**新巴巴运动网 项目第一天**

**传智.上官云**

# 课前回顾

1. 单点登陆 （搭建环境）
2. 去登陆页面（<http://localhost:8082/login.aspx?return> Url = EncodeURLComment(window.location.href)
3. 提交登陆（1）用户名、密码、ReturnUrl）
4. Value=login.aspx ,method=POST GET
5. 加密
6. 把用户名保存Redis（Session共享） 1）分析K：V CSESSIONID：USER-Name V：fbb2016

2)SessionProvider ( SessionProviderImpl) 手动实例化 （xml配置时间 60分钟）

3）保存用户名到Redis中

7、 重定向到之前访问页面

8、 页面 （登陆、注册、退出，我的订单） 分析：页面（动） 静 ajax判断用户是否登陆

页面（前台系统判断用户是否登陆） ajax 去另外一个系统 返回值被拦截（浏览器）

判断程序（单点登陆系统）

Ajax 返回值 ：json🡪jsonp

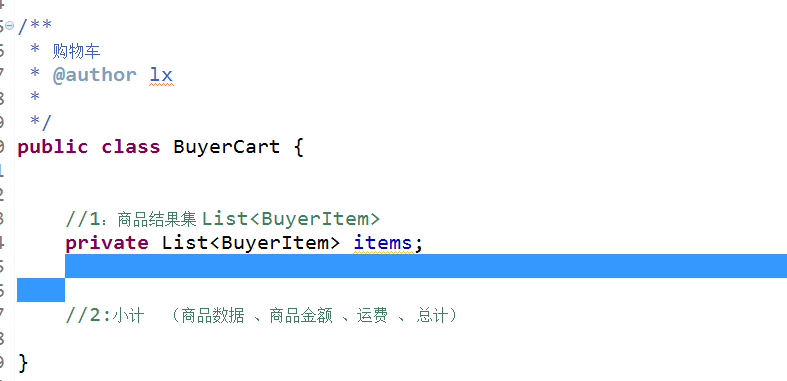
登陆系统（接收String callback） Spring MappingJacksonValue (保存数据 返回JSONP的数据) mjv.setJSONFunction(callback); return mjv;(方法返回值处@ResponseBody MappingJacksonValue

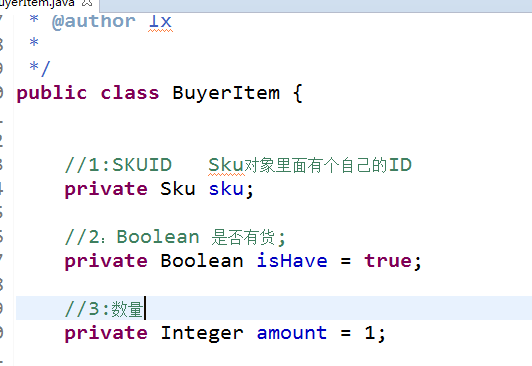
# 今天内容

主打购物车

1. 购物车
2. 面向对象设计购物车
3. 购物车 非登陆 登陆
4. 关闭浏览器还必须在
5. 跨电脑或区域购物车还在
6. 同款数量追加 同款商品合并
7. 排序同学完成

