**品优购电商系统开发**

**第17章**

**跨域解决方案&提交订单**

传智播客.黑马程序员

# 课程目标

目标1：掌握跨域请求CORS

目标2：完成结算页收货人地址选择功能

目标3：完成结算页支付方式选择

目标4：完成结算页商品清单功能

目标5：保存订单功能

# 1.商品详细页跨域请求【CORS】

## 1.1需求分析

从商品详细页点击“加入购物车”按钮，将当前商品加入购物车，并跳转到购物车页面。

## 1.2什么是跨域请求



JavaScript在不同域名之间请求数据，域名不同或者端口不同的请求都叫跨域请求。由于JavaScript同源策略的限制，处于安全的考虑；js只能获取自己域名下的资源，不能跨域请求资源。

注意：只要**协议、域名、端口**有任何一个不同，都被当作是不同的域。

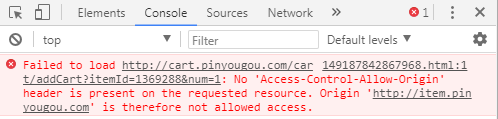
## 1.3 JS跨域请求测试

pinyougou-item-web\src\main\webapp\js\controller\itemController.js，修改methods中addToCart方法：

|  |
| --- |
| *// 加入购物车事件绑定* addToCart : **function** () {  *// 发送跨域异步请求* **axios**.get(**"http://cart.pinyougou.com/cart/addCart?itemId="** + **this**.**sku**.**id** + **"&num="** + **this**.**num**).then(**function**(response){  **if** (response.**data**){  */\*\* 跳转到购物车页面 \*/* ***location***.href=**'http://cart.pinyougou.com/cart.html'**;  }**else**{  *alert*(**"请求失败！"**);  }  }); }, |

发现无法Ajax跨域请求：

No 'Access-Control-Allow-Origin' header is present on the requested resource. Origin 'http://item.pinyougou.com' is therefore not allowed access.



## 1.4跨域请求解决方案

### 1.4.1方案一【JSONP】

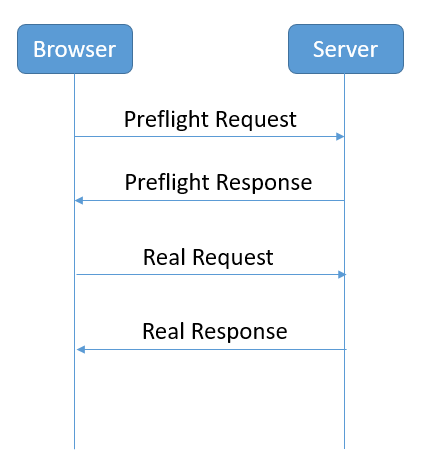
JSONP(JSON with Padding)是JSON的一种“使用模式”，可用于解决主流浏览器的跨域数据访问的问题。

具体实现原理参考：【资料\**JSONP跨域请求原理.pdf**】

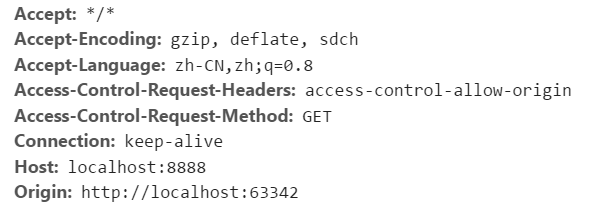
### 1.4.2方案二【CORS】

CORS是一个W3C标准，全称是"跨域资源共享"（Cross-origin Resource Sharing）。CORS需要浏览器和服务器同时支持。目前，所有浏览器都支持该功能，IE浏览器不能低于IE10。

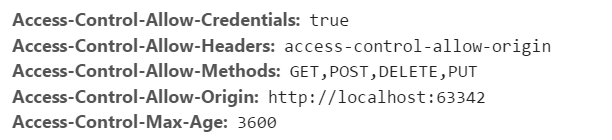
它允许浏览器向跨源服务器，发出XMLHttpRequest请求，从而克服了AJAX只能同源使用的限制。整个CORS通信过程，都是浏览器自动完成，不需要用户参与。对于开发者来说，CORS通信与同源的AJAX通信没有差别，代码完全一样。浏览器一旦发现AJAX请求跨源，就会自动添加一些附加的头信息，有时还会多出一次附加的请求，但用户不会有感觉。因此，实现CORS通信的关键是服务器。只要服务器实现了CORS接口，就可以跨源通信。请求过程如下图：



