

# 一、下面就是实现删除的功能

## 1.首先想到的就是要有一个 连接

<%

ArrayList arrayList = (ArrayList)request.getAttribute("booklist");

**if**(arrayList.size()!=0){

**for**(**int** i = 0 ; i< arrayList.size() ; i++){

Book book = (Book)arrayList.get(i);

%>

<tr align=*"center"*>

<td>1</td>

<td style="width: *123px*; "><%out.println(book.getBookName()); %></td>

<td><%=book.getPrice()%></td>

<td><%=book.getPublishHouse() %></td>

<td><input type=*"text"* name=*"booknum"* values =*""*+<%=book.getShopnum() %>/></td>

<td><a href=*"*shopServlet*?id="*<%=book.getId() %>>删除</a></td>

</tr>

## 2.这样的话我们就是同样跳转到了shop界面（这个界面是和购买之后跳到的是同一个界面的，所以我们可以像在之前学的设置一个type进行接收），所以同样我们也需要在购买的连接上添加一个type

<tr align=*"center"*>

<td style="width: *123px*; "><%out.println(book.getBookName()); %></td>

<td><%out.println(book.getPrice()); %></td>

<td><%out.println(book.getPublishHouse()); %></td>

<td><a href=*"shopServlet?type=add&id="*+<%= book.getId()%>>点击购买</a></td>

</tr>

## 2.2然后就是修正我们之前做好的shopservlet界面了

//接收用户想购买的id

String id = request.getParameter("id");

String type = request.getParameter("type");

MyCar myCar = (MyCar)request.getSession().getAttribute("car");

**if**(type.equals("del")){

//根据id号删除 这个书

myCar.delBook(id);

//然后我们返回自己的购物车界面

request.getRequestDispatcher("/showmyCar").forward(request, response);

}**else** **if**(type.equals("add")){

//进行检测id的值

System.*out*.println(id);

//接收来自yan的session

//将用户的购买的id假如购物车

myCar.addBooks(id, **new** BookService().getBook(id));

//先让我们的id 通过购物车 然后通过

request.setAttribute("booklist", myCar.showMyCar());

request.setAttribute("allPrice", **new** MyCar().getAllPrice());

request.getRequestDispatcher("/showmyCar").forward(request, response);

}

## 3.上上面的想法是很好的，但是我们到了购物车界面报错了，因为我们通过上面的一步还没有穿进去东西，所以这个就是需要我们先传（我们的购物车的信息）进去再进行下一步

**if**(type.equals("del")){

//根据id号删除 这个书

myCar.delBook(id);

//然后我们返回自己的购物车界面

request.setAttribute("booklist", myCar.showMyCar());

request.setAttribute("allPrice", **new** MyCar().getAllPrice());

request.getRequestDispatcher("/showmyCar").forward(request, response);

## 4.然后我们希望在删除之后回到了删除界面但是还能够返回购物界面，添加一个链接，这个链接之后我们必须也要将数据库中的东西传播进去

**<h2><a href=*"/shop/yan.jsp?***

***name=*<%=((User)session.getAttribute("user")).getUser()**

**%>*&password=*<%=((User)session.getAttribute("user")).getPassword()**

**%>*"*></a></h2>**

## 5.在验证中的添加user 的session

**if**(userService.CheckUser(user)){

BookService bookService = **new** BookService();

//下面这个不需要加<Book>

**try**{

MyCar car = **new** MyCar();

**request.getSession().setAttribute("user",user);**

request.getSession().setAttribute("car",car);

ArrayList a = bookService.getAllBook();

request.setAttribute("books", a);

## 6.但是我们上面的出现了错误，因为每次登陆之后我们的点击返回购物页面，购物车中的东西，就会为 0（因为 MyCar car = new MyCar();）这个将购物车给清除了，所以我们在进行第二次登陆的时候要进行判断，看看是不是已经登陆过了（通过验证时候产生的session来进行验证）

**if**(request.getSession().getAttribute("user")!=**null**){

BookService bookService = **new** BookService();

ArrayList a = bookService.getAllBook();

request.setAttribute("books", a);

//上面的这个是为了让我们能够 到了下一个界面进行应用

request.getRequestDispatcher("/shop.jsp").forward(request, response);

}**else**{

**if**(userService.CheckUser(user)){

BookService bookService = **new** BookService();

//下面这个不需要加<Book>

## 7.这样我们回到了 点击返回购物的界面，只要直接返回到了我们的验证界面就可以直接返回到了购物的界面了

<h2><a href=*"/shop/yan.jsp"*></a></h2>

# **8.然后我们要做的就是在自己的购物车中的数量的一栏中，输入我们想要购买的数量进行更新，首先我们希望的就是得到我们所修改的书的id**

（因为我们不可以通过submit按钮来的到我们想要的id因为submit只能得到一个id（而我们想要的是所有的id））这个时候我们可以设置一个影藏的标签用来存储id