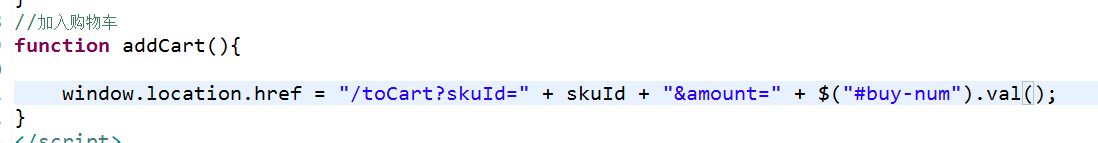
# 面向对象设计购物车





# 加入购物车

## 设置路径



## Controller

# 分析购物车

Cookie中

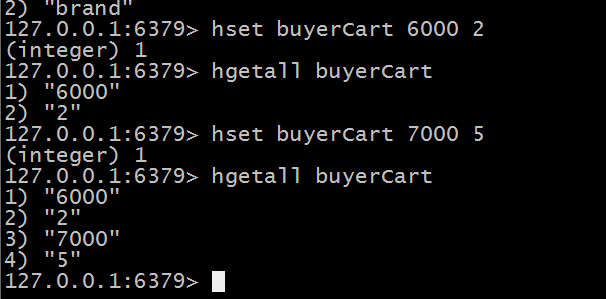
Session中(Redis) 用户名（用户ID）

数据库

非登陆：商品保存Cookie里 关闭浏览器还在 （七天、一个月）用户清理Cookie就没了

登陆 ： Redis 性能比较好 关闭浏览器（不登陆不在的） 登陆就在了 （跨区域） 永久保存

## Redis保存购物车



加入购物车

登陆

非登陆

1：从Request中取Cookies、遍历Cookie 取出之前的购物车

2：判断Cookie中没有购物车

3：有 把购物车中商品添加到Redis的购物车中，清理之前Cookie

4：没有

5：直接添加当前商品到Redis中的购物车里

1：从Request中取Cookies、遍历Cookie 取出之前的购物车

2：判断Cookie中没有购物车

3：有

4：没有 创建购物车

5：追加当前商品到购物车

6：创建Cookie 把新购物车放进去

7：保存Cookie写回浏览器

去购物车页面

判断是否登陆

登陆

1：从Request中取Cookies、遍历Cookie 取出之前的购物车

2：再把购物车保存到Redis中，清理Cookie

3：从Redis中取出所有的购物车

把购物车装满

跳转到购物车页面

回显购物车内容

1：从Request中取Cookies、遍历Cookie 取出之前的购物车

2:有 购物车数量及SKUID

3：没有

把购物车装满

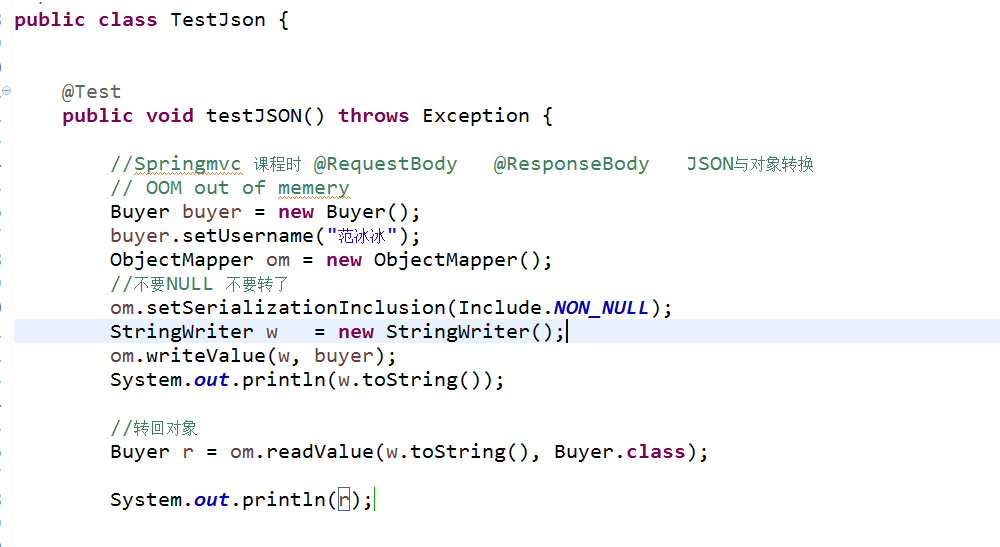
跳转到购物车页面

回显购物车内容

购物车（回显数据到购物车页面

# 购物车

## 对象与JSON字符串互转



## SkuService

入参： SKUID

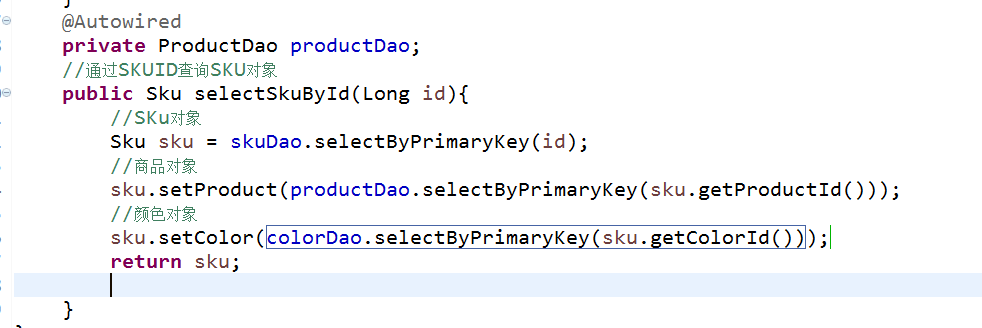
返回值：Sku对象

SKu对象（包括颜色对象、商品对象）

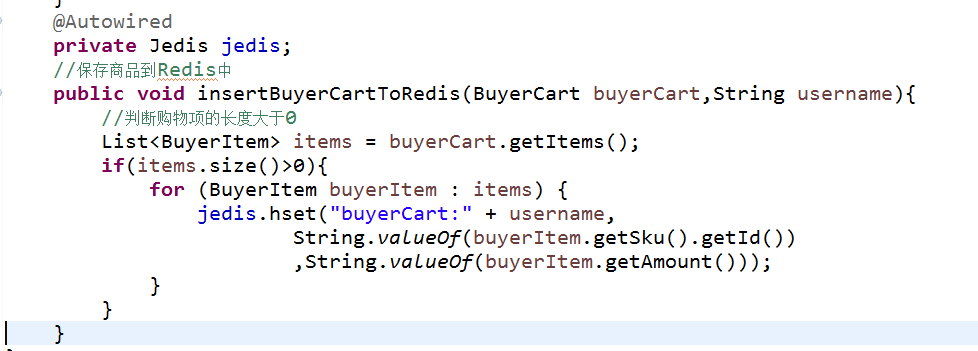
商品表

SKU表

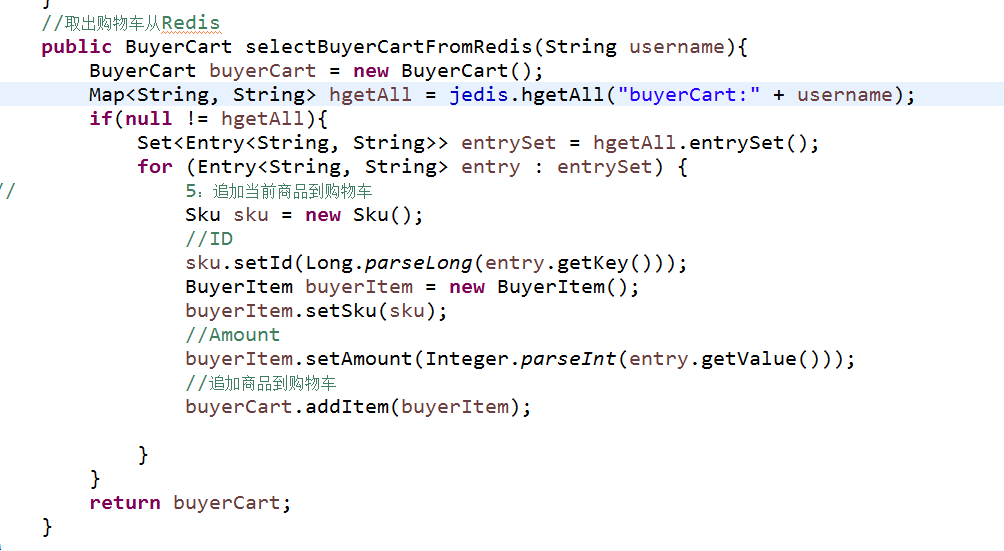
颜色表



保存商品到REdis中



从Redis中取出购物车



## Controller

### 未 优化之前的代码

|  |
| --- |
| package cn.itcast.core.controller;  import java.io.StringWriter;  import java.util.List;  import javax.servlet.http.Cookie;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Controller;  import org.springframework.ui.Model;  import org.springframework.web.bind.annotation.RequestMapping;  import com.fasterxml.jackson.annotation.JsonInclude.Include;  import com.fasterxml.jackson.databind.ObjectMapper;  import cn.itcast.common.utils.RequestUtils;  import