天猫整站J2EE

# 一、基础

### 1.1项目简介

### 1.1.1基于J2EE整套技术 本项目没有使用SSH,SSM框架，而是使用J2EE整套技术来作为解决方案，实现各种业务场景。在项目中借助反射等技术，采用了精妙的设计模式，开发效率并不低于使用框架。

##### 1.1.2 典型场景

一个完整的J2EE应用包含的各种典型商务应用场景的需求，展示相应的解决思路与办法

### 1.2 **开发流程**

##### 1.2.1技术准备

Java:java基础和java中级的内容

前端：html，css，js，jquery

J2EE：tomcat，servlet，jsp，filter

数据库：mysql

##### 1.2.2 开发流程

模仿天猫整站是一个比较大的项目，将按照商业项目的开发流程有条不紊的一一展开：  
1. 需求分析  
首先确定要做哪些功能，需求分析包括[前台](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-898/898.html)和[后台](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-941/941.html)。   
前台又分为单纯要展示的那些功能-[需求分析-展示](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-898/898.html)，以及会提交数据到服务端的哪些功能-[需求分析-交互](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-940/940.html)。  
  
2. 表结构设计  
接着是表结构设计，表结构设计是围绕功能需求进行，如果表结构设计有问题，那么将会影响功能的实现。除了[表与表关系](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-944/944.html)，[建表SQL语句](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-945/945.html)之外，为了更好的帮助大家理解表结构以及关系，还特意把[表与页面功能](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-947/947.html)一一对应起来  
  
3. 原型  
接着是界面原型，与客户沟通顺畅的项目设计流程里一定会有原型这个环节。 借助界面原型，可以低成本，高效率的与客户达成需求的一致性。 同样的，原型分为了[前台原型](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-948/948.html)和[后台原型](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-971/971.html)。  
  
4. 实体类设计  
接着开始[实体类的设计与开发](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-972/972.html)，实体类不仅仅是对数据库中的表的一一映射，同时还需要兼顾对业务功能的支持。   
  
5. DAO类设计  
然后是[DAO类的设计](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-983/983.html)，除了进行典型的ORM支持功能之外，也需要提供各种业务方法。  
  
6. 业务类介绍  
讲解的[业务类](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-995/995.html)的概念，以及为什么在本项目中没有包含这一层  
  
7. 后台-分类管理  
接下来开始进行功能开发，按照模块之间的依赖关系，首先进行[后台-分类管理](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-996/996.html)功能开发。严格来说，这是开发的第一个功能，所以讲解的十分详细，不仅提供了[可运行的项目](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-997/997.html)，还详细解释了其中用到的[设计模式](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-998/998.html)，[JSP包含关系](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1006/1006.html)，以及每个具体的功能： [查询](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-999/999.html)，[分页](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1000/1000.html)，[增加](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1001/1001.html)，[删除](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1002/1002.html)，[编辑](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1003/1003.html)，[修改](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1004/1004.html)。 把每个细节都掰的很细，可以更好的理解，消化和吸收。 在把[后台-分类管理](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-996/996.html) 吃透之后，后续的其他后台管理功能，做起来就会更加顺畅。  
  
8. 后台-其他管理  
在把[后台-分类管理](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-996/996.html) 消化吸收之后，就可以加速进行 [后台其他页面](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1008/1008.html)的学习。  
  
9. 前台-首页  
前台也包括许多功能， 与[后台-分类管理](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-996/996.html)类似的，首先把[前台-首页](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1015/1015.html)这个功能单独拿出来，进行精讲。[前台-首页](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1015/1015.html) 消化吸收好之后，再进行其他前台功能的开发。  
  
10. 前台无需登录  
从前台模块之间的依赖性，以及开发顺序的合理性来考虑，把前台功能分为了 [无需登录](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1022/1022.html) 即可使用的功能，和[需要登录](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1029/1029.html) 才能访问的功能。 建立在前一步[前台-首页](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1015/1015.html)的基础之上，开始进行一系列的[无需登录](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1022/1022.html)功能开发。  
  
11. 前台需要登录  
最后是[需要登录的前台功能](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1029/1029.html)。 这部分功能基本上都是和购物相关的。 因此，一开始先把[购物流程](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1029/1029.html) 单独拿出来捋清楚，其中还特别注明了[购物流程环节与表关系](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1029/1029.html#step3954)，这样能够更好的建立对前端购物功能的理解。随着这部分功能的开发，就会进入订单生成部分，在此之前，先准备了一个 [订单状态图](http://how2j.cn/k/tmall-j2ee/tmall-j2ee-1043/1043.html#step3978)，在理解了这个图之后，可以更好的进行订单相关功能的开发。  
  
