  **if**(response.**data**.**status** == 1){*// 支付成功  /\*\* 取消定时器 \*/* ***window***.clearInterval(timer);  ***location***.href = **"/order/paysuccess.html"**;  }  **if**(response.**data**.**status** == 3){*// 支付失败  /\*\* 取消定时器 \*/* ***window***.clearInterval(timer);  ***location***.href = **"/order/payfail.html"**;  }  });  }, 3000);  }); } |

## 5.3后端代码

### 5.3.1控制器层

pinyougou-cart-web/src/main/java/com/pinyougou/cart/controller/OrderController.java，新增方法：

|  |
| --- |
| */\*\* 查询支付状态 \*/* @GetMapping(**"/queryPayStatus"**) **public** Map<String,Integer> queryPayStatus(String outTradeNo){  Map<String,Integer> data = **new** HashMap<>();  data.put(**"status"**, 3);  **try**{  *// 调用查询订单接口* Map<String,String> resMap = **weixinPayService**.queryPayStatus(outTradeNo);  **if** (resMap != **null** && resMap.size() > 0){  *// 判断是否支付成功* **if** (**"SUCCESS"**.equals(resMap.get(**"trade\_state"**))){  data.put(**"status"**, 1);  }  **if** (**"NOTPAY"**.equals(resMap.get(**"trade\_state"**))){  data.put(**"status"**, 2);  }  }  }**catch** (Exception ex){  ex.printStackTrace();  }  **return** data; } |

### 5.3.2服务接口层

pinyougou-interface\src\main\java\com\pinyougou\service\WeixinPayService.java，新增方法：

|  |
| --- |
| */\*\*  \* 查询支付状态  \** ***@param outTradeNo*** *订单交易号  \** ***@return Map****集合  \*/* Map<String, String> queryPayStatus(String outTradeNo); |

### 5.3.3服务实现层

pinyougou-pay-service\src\main\java\com\pinyougou\pay\service\impl\WeixinPayServiceImpl.java：

|  |
| --- |
| */\*\* 订单查询请求地址 \*/* @Value(**"${orderquery}"**) **private** String **orderquery**;  */\*\*  \* 查询支付状态  \** ***@param outTradeNo*** *订单交易号  \** ***@return*** *集合  \*/* **public** Map<String, String> queryPayStatus(String outTradeNo) {  */\*\* 创建Map集合封装请求参数 \*/* Map<String, String> param = **new** HashMap<>(5);  */\*\* 公众号 \*/* param.put(**"appid"**, **appid**);  */\*\* 商户号 \*/* param.put(**"mch\_id"**, **partner**);  */\*\* 订单交易号 \*/* param.put(**"out\_trade\_no"**, outTradeNo);  */\*\* 随机字符串 \*/* param.put(**"nonce\_str"**, WXPayUtil.*generateNonceStr*());  **try** {  */\*\* 根据商户密钥签名生成XML格式请求参数 \*/* String xmlParam = WXPayUtil.*generateSignedXml*(param, **partnerkey**);  System.***out***.println(**"请求参数："** + xmlParam);  */\*\* 创建HttpClientUtils对象发送请求 \*/* HttpClientUtils client = **new** HttpClientUtils(**true**);  */\*\* 发送请求，得到响应数据 \*/* String result = client.sendPost(**orderquery**, xmlParam);  System.***out***.println(**"响应数据："** + result);  */\*\* 将响应数据XML格式转化成Map集合 \*/* **return** WXPayUtil.*xmlToMap*(result);  } **catch** (Exception ex) {  **throw new** RuntimeException(ex);  } } |

# 6.支付成功页面【显示金额】

## 6.1问题分析

现在我们支付成功页面显示的是固定的值，怎么显示真正的支付金额呢？我们这里可以使用页面传参来解决。

## 6.2前端代码

### 6.2.1控制器层

pinyougou-cart-web\src\main\webapp\js\controller\payController.js跳转到支付成功页面时带支付总金额参数：

|  |
| --- |
| */\*\*  \* 开启定时器  \* 第一个参数：调用的函数  \* 第二个参数：时间毫秒数(3000毫秒也就是3秒)  \* \*/* **var** timer = *setInterval*(**function**() {  */\*\* 发送请求，查询支付状态 \*/* **axios**.get(**"/order/queryPayStatus?outTradeNo="** + vue.**outTradeNo**).then(**function**(response){  **if**(response.**data**.**status** == 1){*// 支付成功  /\*\* 取消定时器 \*/* ***window***.clearInterval(timer);  ***location***.href = **"/order/paysuccess.html?money="** + vue.**money**;  }  **if**(response.**data**.**status** == 3){*// 支付失败  /\*\* 取消定时器 \*/* ***window***.clearInterval(timer);  ***location***.href = **"/order/payfail.html"**;  }  }); }, 3000); |