cn.itcast.common.web.Constants;  import cn.itcast.core.bean.BuyerCart;  import cn.itcast.core.bean.BuyerItem;  import cn.itcast.core.bean.product.Sku;  import cn.itcast.core.service.product.SkuService;  import cn.itcast.core.service.user.SessionProvider;  /\*\*  \* 购物车  \* 去购物车页面  \* 添加商品到购物车  \* 删除  \* —+  \* @author lx  \*  \*/  @Controller  public class CartController {  @Autowired  private SessionProvider sessionProvider;    //加入购物车  @RequestMapping(value = "/addCart")  public String addCart(Long skuId,Integer amount,Model model  ,HttpServletRequest request,HttpServletResponse response) throws Exception{    ObjectMapper om = new ObjectMapper();  //不要NULL 不要转了  om.setSerializationInclusion(Include.NON\_NULL);  //声明  BuyerCart buyerCart = null;    // 1：从Request中取Cookies、  Cookie[] cookies = request.getCookies();  if(null != cookies&& cookies.length >0){  //遍历Cookie 取出之前的购物车  for (Cookie cookie : cookies) {  // 2：判断Cookie中没有购物车  if(Constants.BUYER\_CART.equals(cookie.getName())){  //转回对象  buyerCart = om.readValue(cookie.getValue(), BuyerCart.class);  break;  }  }  }    //用户是否登陆  String username = sessionProvider.getAttributeForUsername(RequestUtils.getCSESSIONID(request, response));  if(null != username){  if(null != buyerCart){    // 3：有 把购物车中商品添加到Redis的购物车中，  skuService.insertBuyerCartToRedis(buyerCart, username);  //清理之前Cookie4  Cookie cookie = new Cookie(Constants.BUYER\_CART,null);  cookie.setMaxAge(0);  cookie.setPath("/");  response.addCookie(cookie);  }  // 4：没有  // 5：直接添加当前商品到Redis中的购物车里  //程序未写        }else{  // 3：有  // 4：没有 创建购物车  //判断购物车是否为null  if(null == buyerCart){  