Preflight Request(预检请求)：



然后服务器端给我们返回一个Preflight Response(预检响应)



## 1.5 CORS解决方案

### 1.5.1编码设置响应头

a.修改pinyougou-cart-web/src/main/java/com.pinyougou.cart.

controller包下CartController.java类中的addCart方法，添加下面两句代码：

|  |
| --- |
| */\*\* 设置允许访问的域名 \*/* **response**.setHeader(**"Access-Control-Allow-Origin"**,**"http://item.pinyougou.com"**); */\*\* 设置允许操作Cookie \*/* **response**.setHeader(**"Access-Control-Allow-Credentials"**,**"true"**); |

Access-Control-Allow-Origin是**[HTML5](http://lib.csdn.net/base/html5" \o "HTML5知识库" \t "http://blog.csdn.net/relucent/article/details/_blank)**中定义的一种解决资源跨域的策略。

它是通过服务器端返回带有Access-Control-Allow-Origin标识的Response header，用来解决资源的跨域权限问题。

使用方法，在response添加 Access-Control-Allow-Origin：

Access-Control-Allow-Origin: item.pinyougou.com

也可以设置为 \* 表示该资源谁都可以用

b.修改pinyougou-item-web/src/main/webapp/js/controller/目录下的itemController.js：

|  |
| --- |
| *// 加入购物车事件绑定* addToCart : **function** () {  *// 发送跨域异步请求* **axios**.get(**"http://cart.pinyougou.com/cart/addCart?itemId="** + **this**.**sku**.**id** + **"&num="** + **this**.**num**,**{"withCredentials":true}**)  .then(**function**(response){  **if** (response.**data**){  */\*\* 跳转到购物车页面 \*/* ***location***.href=**'http://cart.pinyougou.com/cart.html'**;  }**else**{  *alert*(**"请求失败！"**);  }  }); }, |

注意：开发者必须在AJAX请求中设置withCredentials属性为true，同意浏览器发送Cookie。这样才能操作cart.pinyougou.com域名下的Cookie。即使服务器端同意，也不能访问。

### 1.5.2 SpringMVC跨域注解

SpringMVC的版本在4.2或以上版本，可以使用注解实现跨域，我们只需要在需要跨域的方法上添加注解@CrossOrigin即可(CartController.java)：

|  |
| --- |
| */\*\* 添加商品到购物车 \*/* @GetMapping(**"/addCart"**) @CrossOrigin(origins=**"http://item.pinyougou.com"**,  allowCredentials=**"true"**) **public boolean** addCart(Long itemId, Integer num){  } |

allowCredentials="true" 允许访问Cookie，可以缺省。

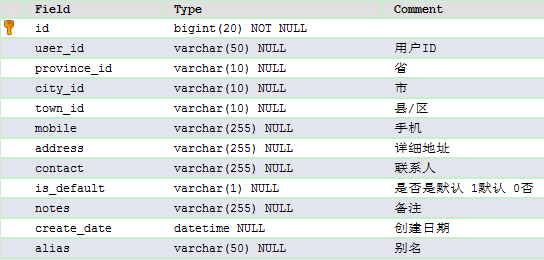
# 2.结算页【收件人地址选择】

## 2.1需求分析

在结算页实现收件人地址选择功能，查询当前登录用户所有的收件地址。

## 2.2数据库结构分析

tb\_address 用户地址表：



## 2.3静态资源

拷贝【资料/order文件夹】至pinyougou-cart-web/src/main/webapp/目录下面

## 2.4收件人地址列表

### 2.4.1前端代码

跳转到结算确认页面，修改pinyougou-cart-web/src/main/webapp/的cart.html(147行)：

|  |
| --- |
| <**div class="sumbtn"**>  <**a class="sum-btn" href="/order/getOrderInfo.html"**>结算</**a**> </**div**> |

pinyougou-cart-web/src/main/webapp/order/getOrderInfo.html引入js:

