之前没有接触过购物车的东东，也不知道购物车应该怎么做，所以在查询了很多资料，总结一下购物车的功能实现。

查询的资料，找到三种方法：

1.用cookie实现购物车；

2.用session实现购物车；

3.用cookie和数据库(购物车信息持久化)实现购物车；

=========================================================================

分析一下这三种方法的优缺点：

1.单纯有cookie实现购物车，这样的购物车不是很理想，设想一下，如果客户端的浏览器把cookie给禁用了，

这种方法就会在这里流产...

2.session中保存购物车的信息，这个只是在一个会话中可用，如果用户没有登录，或者说登录了以后，添加购物车，在关闭浏览器

或者登出后，之前所添加的购物车通通都流产啦...

3.我这里要说就是这种方法啦.....

主要的流程：

A.用户登录前的数据流：用户在没有登录系统的时候，对喜欢的商品进行添加购物车，那么这个时候，我们可以把购物车信息保存

到cookie中，这里会涉及到cookie的添加，修改操作；也即如果之前在cookie中不存对应的cookie，则就对cookie进行添加操作。

如果在cookie中存在对应的cookie，那么，这时候，就要对cookie进行修改操作了(这里涉及到用户对同一个商品进行多次添加购物车的情况)。

B.用户登录后的数据流：用户在登录后，系统首先做的第一件事就是去获取对应的cookies，如果存在相关的购物车cookies，那么就对该购物车

信息进行相应用户User的持久化操作，要么添加，要么修改。（添加操作:该用户所对应的购物车如果没有相应的信息进行添加操作；修改操作：类似的，

如果存在对应用户的购物车信息，就进行修改操作）。用户登录后，也可以进行购物车的添加操作，不过，这里不是添加到cookie中，而是直接持久化到

数据库中。注：用户登录后的数据都是和数据库打交道。

=========================================================================

代码部分：

=========================================================================

注：

1 Conf.IDUONA\_CASHTICKET\_COOKIE\_STARTNAME = "iduona\_cashTicket\_";

[复制代码](javascript:void(0);)

1 /\*\*

2 \* 用户登录

3 \*

4 \* @author hongten

5 \*/

6 public void login() {

7 //用户登录的时候，去读取cookies，并且进行持久话操作，更多的登录操作这里省略啦....

8 peristShoppingCartWhenUserLogin(newUser);

9 }

10

11 /\*\*

12 \* 加入购物车<br>

13 \* ============================================<br>

14 \* 用户登录前：<br>

15 \* 用户在选择现金券的时候，点击现金券的加入购物车的时候，会把该现金券的信息(现金券的id，购买数量)<br>

16 \* 传递到这里，这时候，后台要做的就是从cookie中查询出是否有相同的记录，如果有相同的记录<br>

17 \* 则把相应的记录更新；否则，就添加新的记录<br>

18 \* 用户登录后：<br>

19 \* 用户在登录后，如果有添加购物车操作，则不用保存到cookie中，而是直接持久化购物车信息<br>

20 \*

21 \* @throws Exception

22 \*/

23 public void addToShoppingCart() throws Exception {

24 if (cashTicket == null || cashTicket.getId() == null || cashTicket.getId() < 1) {

25 write("nullId");

26 } else if (q == null || q == "") {

27 // 购买数量，默认情况下面为1

28 q = String.valueOf(1);

29 } else {

30 // 读取所有的cookie

31 Cookie cookies[] = ServletActionContext.getRequest().getCookies();

32 if (cookies == null || cookies.length < 0) {

33 // 没有cookie

34 System.out.println("there is no any cookie ..");

35 } else {

36 // 判断用户是否登录

37 if (getUserInSession() == null) {

38 boolean flag = true;

39 for (Cookie c : cookies) {

40 if (c.getName().equals(Conf.IDUONA\_CASHTICKET\_COOKIE\_STARTNAME + cashTicket.getId())) {

41 // 说明已有的cookies中有相应的cookie，就进行更新操作

42 Integer oldValue = Integer.valueOf(c.getValue());

43 Integer newValue = Integer.valueOf(oldValue + Integer.valueOf(q));

44 fixCookie(c, newValue.toString().trim());

45 flag = false;

46 }

47 }

48 // 说明已有的cookies中没有相应的cookie，就进行添加操作

49 if (flag) {

50 addCookie(Conf.IDUONA\_CASHTICKET\_COOKIE\_STARTNAME + cashTicket.getId(), q.trim());

51 }

52

53 // ==================================================

54 // 测试用，读取所有的cookies

55 readShoppingCartFromCookie();

56 // ==================================================

57

58 write("success");

59 } else {

60 // 如果用户登录，说明session存在user,这时就持久化购物车信息

61 CashTicket cashTicketTemp = cashTicketService.get(cashTicket.getId());

62 if (shoppingCartService.isExistUserAndCashTicket(getUserInSession(), cashTicketTemp)) {

63 ShoppingCart oldShoppingCart = shoppingCartService.getByUserAndCashTicket(getUserInSession(), cashTicketTemp);

64 oldShoppingCart.setAmount(oldShoppingCart.getAmount() + Integer.valueOf(q));

