**新巴巴运动网 项目第十一天**

**传智.上官云**

# 课前回顾

1. 购物车

加入购物车 ---

1）从Cookie取出购物车

2）取出的购物车是空或不是空

3）如果是空创建一个新购物车

4）将当前的款商品加入购物车

非登陆

5）保存购物车到Cookie、并写回浏览器

登陆

5）保存购物车到Redis中 、清空Cookie

去购物车页面

1. 从Cookie取出购物车（共有的）

非登陆

不用写

登陆

1. 把Cookie合并到Redis中 、清空Cookie
2. 取出Redis中的购物车

取出的购物车是空或不是空（共有的）

如果不是空把购物车装满(共有的)

1. 不登陆 关闭浏览器要存在
2. 跨区 必须登陆

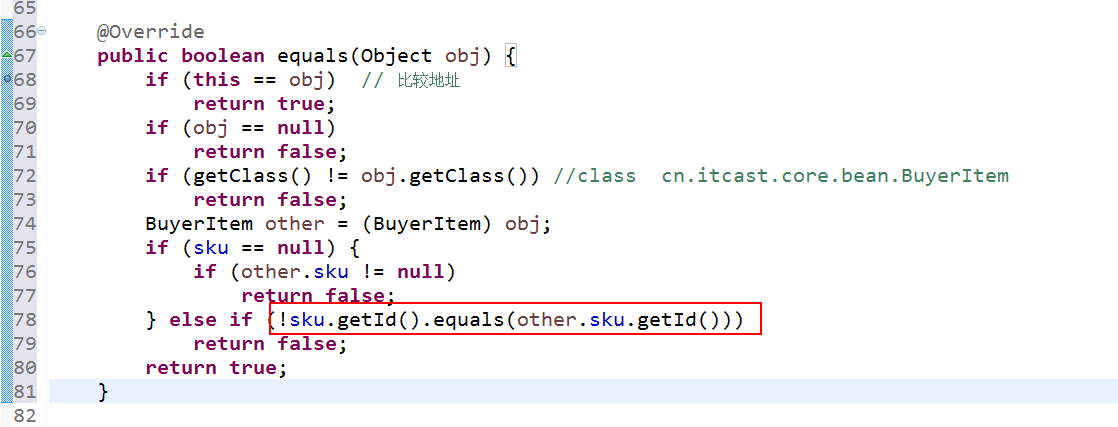
# 今天内容

1. 购物车 问题 同款商品合并（非登陆） 登陆
2. 小计
3. 排序（同学完成）
4. 结算（1）判断用户必须登陆、2）购物车必须有商品3）商品必须有库存
5. 订单表（2张） 面试问题
6. 提交订单（保存订单）
7. 企业整体架构分析（40分钟）
8. 高并发 、高可用 （企业面临的二大问题）
9. Mysql数据库集群（大企业解决方法） 成熟的解决的方案 （出身阿里巴巴研究院Cobar）
10. Mycat