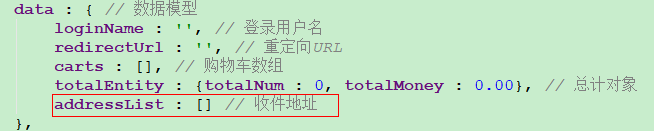
|  |
| --- |
| <**script src="/js/vue.min.js"**></**script**> <**script src="/js/axios.min.js"**></**script**> <**script src="/js/controller/cartController.js"**></**script**> |

getOrderInfo.html显示登录用户名(19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**> |

pinyougou-cart-web\src\main\webapp\js\controller\cartController.js文件：

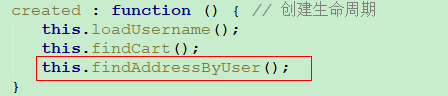
* 在data中定义变量



* 在methods中定义方法

|  |
| --- |
| *// 查询用户收件地址* findAddressByUser : **function** () {  **axios**.get(**"/order/findAddressByUser"**).then(**function**(response){  vue.**addressList** = response.**data**;  }); } |

* 初始化调用



getOrderInfo.html循环列表(71行)：

|  |
| --- |
| <**div v-for="item in addressList"**>  <**div class="con name selected"**>  <**a href="javascript:;"**>{{***item***.contact}}  <**span title="点击取消选择"**>**&nbsp;**</**span**></**a**>  </**div**>  <**div class="con address"**>{{***item***.address}}  <**span**>**&nbsp;&nbsp;**{{***item***.mobile}}</**span**>  <**span class="base" v-if="item.isDefault == 1"**>默认地址</**span**>  <**span**>  <**a data-toggle="modal" data-target=".edit"  data-keyboard="false"**>编辑</**a**>**&nbsp;&nbsp;** <**a href="javascript:;"**>删除</**a**>  </**span**>  </**div**>  <**div class="clearfix"**></**div**> </**div**> |

### 2.4.2后端代码

pinyougou-cart-web/src/main/java/com/pinyougou/cart/controller/包下，创建AddressController.java：

|  |
| --- |
| @RestController @RequestMapping(**"/order"**) **public class** AddressController {  @Reference(timeout = 10000)  **private** AddressService **addressService**;  */\*\* 获取登录用户的地址列表 \*/* @GetMapping(**"/findAddressByUser"**)  **public** List<Address> findAddressByUser(HttpServletRequest request){  *// 获取登录用户名* **String userId = request.getRemoteUser();**  **return addressService**.findAddressByUser(userId);  } } |

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

|  |
| --- |
| **public interface** AddressService {  */\*\*  \* 根据用户编号查询地址  \** ***@param userId*** *用户编号  \** ***@return 地址集合*** *\*/* List<Address> findAddressByUser(String userId); } |

pinyougou-user-service\src\main\java\com\pinyougou\user\service\impl包下，创建AddressServiceImpl.java:

|  |
| --- |
| @Service(interfaceName = **"com.pinyougou.service.AddressService"**) @Transactional **public class** AddressServiceImpl **implements** AddressService {  */\*\* 注入数据访问接口代理对象 \*/* @Autowired  **private** AddressMapper **addressMapper**;  */\*\*  \* 根据用户编号查询地址  \** ***@param userId*** *用户编号  \** ***@return*** *地址集合  \*/* **public** List<Address> findAddressByUser(String userId){  **try**{  *// 创建示范对象* Example example = **new** Example(Address.**class**);  *// 创建条件对象* Example.Criteria criteria = example.createCriteria();  *// userId = ?* criteria.andEqualTo(**"userId"**, userId);  *// 按默认地址排序* example.orderBy(**"isDefault"**).desc();  *// 条件查询* **return addressMapper**.selectByExample(example);  }**catch** (Exception ex){  **throw new** RuntimeException(ex);  }  }  @Override  **public void** save(Address address) {}  @Override  **public void** update(Address address) {}  @Override  **public void** delete(Serializable id) {}  @Override  **public void** deleteAll(Serializable[] ids) {}  @Override  **public** Address findOne(Serializable id) {  **return null**;  }  @Override  **public** List<Address> findAll() {  **return null**;  }  @Override  **public** List<Address> findByPage(Address address, **int** page, **int** rows) {  **return null**;  } } |