12. 总结  
最后总结整个项目的项目结构，都实现了哪些典型场景，运用了哪些设计模式，把学习到的知识都沉淀下来，转换，消化，吸收为自己的技能。

# 二、需求分析

### 2.1概述

以下需求都是基于[模仿天猫整站](http://how2j.cn/tmall)整理出来的，主要分为3类：

2.1.1 前端展示   
在前端页面上显示数据库中的数据，如首页，产品页，购物车，分类页面等等。

##### 2.1.2 前端交互

这里的交互指的是通过POST,GET等http协议，与服务端进行同步或者异步数据交互。 比如购买，购物车，生成订单，登录等等功能。

##### 2.1.3 后台功能

比如分类管理，分类属性管理，产品管理，产品图片管理，用户管理，订单管理等等。

### 2.2前台展示

##### 2.2.1 首页

步骤 1 : 先看完整页面效果

步骤 2 : 在横向导航栏上提供4个分类连接

步骤 3 : 在纵向导航栏上提供全部17个分类连接

步骤 4 : 当鼠标移动到某一个纵向分类连接的时候，显示这个分类下的推荐商品

步骤 5 : 按照每种分类显示5个商品的方式，显示所有17种分类

##### 2.2.2 产品页

步骤 1 : 显示分辨率为950x100的当前商品对应的分类图片

步骤 2 : 显示本商品的5个单独图片

步骤 3 : 商品的基本信息，如标题，小标题，价格，销量，评价数量，库存等

步骤 4 : 商品详情

步骤 5 : 评价信息

步骤 6 : 5张商品详细图片

##### 2.2.3 分类页

步骤 1 : 显示分辨率为950x100的当前分类图片

步骤 2 : 显示本分类下的所有产品

##### 2.2.4 搜索结果页

步骤 1 :显示满足查询条件的商品

##### 2.2.5 购物车查看页

步骤 1 :在购物车中显示订单项

##### 2.2.6 结算页

步骤 1 :在结算页面显示被选中的订单项

##### 2.2.7 确认支付页

步骤 1 :确认支付页面显示本次订单的金额总数

##### 2.2.8 支付成功页

步骤 1 :付款成功时，显示本次付款金额

##### 2.2.9 我的订单页

步骤 1 :显示所有订单，以及对应的订单项

##### 2.2.10 确认收货页

步骤 1 : 显示订单项内容

步骤 2 : 显示订单信息，收货人地址等

##### 2.2.11 评价页

步骤 1 : 显示要评价的商品信息，商品当前的总评价数

步骤 2 : 评价成功后，显示当前商品所有的评价信息

##### 2.2.12 页头信息展示

步骤 1 : 未登录状态

步骤 2 : 已登录状态

### 2.3前台交互

##### 2.3.1 分类页排序

步骤 1 : 按照综合，人气，新品，销量，价格排序

步骤 2 : 进行价格筛选，这个不算，这个是做在浏览器端的，并不是和服务器端的交互

##### 2.3.2 立即购买

步骤 1 : 在产品页，未登录状态时候，点击立即购买

步骤 2 : 在产品页，如果已经登录，点击购买

##### 2.3.3 加入购物车

步骤 1 : 在未登录状态，点击加入购物车

步骤 2 : 在已登录状态，点击加入购物车

##### 2.3.4 调整订单项数量

步骤 1 :在购物车页面，调整订单项数量

##### 2.3.5 删除订单项

步骤 1 :在购物车页面，删除订单项

##### 2.3.6 生成订单

步骤 1 :在结算页面，提交订单

##### 2.3.7 订单页功能

步骤 **1** :我的订单页面：

订单页面上有3个按钮，付款，确认收货和评价，这些按钮都没有提交数据到服务端，而是提交到下一个页面的二次操作，才提交数据到服务端的。

##### 2.3.8 确认付款

点击确认支付，提交本信息到服务端，服务端修改订单中的支付日期

##### 2.3.9 确认收货

确认收货页面，点击确认支付，提交数据到服务端，并且修改订单中的确认收货日期

##### 2.3.10 提交评价信息

提交评价信息到服务端

##### 2.3.11 登录

输入正确账号密码，登录成功后跳转到首页，失败提示错误信息

##### 2.3.12 注册

步骤 1 : 注册页面