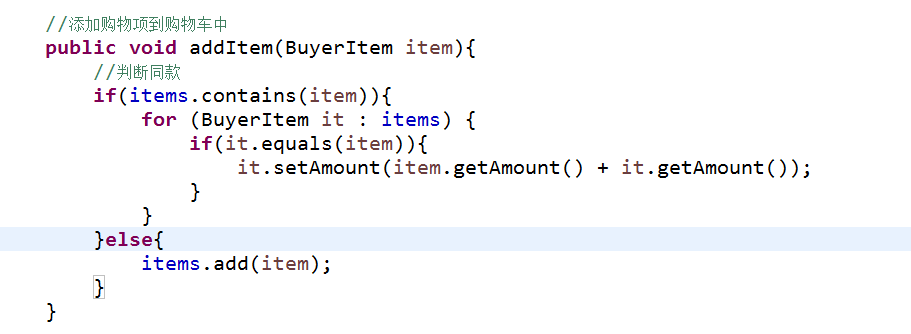
# 购物车 问题 同款商品合并（非登陆） 登陆

## 同款商品合并（非登陆）

BuyerItem

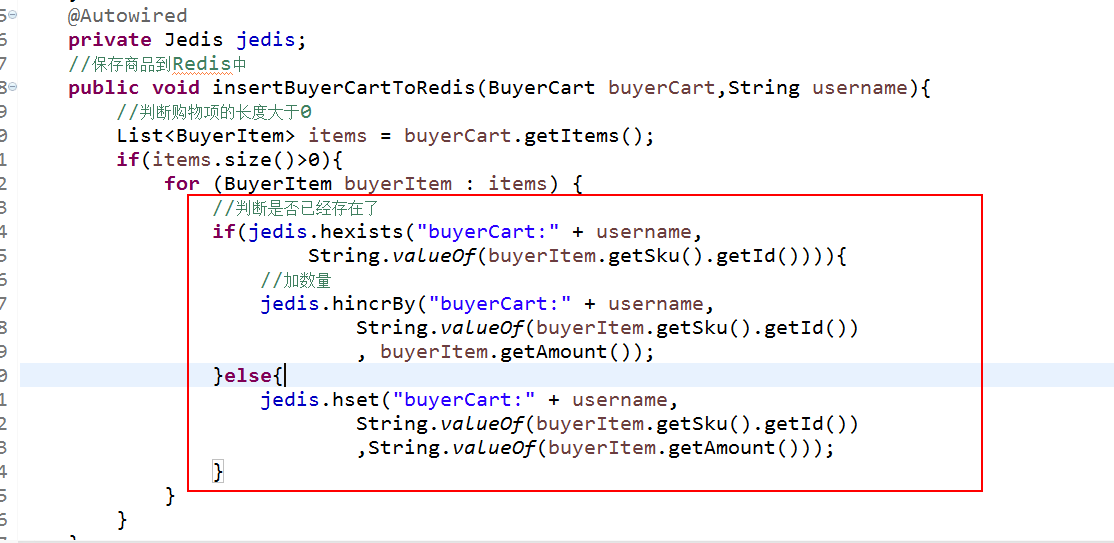


BuyerCart



## 登陆时同款商品合并

SkuService中



## 小计

PO类里算 BuyerCart中计算 （面向对象）

注意事项：

购物车不是标准JavaBean

在JSON字符串转换时 报错

|  |
| --- |
| //2:小计 （商品数据 、商品金额 、运费 、 总计）  //商品数量  @JsonIgnore  public Integer getProductAmount(){  Integer result = 0;  //计算过程  for (BuyerItem buyerItem : items) {  result += buyerItem.getAmount();  }  return result;  }  //商品金额  @JsonIgnore  public Float getProductPrice(){  Float result = 0f;  //计算过程  for (BuyerItem buyerItem : items) {  result += buyerItem.getAmount()\*buyerItem.getSku().getPrice();  }  return result;  }  // 运费  @JsonIgnore  public Float getFee(){  Float result = 0f;  if(getProductPrice() < 79){  result = 5f;  }  return result;  }  //总金额  @JsonIgnore  public Float getTotalPrice(){  return getProductPrice() + getFee();  } |