buyerCart = new BuyerCart();  }  //  // 5：追加当前商品到购物车  Sku sku = new Sku();  //ID  sku.setId(skuId);  BuyerItem buyerItem = new BuyerItem();  buyerItem.setSku(sku);  //Amount  buyerItem.setAmount(amount);  //追加商品到购物车  buyerCart.addItem(buyerItem);  // 6：创建Cookie 把新购物车放进去  StringWriter w = new StringWriter();  om.writeValue(w, buyerCart);  Cookie cookie = new Cookie(Constants.BUYER\_CART,w.toString());  //设置时间 写程序1天  cookie.setMaxAge(60\*60\*24);  //设置路径  cookie.setPath("/");  //上线后 申请域名  // 7：保存写回浏览器  response.addCookie(cookie);  }    return "redirect:/toCart";  }  @Autowired  private SkuService skuService;  //去购物车页面  @RequestMapping(value = "/toCart")  public String toCart(Model model  ,HttpServletRequest request,HttpServletResponse response) throws Exception{  // 1：从Request中取Cookies、遍历Cookie 取出之前的购物车  ObjectMapper om = new ObjectMapper();  //不要NULL 不要转了  om.setSerializationInclusion(Include.NON\_NULL);  //声明  BuyerCart buyerCart = null;    // 1：从Request中取Cookies、  Cookie[] cookies = request.getCookies();  if(null != cookies&& cookies.length >0){  //遍历Cookie 取出之前的购物车  for (Cookie cookie : cookies) {  // 2：判断Cookie中没有购物车  if(Constants.BUYER\_CART.equals(cookie.getName())){  //转回对象  buyerCart = om.readValue(cookie.getValue(), BuyerCart.class);  break;  }  }  }  if(null != buyerCart){  // 2:有 购物车数量及SKUID  // 把购物车装满  List<BuyerItem> items = buyerCart.getItems();  for (BuyerItem buyerItem : items) {  buyerItem.setSku(skuService.selectSkuById(buyerItem.getSku().getId()));  }    }  // 3：没有  // 回显购物车内容  model.addAttribute("buyerCart", buyerCart);  // 跳转到购物车页面  return "cart";    }  } |