##### 2.3.13 退出

退出登录

##### 2.3.14 搜索

提交关键字到服务端，服务端查询数据库，并返回匹配的产品

##### 2.3.15 前台需求列表

### 2.4后台管理

##### 2.4.1 分类管理

步骤 1 : 分页查询

步骤 2 : 新增分类

步骤 3 : 编辑修改

步骤 4 : 删除

##### 2.4.2 属性管理

步骤 1 : 属性概念

这里的属性，指的是产品属性。假定同一类产品，都有相同的属性

步骤 2 : 分页查询

步骤 3 : 新增属性

步骤 4 : 编辑修改

步骤 5 : 删除

##### 2.4.3 产品管理

步骤 1 : 分页查询

步骤 2 : 新增产品

步骤 3 : 编辑修改

步骤 4 : 删除

##### 2.4.4 产品图片管理

步骤 1 : 产品图片概念

一件产品，对应多条 单个图片  
一件产品，对应多条 详情图片

步骤 2 : 产品图片管理

##### 2.4.5 产品属性管理

步骤 1 : 产品属性值设置

根据属性管理中的讲解，一种产品所具备的属性，在其对应的分类中进行了维护。那么，要修改产品的这些属性值，就在本页面进行。

这里采用异步提交方式，编辑即修改,修改成功用绿色边框表示

##### 2.4.6 用户管理

用户管理就提供了一个分页查询，做的比较简约，几个原因：  
1. 用户是不能删除的  
2. 用户信息的修改，应该交由前端由客户自己修改，比如密码，基本信息等  
3. 用户的增加，是前端注册导致的，后台不负责用户的增加

##### 2.4.7 订单管理

步骤 1 : 订单分页查询

步骤 2 : 查看详情

步骤 3 : 发货

##### 2.4.8 后台需求列表

# 三、表结构设计

### 3.1概述

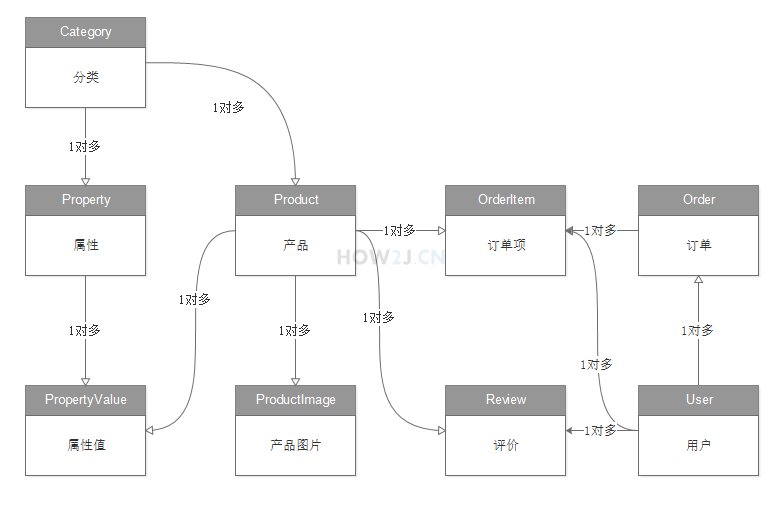
在需求分析结束之后，接下来要做的不是马上开发功能，而是进行表结构的设计。

### 3.2数据库

创建数据库:tmall。并且将数据库的编码设置为utf8，便于存放中文

### 3.3 表与表关系

##### 3.3.1 表关系图



##### 3.3.2 表

|  |  |  |
| --- | --- | --- |
| 表名 | 中文含义 | 介绍 |
| Category | 分类表 | 存放分类信息，如女装，平板电视，沙发等 |
| Property | 属性表 | 存放属性信息，如颜色，重量，品牌，厂商，型号等 |
| Product | 产品表 | 存放产品信息，如LED40EC平板电视机，海尔EC6005热水器 |
| PropertyValue | 属性值表 | 存放属性值信息，如重量是900g,颜色是粉红色 |
| ProductImage | 产品图片表 | 存放产品图片信息，如产品页显示的5个图片 |
| Review | 评论表 | 存放评论信息，如买回来的蜡烛很好用，么么哒 |
| User | 用户表 | 存放用户信息，如斩手狗，千手小粉红 |
| Order | 订单表 | 存放订单信息，包括邮寄地址，电话号码等信息 |
| OrderItem | 订单项表 | 存放订单项信息，包括购买产品种类，数量等 |

##### 3.3.3表关系

从最上面的表关系图中可以看到，有关系的表之间，都是一对多关系(反过来就是多对一关系)，并没有一对一关系，和多对多关系。

### 3.4 建表sql语句

##### 3.4.1 建表顺序

在建表过程中，会设置外键约束，所以表和表之间有依赖关系。 因此会先建立被外键指向的表，比如User,Category，然后再是其他表

##### 3.4.2 表的字段

1.用户表user

主键Id,name用户名,password登陆密码

2.分类表category

主键id，name分类名

3.属性表property

主键id，名称name，外键cid指向category的id

4.产品表

主键id， name产品名称，subTitle小标题，orignalPrice原始价格，promotePrice优惠价格，stock库存，createDate创建日期，外键cid指向category的id