### 6.2.2页面

修改pinyougou-cart-web/src/main/webapp/order/paysuccess.html页面

|  |
| --- |
| <**script src="/js/vue.min.js"**></**script**> <**script src="/js/axios.min.js"**></**script**> <**script src="/js/location.js"**></**script**> |

paysuccess.html编写js代码：

|  |
| --- |
| <**script type="text/javascript"**>  ***window***.onload = **function** () {  **var** vue = **new Vue**({  **el** : **'#app'**, *// 元素绑定* **data** : { *// 数据模型* **loginName** : **''**, *// 登录用户名* **redirectUrl** : **''**, *// 重定向URL* **money** : 0 *// 金额* },  **methods** : { *// 操作方法  // 加载用户* loadUsername : **function** () {  *// 定义重定向URL* **this**.**redirectUrl** = ***window***.*encodeURIComponent*(***location***.href);  *// 获取登录用户名* **axios**.get(**"/user/showName"**).then(**function**(response){  vue.**loginName** = response.**data**.**loginName**;  });  },  *// 获取金额* getMoney : **function** () {  **this**.**money** = **this**.getUrlParam(**"money"**);  }  },  created : **function** () { *// 创建生命周期* **this**.loadUsername();  **this**.getMoney();  }  });  } </**script**> |

paysuccess.html显示登录用户名(50行)：

|  |
| --- |
| <**ul class="fl"**>  <**li class="f-item"**>  {{**loginName**}}，品优购欢迎您！**&nbsp;** <**a :href="'http://sso.pinyougou.com/logout?service=' + redirectUrl"**>退出</**a**>  </**li**> </**ul**> |

paysuccess.html用表达式显示金额(87行)：

|  |
| --- |
| <**p**>支付金额：￥{{money}}元</**p**> |

# 7.品优购【支付日志】

## 7.1需求分析

现在我们系统还有三个问题需要解决：

1. 系统中无法查询到支付记录。
2. 生成微信支付二维码的订单交易号与金额不能写死。
3. 支付成功后，支付状态【支付成功|支付失败】，订单状态【已付款｜未付款】没有改变。

## 7.2表结构分析

**tb\_paylog 支付日志表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 长度 | 含义 |
| out\_trade\_no | varchar | 30 | 支付订单号 |
| create\_time | datatime |  | 创建时间 |
| pay\_time | datatime |  | 支付完成时间 |
| total\_fee | bigint |  | 支付金额（分） |
| transaction\_id | varchar | 30 | 交易流水号 |
| trade\_state | varchar | 1 | 交易状态 |
| pay\_type | varchar | 1 | 支付类型：  1:微信  2:支付宝  3:网银 |
| order\_list | varchar | 200 | 订单表ID串，用逗号分隔 |

## 7.3插入支付日志

### 7.3.1实现思路

* 在用户下订单时，判断如果为微信支付，就往支付日志表添加一条记录，信息包括支付总金额、订单ID（多个）、用户ID 、下单时间等信息，支付状态为0（未支付）。
* 生成的支付日志对象放入redis中，以用户ID作为key。