# 结算

## 区别

路径 /login.aspx

/search

/toCart

/addCart

必须登陆的请求路径

/buyer/trueBuy

## 设置结算的请求路径

## CartController

方法

结算

1. 购物车中必须有商品
2. 购物车中必须有库存

## Springmvc拦截器

1. 登陆成功后 回跳回首页 （同学 回跳回购物车页面）

Springmvc拦截器 进入Controller之前 拦截下来(处理) 根据判断的情况决定让不让进入Controller

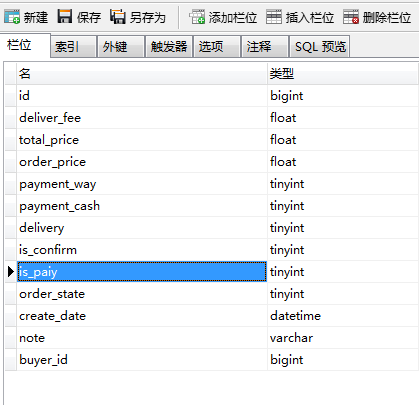


结算

|  |
| --- |
| //结算  @RequestMapping(value = "/buyer/trueBuy")  public String trueBuy(Long[] skuIds,Model model,HttpServletRequest request,HttpServletResponse response){  // 1） 购物车中必须有商品  String username = sessionProvider.getAttributeForUsername(RequestUtils.getCSESSIONID(request, response));  BuyerCart buyerCart = skuService.selectBuyerCartFromRedis(username);  List<BuyerItem> items = buyerCart.getItems();  //标记  Boolean flag = false;  if(items.size()>0){  // 把购物车装满  for (BuyerItem buyerItem : items) {  buyerItem.setSku(skuService.selectSkuById(buyerItem.getSku().getId()));  // 2） 购物车中必须有库存  if(buyerItem.getAmount() > buyerItem.getSku().getStock()){  //无货  buyerItem.setIsHave(false);  flag = true;  }  }  //至少一款无货  if(flag){  //视图 有一个无货不能进入下个订单页面  model.addAttribute("buyerCart", buyerCart);  return "cart";  }  }else{  return "redirect:/toCart";  }  //视图 如果都有货进入下个订单页面  return "order";    } |

# 提交订单

## 订单表 2张



订单表

ID（订单编号）全国唯一Redis

运费　　　　　由购物车提供

总价

订单金额

支付方式： 由页面传递

支付要求： 由页面传递

运输方式：

是否电话确认 空

支付状态：0到付1待付款,2已付款,3待退款,4退款成功,5退款失败

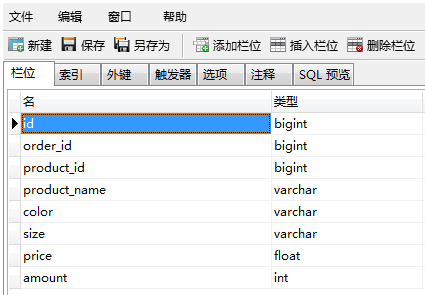
订单状态：0:提交订单 1:仓库配货 2:商品出库 3:等待收货 4:完成 5待退货 6已退货

时间 ： 后台程序自己写的

留言：页面传递

用户ID ：

## 订单详情表



订单详情表

ID

订单ID

商品编号

商品名称

颜色

尺码

价格

数量

购物车提供

# 提交订单

## 提交

Form

## OrderService

入参: 订单对象 用户名

用户名去取

返回值：void

保存订单 用户ID

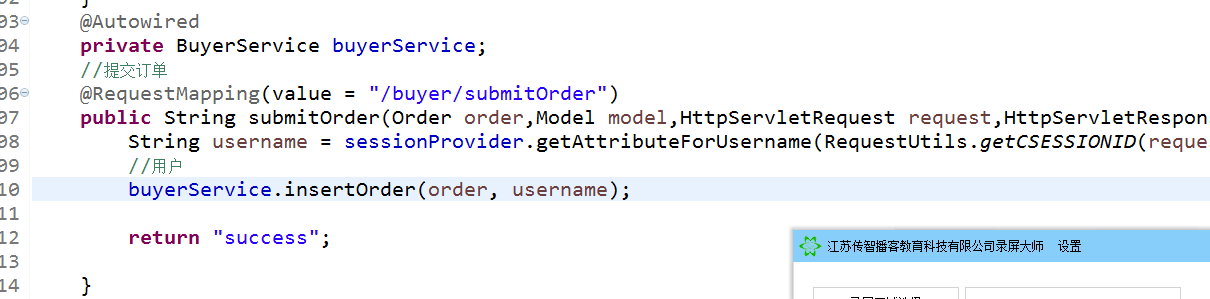
保存订单详情

|  |
| --- |
| //保存订单  public void insertOrder(Order order,String username){  // ID（订单编号）全国唯一Redis  Long id = jedis.incr("oid");  order.setId(id);  //加载购物车  BuyerCart buyerCart = selectBuyerCartFromRedis(username);  List<BuyerItem> items = buyerCart.getItems();  for (BuyerItem buyerItem : items) {  buyerItem.setSku(selectSkuById(buyerItem.getSku().getId()));  }  // 运费　　　　　由购物车提供  order.setDeliverFee(buyerCart.getFee());  // 总价  order.setTotalPrice(buyerCart.getTotalPrice());  // 订单金额  order.setOrderPrice(buyerCart.getProductPrice());  // 支付状态：0到付1待付款,2已付款,3待退款,4退款成功,5退款失败  if(order.getPaymentWay() == 1){  order.setIsPaiy(0);  }else{  order.setIsPaiy(1);  }  // 订单状态：0:提交订单 1:仓库配货 2:商品出库 3:等待收货 4:完成 5待退货 6已退货  order.setOrderState(0);  // 时间 ： 后台程序自己写的  order.setCreateDate(new Date());  // 用户ID ： 前台用户做注册 （ 用户名、密码）用户ID 由Redis生成 用户名： 用户ID 保存到Redis中 5个  String uid = jedis.get(username);  order.setBuyerId(Long.parseLong(uid));  //保存订单  orderDao.insertSelective(order);  //保存订单详情  for (BuyerItem buyerItem : items) {  Detail detail = new Detail();  // ID  // 订单ID  detail.setOrderId(id);  // 商品编号  detail.setProductId(buyerItem.getSku().getProductId());  // 商品名称  detail.setProductName(buyerItem.getSku().getProduct().getName());  // 颜色  detail.setColor(buyerItem.getSku().getColor().getName());  // 尺码  detail.setSize(buyerItem.getSku().getSize());  // 价格  detail.setPrice(buyerItem.getSku().getPrice());  // 数量  detail.setAmount(buyerItem.getAmount());  // 购物车提供  detailDao.insertSelective(detail);  }    //清空购物车  jedis.del("buyerCart:fbb2016");  //练习 hash 指定 K        } |