<td><%=book.getId()%><input type=*"hidden"*

value=<%=book.getId()%>

name=*"id"*></td>

8.2然后我们的操作就是通过更新按钮 界面跳转出去，在跳转回来

<form method=*"post"* action=*"shopServlet?type=update"*>

8.3然后我们到了shopServlet?界面中进行接收，然后进行处理

}**else** **if**(type.equals("update")){

**MyCar myCar = (MyCar)request.getSession().getAttribute("car");**

//通过id构成一个数组

String booksid[] = request.getParameterValues("id");

//将书的数量通过num狗狗成一个数组

String booksnum[] = request.getParameterValues("num");

**for**(**int** i = 0 ; i < booksid.length ; i++)

{

//这样就开始更新购物车里面的东西了

myCar.updatebooks(booksid[i], booksnum[i],**new** BookService().getBook(booksid[i]) );

}

//然后我们返回自己的购物车界面

request.setAttribute("booklist", myCar.showMyCar());

request.setAttribute("allPrice", **new** MyCar().getAllPrice());

request.getRequestDispatcher("/showmyCar").forward(request, response);

}

8.4然后我们添加在myCar中的更新的的功能

}

//更新书籍

**public** **void** updatebooks(String id,String nums,Book book){

//通过购物车的到book

book = hm.get(id) ;

book.setShopnum(Integer.*parseInt*(nums));

}

//清空购物车

**public** **void** CleanBooks(){

hm.clear();

}

# 9.然后我们我们基本的功能已经实现了，开始提交我们的订单了,这个时候我们可以新建一个servlet，用来提交订单，然后发送

**public** **class** my **extends** HttpServlet {

//用来接收哦我们的订单

**public** **void** doGet(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

MyCar myCar = (MyCar) request.getSession().getAttribute("myacar");

**float** allprices = myCar.getAllPrice();

ArrayList books = myCar.showMyCar();

request.setAttribute("books", books);

request.setAttribute("allprices", allprices);

//然后跳转到我们希望显示的界面

request.getRequestDispatcher("Mydingdan").forward(request, response);

}

**public** **void** doPost(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException {

**this**.doGet(request, response);

}

}

10.然后我们显示我们的信息

<table>

<tr><td colspan=*"2"*>用户购物车中的东西</td></tr>

···<tr><td>id</td><td>书名</td><td>价格</td><td>数量</td></tr>

<%

ArrayList arrayList = (ArrayList)request.getAttribute("bookst");

**if**(arrayList.size()!=0){

**for**(**int** i = 0 ; i< arrayList.size() ; i++){

Book book = (Book)arrayList.get(i); %>

<tr align=*"center"*><td><%=book.getId()%></td> <td style="width: *123px*; "><%out.println(book.getBookName()); %></td>

<td><%=book.getPrice()%></td>

<td><%=book.getPublishHouse() %></td>

<td><input type=*"text"* name=*"booknum"* value =*""*+<%=book.getShopnum() %>/></td>

</tr>

<tr align=*"center"*>

<td colspan=*"6"*>购物总价格${allPrice}</td>

<%

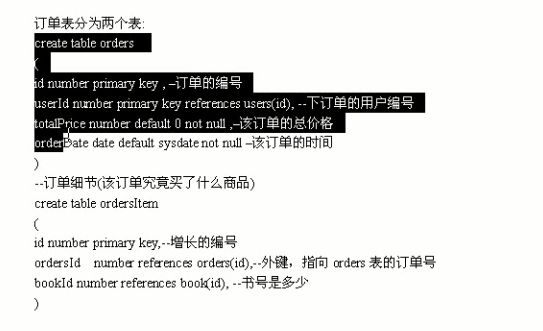
}

%>

# 11.然后就是我们开始将订单写到我们的数据库中去确认订单（因为我们要保存这个订单（防止用户发生意外））



11.1我们的订单的表要分成两个表



# 11.2然后就是开始创建了

## 一、下面这个表示关于用户的，包含用户的一些信息（id,uid(下单的用户编号),totalPrice,orderDate）

create table orders(

id int primary key,

uid int not null, 下单用户的编号

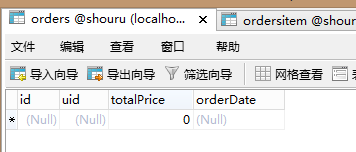
//下面这个表示userid是外键（其实他的名字没什么太大的用处），但是他指的是uid 指向主表中的id

constraint userId foreign key (uid) references lu(id),

totalPrice int default 0 not null,

orderDate date not null

);