### 7.3.2服务实现层

pinyougou-order-service/src/main/java/com.pinyougou.order.service.impl/OrderServiceImpl.java：

|  |
| --- |
| @Autowired **private** PayLogMapper **payLogMapper**; */\*\*  \* 保存订单  \** ***@param order*** *订单实体  \*/* **public void** save(Order order){  **try**{  */\*\* 根据用户名获取Redis中购物车数据 \*/* List<Cart> carts = (List<Cart>) **redisTemplate** .boundValueOps(order.getUserId()).get();   */\*\* 定义订单ID集合(一次支付对应多个订单) \*/* List<String> orderIdList = **new** ArrayList<>();  */\*\* 定义多个订单支付的总金额（元） \*/* **double** totalMoney = 0;   */\*\* 迭代购物车数据 \*/* **for** (Cart cart : carts){  */\*\* 生成订单主键id \*/* **long** orderId = **idWorker**.nextId();  ......  */\*\* 保存数据到订单表 \*/* **orderMapper**.insertSelective(order1);   */\*\* 记录订单id\*/* orderIdList.add(String.*valueOf*(orderId));  */\*\* 记录总金额 \*/* totalMoney += money;  }  */\*\* 判断是否为微信支付 \*/* **if**(**"1"**.equals(order.getPaymentType())){  */\*\* 创建支付日志对象 \*/* PayLog payLog = **new** PayLog();  */\*\* 生成订单交易号 \*/* String outTradeNo = String.*valueOf*(**idWorker**.nextId());  */\*\* 设置订单交易号 \*/* payLog.setOutTradeNo(outTradeNo);  */\*\* 创建时间 \*/* payLog.setCreateTime(**new** Date());  */\*\* 支付总金额(分) \*/* payLog.setTotalFee((**long**)(totalMoney \* 100));  */\*\* 用户ID \*/* payLog.setUserId(order.getUserId());  */\*\* 支付状态 \*/* payLog.setTradeState(**"0"**);  /\*\* 订单号集合，逗号分隔 \*/  String ids = orderIdList.toString().replace("[", "")  .replace("]", "").replace(" ","");  /\*\* 设置订单号 \*/payLog.setOrderList(ids);  */\*\* 支付类型 \*/* payLog.setPayType(**"1"**);  */\*\* 往支付日志表插入数据 \*/* **payLogMapper**.insertSelective(payLog);  */\*\* 存入缓存 \*/* **redisTemplate.boundValueOps("payLog\_" + order.getUserId()).set(payLog);**  }  */\*\* 删除该用户购物车中的数据 \*/* **redisTemplate**.delete(order.getUserId());  }**catch**(Exception ex){  **throw new** RuntimeException(ex);  } } |

## 7.4完善支付二维码生成

### 7.4.1实现思路

* 在生成微信支付二维码时从Redis中读取支付日志对象，得到订单交易号和金额。

### 7.4.2控制器层

pinyougou-cart-web/src/main/java/com/pinyougou/cart/controller/OrderController.java，修改genPayCode方法

|  |
| --- |
| */\*\* 生成微信支付二维码 \*/* @GetMapping(**"/genPayCode"**) **public** Map<String, String> genPayCode(HttpServletRequest request){  */\*\* 获取登录用户名 \*/* String userId = request.getRemoteUser();  */\*\* 从Redis查询支付日志 \*/* PayLog payLog = **orderService**.findPayLogFromRedis(userId);  */\*\* 调用生成微信支付二维码服务方法 \*/* **return weixinPayService**.genPayCode(payLog.getOutTradeNo(),  String.*valueOf*(payLog.getTotalFee())); } |

### 7.4.3服务接口层

pinyougou-interface\src\main\java\com\pinyougou\service\OrderService.java，新增方法：

|  |
| --- |
| */\*\* 根据用户查询支付日志 \*/* PayLog findPayLogFromRedis(String userId); |

### 7.4.4服务实现层

pinyougou-order-service\src\main\java\com\pinyougou\order\service\impl\OrderServiceImpl.java：

|  |
| --- |
| */\*\* 根据用户查询支付日志 \*/* **public** PayLog findPayLogFromRedis(String userId){  **try** {  **return** (PayLog) **redisTemplate** .boundValueOps(**"payLog\_"** + userId).get();  }**catch**(Exception ex){  **throw new** RuntimeException(ex);  } } |