显示效果如下：



## 2.5地址选择&样式控制

修改getOrderInfo.html页面-点击选择(71行)：

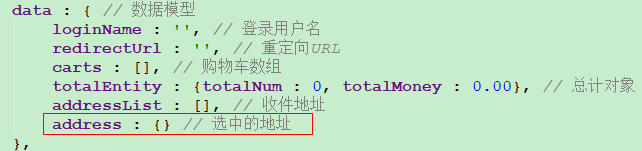
|  |
| --- |
| <**div v-for="item in addressList"**>  <**div :class="isSelectedAddress(item) ? 'con name selected' : 'con name'"**>  <**a href="javascript:;"  @click="selectAddress(item);"**>  {{***item***.contact}}  <**span title="点击取消选择"**>**&nbsp;**</**span**>  </**a**>  </**div**>  <**div class="con address"**>{{***item***.address}}  <**span**>**&nbsp;&nbsp;**{{***item***.mobile}}</**span**>  <**span class="base" v-if="item.isDefault == 1"**>默认地址</**span**>  <**span**>  <**a data-toggle="modal" data-target=".edit"  data-keyboard="false"**>编辑</**a**>**&nbsp;&nbsp;** <**a href="javascript:;"**>删除</**a**>  </**span**>  </**div**>  <**div class="clearfix"**></**div**> </**div**> |

修改getOrderInfo.html页面，显示选择的地址(243行)：

|  |
| --- |
| <**div class="fc-receiverInfo"**>  寄送至:{{address.address}} **&nbsp;** 收货人：{{address.contact}} **&nbsp;** 手机号码：{{address.mobile}} </**div**> |

pinyougou-cart-web\src\main\webapp\js\controller\cartController.js文件：

* 在data中定义变量



* 在methods中定义方法

|  |
| --- |
| *// 选择地址* selectAddress : **function** (item) {  **this**.**address** = item; }, *// 判断是否为选中的地址* isSelectedAddress : **function** (item) {  **return this**.**address** == item; } |

## 2.6默认地址显示

pinyougou-cart-web\src\main\webapp\js\controller\cartController.js，修改methods中的findAddressByUser方法：

|  |
| --- |
| *// 查询用户收件地址* findAddressByUser : **function** () {  **axios**.get(**"/order/findAddressByUser"**).then(**function**(response){  vue.**addressList** = response.**data**;  *// 取第一个作为默认收件地址* vue.**address** = response.**data**[0];  }); }, |

# 3.结算页【支付方式选择】

## 3.1需求分析

实现支付方式的选择，品优购支持两种支付方式：微信支付和货到付款

## 3.2支付方式选择

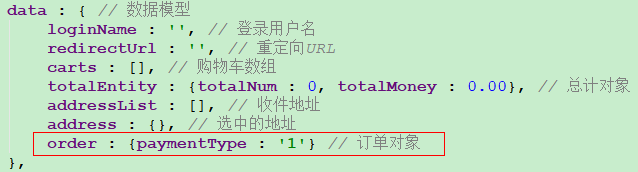
### 3.2.1页面

pinyougou-cart-web/src/main/webapp/order/getOrderInfo.html(166行)：

|  |
| --- |
| <**ul class="payType"**>  <**li :class="order.paymentType == 1 ? 'selected' : ''"  @click="order.paymentType = 1;"**>  微信付款<**span title="点击取消选择"**></**span**>  </**li**>  <**li :class="order.paymentType == 2 ? 'selected' : ''"  @click="order.paymentType = 2;"**>  货到付款<**span title="点击取消选择"**></**span**>  </**li**> </**ul**> |