## Controller

入参： 订单对象

返回值：有 订单确认：下一个 订单号



# 企业整体架构分析

略

# 负载均衡

由于目前现有网络的各个核心部分随着业务量的提高，访问量和数据流量的快速增长，其处理能力和计算强度也相应地增大，使得单一的服务器设备根本无法承担。在此情况下，如果扔掉现有设备去做大量的硬件升级，这样将造成现有资源的浪费，而且如果再面临下一次业务量的提升时，这又将导致再一次硬件升级的高额成本投入，甚至性能再卓越的设备也不能满足当前业务量增长的需求。   
　　针对此情况而衍生出来的一种廉价有效透明的方法以扩展现有网络设备和服务器的带宽、增加吞吐量、加强网络数据处理能力、提高网络的灵活性和可用性的技术就是负载均衡（Load Balance）。

## 负载均衡的种类

1. 一种是通过硬件来进行解决，常见的硬件有NetScaler、F5、Radware和Array等商用的负载均衡器，但是它们是比较昂贵的

2）一种是通过软件来进行解决的，常见的软件有LVS、Nginx、apache等,它们是基于Linux系统并且开源的负载均衡策略.

## 什么是Apache + JK?

Apache是世界使用排名第一的Web[服务器](http://baike.baidu.com/view/899.htm)软件。它可以运行在几乎所有广泛使用的[计算机平台](http://baike.baidu.com/view/2269685.htm)上，由于其[跨平台](http://baike.baidu.com/view/469855.htm)和安全性被广泛使用，是最流行的Web服务器端软件

JK是apache提供的一款为解决大量请求而分流处理的开源插件

## 什么是nginx?

**Nginx**（发音同 engine x）是一款轻量级的[Web](http://baike.baidu.com/view/3912.htm) 服务器/反向代理服务器及电子邮件（IMAP/POP3）代理服务器，并在一个BSD-like 协议下发行。由俄罗斯的程序设计师Igor Sysoev(伊戈尔·西索夫)所开发，供俄国大型的入口网站及搜索引擎Rambler(漫步者)（俄文：Рамблер）使用。其特点是占有[内存](http://baike.baidu.com/view/1082.htm)少，[并发](http://baike.baidu.com/view/684757.htm)能力强，事实上nginx的并发能力确实在同类型的网页服务器中表现较好，中国大陆使用nginx网站用户有：[新浪](http://baike.baidu.com/view/2410.htm)、[网易](http://baike.baidu.com/view/6043.htm)、 [腾讯](http://baike.baidu.com/view/1591.htm)等。

优点：

1:可运行linux,并有 [Windows](http://baike.baidu.com/view/4821.htm) 移植版。

2:在高连接并发的情况下，Nginx是Apache服务器不错的替代品Nginx在[美国](http://baike.baidu.com/view/2398.htm)是做虚拟主机生意的老板们经常选择的软件平台之一。能够支持高达 50,000 个并发连接数的响应

Tomcat 300-500-800

## 什么是LVS?

LVS的英文全称是Linux Virtual Server，即Linux虚拟服务器。它是我们国家的章文嵩博士的一个开源项目。在linux内核2.6中，它已经成为内核的一部分，在此之前的内核版本则需要重新编译内核。

# Mysql集群原理