## 7.5.修改支付状态&订单状态

### 7.5.1实现思路

当用户支付成功后，修改支付日志的支付状态为1（已支付），并记录微信支付传递给我们的交易流水号。根据订单ID（多个）修改订单的状态为2（已付款）

主要功能说明：

1. 修改支付日志状态
2. 修改关联的订单状态
3. 清除缓存中的支付日志对象

### 7.5.2控制器层

pinyougou-cart-web/src/main/java/com/pinyougou/cart/controller/OrderController.java，修改queryPayStatus方法：

|  |
| --- |
| */\*\* 查询支付状态 \*/* @GetMapping(**"/queryPayStatus"**) **public** Map<String,Integer> queryPayStatus(String outTradeNo){  Map<String,Integer> data = **new** HashMap<>();  data.put(**"status"**, 3);  **try**{  *// 调用查询订单接口* Map<String,String> resMap = **weixinPayService**.queryPayStatus(outTradeNo);  **if** (resMap != **null** && resMap.size() > 0){  *// 判断是否支付成功* **if** (**"SUCCESS"**.equals(resMap.get(**"trade\_state"**))){  */\*\* 修改订单状态 \*/* **orderService**.updateOrderStatus(outTradeNo,  resMap.get(**"transaction\_id"**));  data.put(**"status"**, 1);  }  **if** (**"NOTPAY"**.equals(resMap.get(**"trade\_state"**))){  data.put(**"status"**, 2);  }  }  }**catch** (Exception ex){  ex.printStackTrace();  }  **return** data; } |

### 7.5.3服务接口层

pinyougou-interface\src\main\java\com\pinyougou\service\OrderService.java，新增方法：

|  |
| --- |
| */\*\*  \* 修改订单状态  \** ***@param outTradeNo*** *订单交易号  \** ***@param transactionId*** *微信交易流水号  \*/* **void** updateOrderStatus(String outTradeNo, String transactionId); |

### 7.5.4服务实现层

pinyougou-order-service\src\main\java\com\pinyougou\order\service\impl\OrderServiceImpl.java：

|  |
| --- |
| */\*\*  \* 修改订单状态  \** ***@param outTradeNo*** *订单交易号  \** ***@param transactionId*** *微信交易流水号  \*/* **public void** updateOrderStatus(String outTradeNo, String transactionId){  **try** {  */\*\* 修改支付日志状态 \*/* PayLog payLog = **payLogMapper**.selectByPrimaryKey(outTradeNo);  payLog.setPayTime(**new** Date());  payLog.setTradeState(**"1"**); *// 已支付* payLog.setTransactionId(transactionId);*// 交易流水号* **payLogMapper**.updateByPrimaryKeySelective(payLog);   */\*\* 修改订单状态 \*/* String[] orderIds = payLog.getOrderList().split(**","**); *// 订单号列表  /\*\* 循环订单号数组 \*/* **for**(String orderId : orderIds){  Order order = **new** Order();  order.setOrderId(Long.*valueOf*(orderId));  order.setPaymentTime(**new** Date()); *// 支付时间*  order.setStatus(**"2"**); *// 已付款* **orderMapper**.updateByPrimaryKeySelective(order);  }  */\*\* 清除redis缓存数据 \*/* **redisTemplate**.delete(**"payLog\_"** + payLog.getUserId());  }**catch**(Exception ex){  **throw new** RuntimeException(ex);  } } |

# 8.显示登录用户名

## 8.1支付失败页面

修改pinyougou-cart-web/src/main/webapp/order/payfail.html：

|  |
| --- |
| <**script src="/js/vue.min.js"**></**script**> <**script src="/js/axios.min.js"**></**script**> <**script type="text/javascript"**>  ***window***.onload = **function** () {  **var** vue = **new Vue**({  **el** : **'#app'**, *// 元素绑定* **data** : { *// 数据模型* **loginName** : **''**, *// 登录用户名* **redirectUrl** : **''** *// 重定向URL* },  **methods** : { *// 操作方法  // 加载用户* loadUsername : **function** () {  *// 定义重定向URL* **this**.**redirectUrl** = ***window***.*encodeURIComponent*(***location***.href);  *// 获取登录用户名* **axios**.get(**"/user/showName"**).then(**function**(response){  vue.**loginName** = response.**data**.**loginName**;  });  }  },  created : **function** () { *// 创建生命周期* **this**.loadUsername();  }  });  } </**script**>  <**ul class="fl"**>  <**li class="f-item"**>  {{***loginName***}}，品优购欢迎您！**&nbsp;** <**a :href="'http://sso.pinyougou.com/logout?service=' + redirectUrl"**>退出</**a**>  </**li**> </**ul**> |