### 3.2.2控制层

pinyougou-cart-web\src\main\webapp\js\controller\cartController.js，在data中定义变量：



# 4.结算页【商品清单】

## 4.1需求分析

显示**购物车**中的商品清单以及合计数量、合计金额

## 4.2购物车商品清单显示

getOrderInfo.html循环显示商品清单(184行)：

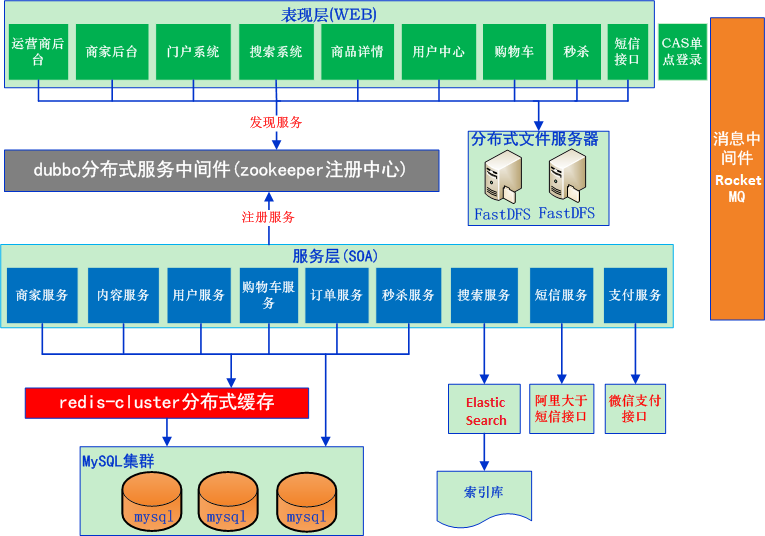
|  |
| --- |
| <**li v-for="cart in carts"**>  <**div v-for="item in cart.orderItems"**>  <**ul class="yui3-g"**>  <**li class="yui3-u-1-6"**>  <**span**><**img :src="item.picPath" width="100"**/></**span**>  </**li**>  <**li class="yui3-u-7-12"**>  <**div class="desc"**>{{***item***.**title**}}</**div**>  <**div class="seven"**>7天无理由退货</**div**>  </**li**>  <**li class="yui3-u-1-12"**>  <**div class="price"**>￥{{***item***.price}}</**div**>  </**li**>  <**li class="yui3-u-1-12"**>  <**div class="num"**>X{{***item***.num}}</**div**>  </**li**>  <**li class="yui3-u-1-12"**>  <**div class="exit"**>有货</**div**>  </**li**>  </**ul**>  </**div**> </**li**> |

## 4.3合计总件数&合计金额显示

修改 getOrderInfo.html(230行)

|  |
| --- |
| <**span**><**b class="number"**>{{**totalEntity**.**totalNum**}}</**b**>件商品，总商品金额</**span**> <**em class="allprice"**>¥{{**totalEntity**.**totalMoney**.toFixed(2)}}</**em**>  ...... 244行左右  应付金额:　<**span class="price"**>¥{{totalEntity.totalMoney.toFixed(2)}}</**span**> |

# 搭建订单服务工程



## 5.1 pinyougou-order-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-order-service/pom.xml：

|  |
| --- |
| <**properties**>  *<!-- 指定端口号 -->* <**port**>9007</**port**> </**properties**> |

pinyougou-order-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"**/>  *<!-- 导入数据访问配置文件 -->* <**import resource="classpath:applicationContext-mapper.xml"**/>  *<!-- 导入redis配置文件 -->* <**import resource="classpath:applicationContext-redis.xml"**/>  *<!-- ############## 配置dubbo服务提供者 ############## -->  <!-- 配置当前应用的名称 -->* <**dubbo:application name="pinyougou-order-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20886端口暴露服务 -->* <**dubbo:protocol name="dubbo" port="20886"**/>  *<!-- 配置采用包扫描来暴露服务 -->* <**dubbo:annotation package="com.pinyougou.order.service.impl"**/> </**beans**> |