65 if (shoppingCartService.update(oldShoppingCart)) {

66 write("success");

67 }

68 } else {

69 ShoppingCart shoppingCartTemp = new ShoppingCart();

70 shoppingCartTemp.setAmount(Integer.valueOf(q));

71 shoppingCartTemp.setUser(getUserInSession());

72 shoppingCartTemp.setCashTicket(cashTicketTemp);

73 shoppingCartTemp.setCreateTime(new Date());

74 shoppingCartTemp.setStatusType(StatusType.POSITIVE);

75 shoppingCartTemp.setUuid(UUID.randomUUID().toString());

76 if (shoppingCartService.save(shoppingCartTemp)) {

77 write("success");

78 }

79 }

80 }

81 }

82 }

83 }

84

85 /\*\*

86 \* 从cookie中读取购物车信息

87 \*

88 \* @throws Exception

89 \* @return

90 \*/

91 public void readShoppingCartFromCookie() throws Exception {

92 System.out.println("======================================================");

93 Cookie cookies[] = ServletActionContext.getRequest().getCookies();

94 if (cookies == null || cookies.length < 0) {

95 // System.out.println("there is no any cookie ..");

96 // 没有cookie

97 } else {

98 for (Cookie c : cookies) {

99 System.out.println("haha there are many cookies :" + c.getName() + " " + c.getValue());

100 }

101 }

102 }

103

104 /\*\*

105 \* 添加cookie操作

106 \*

107 \* @param name

108 \* cookie的name

109 \* @param value

110 \* cookie的value

111 \*/

112 public void addCookie(String name, String value) {

113 Cookie cookie = new Cookie(name.trim(), value.trim());

114 cookie.setMaxAge(2 \* 60 \* 60 \* 1000);// 设置为2个钟

115 ServletActionContext.getResponse().addCookie(cookie);

116 }

117

118 /\*\*

119 \* 更新cookie操作

120 \*

121 \* @param c

122 \* 要修改的cookie

123 \* @param value

124 \* 修改的cookie的值

125 \*/

126 public void fixCookie(Cookie c, String value) {

127 c.setValue(value.trim());

128 c.setMaxAge(2 \* 60 \* 60 \* 1000);// 设置为2个钟

129 ServletActionContext.getResponse().addCookie(c);

130 }

131

132 /\*\*

133 \* 当用户登录的时候，持久化cookie中的购物车信息，更新为本用户的购物车信息

134 \*/

135 public void peristShoppingCartWhenUserLogin(User user) {

136 if (null != user) {

137 Cookie cookies[] = ServletActionContext.getRequest().getCookies();

138 if (cookies != null) {

139 for (Cookie c : cookies) {

140 if (c.getName().startsWith(Conf.IDUONA\_CASHTICKET\_COOKIE\_STARTNAME)) {

141 // 获取cookie的名称："iduona\_cashTicket\_45" 和 cookie的值： "21"

142 String name = c.getName();

143 Integer amount = Integer.valueOf(Integer.valueOf(c.getValue())+Integer.valueOf(q));

144 Integer ct\_id = Integer.valueOf(name.substring(name.lastIndexOf("\_") + 1));

145 CashTicket temp = cashTicketService.get(ct\_id);

146 ShoppingCart shoppingCartTemp = new ShoppingCart();

147 if (null != temp) {

148 if (shoppingCartService.isExistUserAndCashTicket(user, temp)) {

149 // 进行更新操作

150 ShoppingCart oldShoppingCart = shoppingCartService.getByUserAndCashTicket(user, temp);

151 oldShoppingCart.setAmount(amount);

152 shoppingCartService.update(oldShoppingCart);

153 } else {

154 // 否则进行保存记录

155 shoppingCartTemp.setAmount(amount);

156 shoppingCartTemp.setUser(user);

157 shoppingCartTemp.setCashTicket(temp);

158 shoppingCartTemp.setCreateTime(new Date());

159 shoppingCartTemp.setStatusType(StatusType.POSITIVE);

160 shoppingCartTemp.setUuid(UUID.randomUUID().toString());

161 shoppingCartService.save(shoppingCartTemp);

162 }

163 }

164 }

165 }

166 // 移除所有的现金券cookies

167 removeAllCookies();

168 }

169 }

170 }

171

172 /\*\*

173 \* 移除所有的现金券cookies操作

174 \*/

175 public void removeAllCookies() {

176 Cookie cookies[] = ServletActionContext.getRequest().getCookies();

177 if (cookies == null || cookies.length < 0) {

178 // 没有cookie

179 System.out.println("there is no any cookie ..");

180 } else {

181 System.out.println("开始删除cookies..");

182 for (Cookie c : cookies) {

183 if (c.getName().startsWith(Conf.IDUONA\_CASHTICKET\_COOKIE\_STARTNAME)) {

184 c.setMaxAge(0);// 设置为0

185 ServletActionContext.getResponse().addCookie(c);

186 }

187 }

188 }

189 }

[复制代码](javascript:void(0);)

这是部分代码....

效果：

用户没有登录的情况下

用户登录了以后：

数据库里面的情况：

登录前数据

 I'm Hongten