## 二、下面这个就是订单细节了

（id, ordersid，用户的订单编号（使用外键相连接），bookid 用户书的编号（使用外键相连接），booknum以及书的量）

create table ordersItem(

id int primary key,

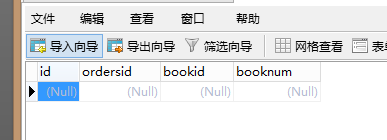
ordersid int , 与用户的订单的id 形成一致

constraint ordersids foreign key(ordersid) references orders(id),

bookid int , 用户的书的编号

constraint bookids foreign key(bookid) references book(id)

booknum int);



# 12，确定订单的按钮应该设立一个JavaScript函数用来跳转我们的界面了（这个在上一页和下一页的时候见过）

funtion gosubmitOrders(){

window.location.href=”/submitOrders”;

}

# 13.点击确认订单就会创建数据库，将订单写入到数据库中

# 1.确认了订单，势必要走到一个jsp界面中进行先显示用户下了订单，并给购物公司提交申请了

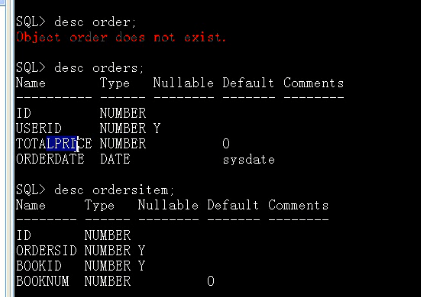
<body>

<h1>提交订单完毕</h1>

<font>给你发送了电子邮件</font>

</body>

## 2然后我们在新建一个submitOrdersy用来处理订单中的东西(但是千万记得建立一个java文件来处理相应数据库之间的联系orderService)



# 3.下面就是我们对于将用户的信息保留在数据库中，

**public** **class** OrderService {

Connection ct;

PreparedStatement ps;

ResultSet rs;

//用来处理两张表之间的联系

//要下订单，需要穿什么参数过来 很显然是 购物车 和用户

**public** **void** submitorder(MyCar myCar, User user) **throws** SQLException{

**try** {

String sql = "insert into orders orders\_seq,?,?system";

//因为这个逻辑比价复杂所以我们不是用sqlhelper 里面的东西了

ct =SqlHelper1.*getConnection*();

//为了保证我们的订单号是稳定的，我们必须将事物隔离基本升,为了得到详细表中的与这个表关联的id

ct.setAutoCommit(**false**);

ct.setTransactionIsolation(Connection.*TRANSACTION\_SERIALIZABLE*);

ps = ct.prepareStatement(sql);

ps.setInt(1, user.getId());

ps.setFloat(2, myCar.getAllPrice());

ps.executeUpdate();

//这样我们的第一张表就建立好了，下一步就是讲我们的 第二张细节表建立好了

//第二张表中需得到我们的第一张表的订单号（下面这个是如何从刚刚建立的表中得到他地 订单号）

sql = "select orders\_seq.currval from orders"

ps = ct.prepareStatement(sql);

rs = ps.executeQuery();

**int** orderid = 0; //用户的订单编号

**if**(rs.next()){

orderid = rs.getInt(1); //这样就取出来我们的 用户的订单那编号好了

}

//然后就是讲用户里面的东西全部取出来，然后交个我们的公司

ArrayList arrayList = myCar.showMyCar();

**if**(arrayList.size()!=0){

Book book = (Book)arrayList.get(0);

sql = "insert into ordersitem values(ordersitem\_seq,?,?,?)";

ps = ct.prepareStatement(sql);

ps.executeUpdate();

ps.setInt(1, orderid);

ps.setInt(2, book.getId());

ps.setInt(2, book.getShopnum());

ps.executeUpdate();

}

//将数据整体提交ct.commit();

} **catch** (Exception e) {

ct.rollback();

}**finally**{

}

}

}

4.然后就是到了我们的jsp文件中调用上面的类让他完成操作

**public** **class** SubmitOrders **extends** HttpServlet {

//这个是用来处理用户订单的请求

**public** **void** doGet(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

OrderService orderService = **new** OrderService();

MyCar myCar = (MyCar)request.getSession().getAttribute("mycar");

User user = (User)request.getSession().getAttribute("user");

**try** {

orderService.submitorder(myCar, user);

然后跳转转到发送邮件的界面

requset.getRequsetdispatcher(“Shuju”).

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}