### 优化之后的代码

|  |
| --- |
| package cn.itcast.core.controller;  import java.io.StringWriter;  import java.util.List;  import javax.servlet.http.Cookie;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Controller;  import org.springframework.ui.Model;  import org.springframework.web.bind.annotation.RequestMapping;  import com.fasterxml.jackson.annotation.JsonInclude.Include;  import com.fasterxml.jackson.databind.ObjectMapper;  import cn.itcast.common.utils.RequestUtils;  import cn.itcast.common.web.Constants;  import cn.itcast.core.bean.BuyerCart;  import cn.itcast.core.bean.BuyerItem;  import cn.itcast.core.bean.product.Sku;  import cn.itcast.core.service.product.SkuService;  import cn.itcast.core.service.user.SessionProvider;  /\*\*  \* 购物车  \* 去购物车页面  \* 添加商品到购物车  \* 删除  \* —+  \* @author lx  \*  \*/  @Controller  public class CartController {  @Autowired  private SessionProvider sessionProvider;    //加入购物车  @RequestMapping(value = "/addCart")  public String addCart(Long skuId,Integer amount,Model model  ,HttpServletRequest request,HttpServletResponse response) throws Exception{    ObjectMapper om = new ObjectMapper();  //不要NULL 不要转了  om.setSerializationInclusion(Include.NON\_NULL);  //声明  BuyerCart buyerCart = null;    // 1：从Request中取Cookies、  Cookie[] cookies = request.getCookies();  if(null != cookies&& cookies.length >0){  //遍历Cookie 取出之前的购物车  for (Cookie cookie : cookies) {  // 2：判断Cookie中没有购物车  if(Constants.BUYER\_CART.equals(cookie.getName())){  //转回对象  buyerCart = om.readValue(cookie.getValue(), BuyerCart.class);  break;  }  }  }  // 3：有  // 4：没有 创建购物车  //判断购物车是否为null  if(null == buyerCart){  buyerCart = new BuyerCart();  }  // 5：追加当前商品到购物车  Sku sku = new Sku();  //ID  sku.setId(skuId);  BuyerItem buyerItem = new BuyerItem();  buyerItem.setSku(sku);  //Amount  buyerItem.setAmount(amount);  //追加商品到购物车  buyerCart.addItem(buyerItem);    //用户是否登陆  String username = sessionProvider.getAttributeForUsername(RequestUtils.getCSESSIONID(request, response));  if(null != username){  // 3：有 把购物车中商品添加到Redis的购物车中，  skuService.insertBuyerCartToRedis(buyerCart, username);  //清理之前Cookie4  Cookie cookie = new Cookie(Constants.BUYER\_CART,null);  cookie.setMaxAge(0);  cookie.setPath("/");  response.addCookie(cookie);  }else{  // 6：创建Cookie 把新购物车放进去  StringWriter w = new StringWriter();  om.writeValue(w, buyerCart);  Cookie cookie = new Cookie(Constants.BUYER\_CART,w.toString());  //设置时间 写程序1天  cookie.setMaxAge(60\*60\*24);  //设置路径  cookie.setPath("/");  //上线后 申请域名  // 7：保存写回浏览器  response.addCookie(cookie);  }    return "redirect:/toCart";  }  @Autowired  private SkuService skuService;  //去购物车页面  @RequestMapping(value = "/toCart")  public String toCart(Model model  ,HttpServletRequest request,HttpServletResponse response) throws Exception{  // 1：从Request中取Cookies、遍历Cookie 取出之前的购物车  ObjectMapper om = new ObjectMapper();  //不要NULL 不要转了  om.setSerializationInclusion(Include.NON\_NULL);  //声明  BuyerCart buyerCart = null;    // 1：从Request中取Cookies、  Cookie[] cookies = request.getCookies();  if(null != cookies&& cookies.length >0){  //遍历Cookie 取出之前的购物车  for (Cookie cookie : cookies) {  // 2：判断Cookie中没有购物车  if(Constants.BUYER\_CART.equals(cookie.getName())){  //转回对象  buyerCart = om.readValue(cookie.getValue(), BuyerCart.class);  break;  }  }  }  if(null != buyerCart){  // 2:有 购物车数量及SKUID  // 把购物车装满  List<BuyerItem> items = buyerCart.getItems();  for (BuyerItem buyerItem : items) {  buyerItem.setSku(skuService.selectSkuById(buyerItem.getSku().getId()));  }    }  // 3：没有  // 回显购物车内容  model.addAttribute("buyerCart", buyerCart);  // 跳转到购物车页面  return "cart";    }  } |

# 问题

## 同款商品合并

非登陆

登陆

## 小计

## 排序

非登陆

登陆

## 去结算

## 提交订单

## 付款（不做了）