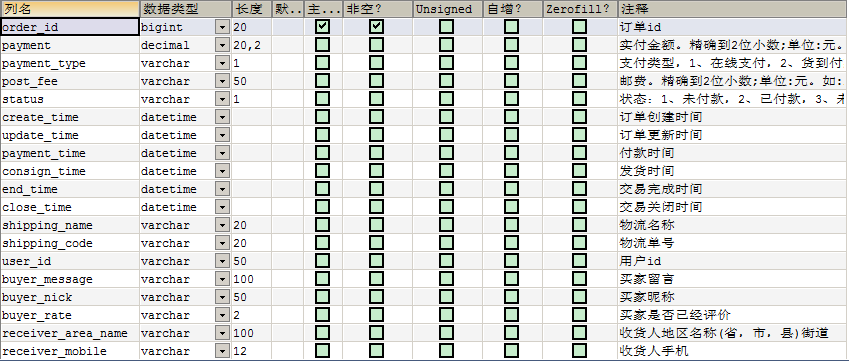
# 保存订单

## 6.1需求分析

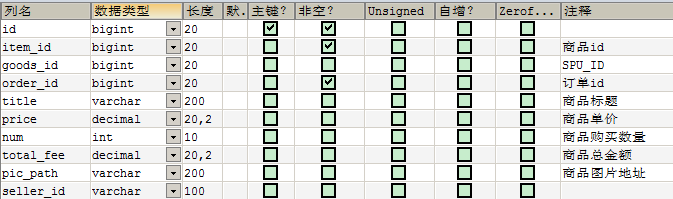
点击订单结算页的提交订单 ，将购物车保存到订单表和订单明细表中，并将购物车数据清除

## 6.2数据库结构分析

tb\_order订单主表

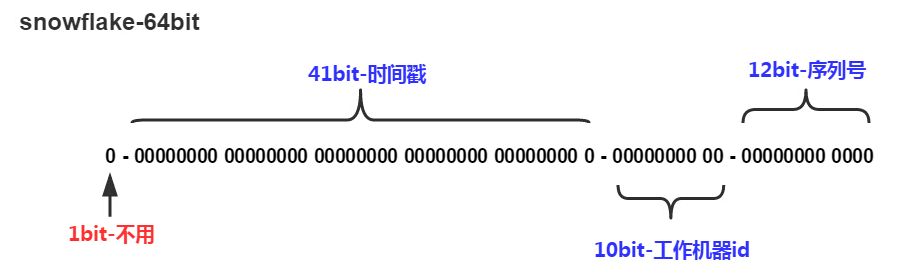


tb\_order\_item订单明细表



## 6.3分布式ID生成器

我们采用的是开源的twitter(非官方中文惯称：推特.是国外的一个网站，是一个社交网络及微博客服务)的snowflake算法。



1. 将“资料”下的IdWorker.java 拷贝到pinyougou-common/src/main/java/ com/pinyougou/common/util包下。
2. pinyougou-order-service/src/main/resources/applicationContext-service.xml文件中，添加配置

|  |
| --- |
| *<!-- 配置IdWorker分布式ID生成器 -->* <**bean id="idWorker" class="com.pinyougou.common.util.IdWorker"**>  *<!-- 工作id -->* <**constructor-arg name="workerId" value="0"**/>  *<!-- 数据中心id -->* <**constructor-arg name="datacenterId" value="0"**/> </**bean**> |

## 6.4前端代码

### 6.4.1页面

pinyougou-cart-web/src/main/webapp/order/getOrderInfo.html(250行)：

|  |
| --- |
| <**a class="sui-btn btn-danger btn-xlarge"  href="javascript:;" @click="saveOrder();"**>提交订单</**a**> |

### 6.4.2控制器层

pinyougou-cart-web\src\main\webapp\js\controller\cartController.js，在methods中定义方法：

|  |
| --- |
| *// 保存订单* saveOrder : **function** () {  *// 设置收件人地址* **this**.**order**.**receiverAreaName** = **this**.**address**.address;  *// 设置收件人手机号码* **this**.**order**.**receiverMobile** = **this**.**address**.mobile;  *// 设置收件人* **this**.**order**.**receiver** = **this**.**address**.contact;  *// 设置订单来源(pc端)* **this**.**order**.**sourceType** = 2;  *// 发送异步请求* **axios**.post(**"/order/save"**, **this**.**order**)  .then(**function** (response) {  **if** (response.**data**){  *// 如果是微信支付，跳转到扫码支付页面* **if** (vue.**order**.**paymentType** == 1){  ***location***.href = **"/order/pay.html"**;  }**else**{  *// 如果是货到付款，跳转到成功页面* ***location***.href = **"/order/paysuccess.html"**;  }  }**else**{  *alert*(**"订单提交失败！"**);  }  }); } |

## 6.5后端代码

### 6.5.1控制器层

pinyougou-cart-web/src/main/java/com/pinyougou/cart/controller/包下，创建OrderController.java：

|  |
| --- |
| @RestController @RequestMapping(**"/order"**) **public class** OrderController {  @Reference(timeout = 10000)  **private** OrderService **orderService**;  */\*\* 保存订单 \*/* @PostMapping(**"/save"**)  **public boolean** save(@RequestBody Order order,  HttpServletRequest request){  **try**{   ***// 获取登录用户名* String userId = request.getRemoteUser();  order.setUserId(userId);  orderService.save(order);**  **return true**;  }**catch**(Exception ex){  ex.printStackTrace();  }  **return false**;  } } |

### 6.5.2服务层

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

|  |
| --- |
| @Service @Transactional **public class** OrderServiceImpl **implements** OrderService {  @Autowired  **private** OrderMapper **orderMapper**;  @Autowired  **private** OrderItemMapper **orderItemMapper**;  @Autowired  **private** IdWorker **idWorker**;  @Autowired  **private** RedisTemplate<String, Object> **redisTemplate**;  */\*\* 保存订单 \*/*  @Override**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(**"cart\_"** + order.getUserId()).get();  *// 迭代购物车数据* **for** (Cart cart : carts){  */\*\* #######* ***往订单表插入数据*** *######### \*/  // 生成订单主键id* **long** orderId = **idWorker**.nextId();  *// 创建新的订单* Order order1 = **new** Order();  *// 设置订单id* order1.setOrderId(orderId);  *// 设置支付类型* order1.setPaymentType(order.getPaymentType());  *// 设置支付状态码为“未支付”* order1.setStatus(**"1"**);  *// 设置订单创建时间* order1.setCreateTime(**new** Date());  *// 设置订单修改时间* order1.setUpdateTime(order1.getCreateTime());  *// 设置用户名* order1.setUserId(order.getUserId());  *// 设置收件人地址* order1.setReceiverAreaName(order.getReceiverAreaName());  *// 设置收件人手机号码* order1.setReceiverMobile(order.getReceiverMobile());  *// 设置收件人* order1.setReceiver(order.getReceiver());  *// 设置订单来源* order1.setSourceType(order.getSourceType());  *// 设置商家id* order1.setSellerId(cart.getSellerId());   *// 定义该订单总金额* **double** money = 0;  */\*\* #######* ***往订单明细表插入数据*** *######### \*/* **for** (OrderItem orderItem : cart.getOrderItems()){  *// 设置主键id* orderItem.setId(**idWorker**.nextId());  *// 设置关联的订单id* orderItem.setOrderId(orderId);  *// 累计总金额* money += orderItem.getTotalFee().doubleValue();  *// 保存数据到订单明细表* **orderItemMapper**.insertSelective(orderItem);  }  *// 设置支付总金额* order1.setPayment(**new** BigDecimal(money));  *// 保存数据到订单表* **orderMapper**.insertSelective(order1);  }  *// 删除该用户购物车数据* **redisTemplate**.delete(**"cart\_"** + order.getUserId());  }**catch** (Exception ex){  **throw new** RuntimeException(ex);  }  }  @Override  **public void** update(Order order) {  }  @Override  **public void** delete(Serializable id) {  }  @Override  **public void** deleteAll(Serializable[] ids) {  }  @Override  **public** Order findOne(Serializable id) {  **return null**;  }  @Override  **public** List<Order> findAll() {  **return null**;  }  @Override  **public** List<Order> findByPage(Order order, **int** page, **int** rows) {  **return null**;  } } |