5.属性值表propertyvalue

主键id，属性值value，外键ptid指向property的id，外键pid指向product的id

6.产品图片表

主键id，type图片文件，外键pid指向product的id

7.评价表

主键id，content评价内容，createDate创建日期，外键pid指向product的id，外键uid指向用户表的id

8.订单表

orderCode：订单号  
address:收货地址  
post: 邮编  
receiver: 收货人信息  
mobile: 手机号码  
userMessage: 用户备注信息  
createDate: 订单创建日期  
payDate: 支付日期  
deliveryDate: 发货日期  
confirmDate：确认收货日期  
status: 订单状态  
外键uid，指向用户表id字段

9.订单项表

外键pid，指向产品表id字段  
外键oid，指向订单表id字段  
外键uid，指向用户表id字段  
number字段表示购买数量

# 四、实体类

### 4.1概述

所谓的实体类，就是对于数据库中的表的互相映射的类。 这是一种[ORM](http://how2j.cn/k/jdbc/jdbc-orm/391.html)的设计思想，即一个对象，对应数据库里的一条记录

### 4.2所有实体类

对于一对多的表，在“一”的类中添加“多”的类集合，在“多”的类中添加“一”的类。

# 五．DAO类设计

### 5.1 概述

DAO是Data Access Object的缩写，这些类专门用于进行数据库访问的操作。

1. 首先讲解两个工具类，DBUtil和DateUtil，因为在DAO类中会用到这两个工具类

2. 接着按照依赖顺序，逐个讲解不同的DAO类以及其中的方法

### 5.2 工具类

1.DBUtil类初始化驱动，并返回数据库连接

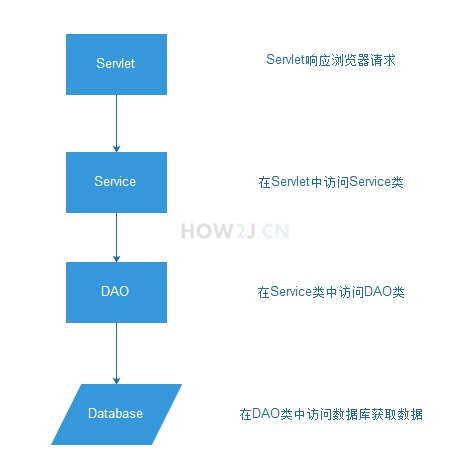
2.DateUtil用于实体类和数据库中日期格式的转换

### 5.3 DAO类

# 六．Service

### 6.1 通常的设计流程

作为J2EE web 应用，一般会按照如图所示的设计流程进行  
Servlet -> Service（业务类） -> DAO -> database   
当浏览器提交请求到tomcat web 服务器的时候，对应的servlet的doGet/doPost方法会被调用，接着在servlet中调用Service类，然后在Service类中调用DAO类，最后在DAO中访问数据库获取相应的数据。



### 6.2 不使用Service的原因

在本模仿天猫整站-J2ee 版本中，不使用Service这一层。 原因是在DAO进行了比较详细的设计，已经提供了很好的支持业务的方法。

# 七．后台

### 7.1原型-后端

7.1.1 概述

本项目里的后端又不仅仅是CRUD维护管理页面，还有一些其他略复杂的功能，所以还是把这些原型界面都做出来。 后续开发功能的时候，就可以基于这些原型界面进行。

##### 7.1.2 分类管理



##### 7.1.3 属性管理



##### 7.1.4 产品管理



##### 7.1.5 产品图片管理



##### 7.1.6 产品属性管理



##### 7.1.7 用户管理



##### 7.1.8 订单管理



### 7.2 后台-分类管理

##### 7.2.1 概述

开发整站的顺序，通常来说还是按照依赖性来进行，前端需要的数据，都要先通过后台的功能维护在数据库中，才可以拿到。 所以，先进行后台功能的开发，然后再是前台功能的开发。   
 后台在系统设计的时候，并不是简单的每个功能对应一个Servlet，而是使用了反射的技术，结合过滤器Filter进行了封装，使得开发配置以及维护成本降低了很多。

##### 7.2.2 静态资源

接下来，解析这个可运行项目。 首先从最简单的静态资源开始。

##### 7.2.3 Filter配合Servlet

步骤一：每种实体类，对应了一个Servlet，而不是对应了5个，这样首先从Servlet数量上来讲，就大大的减少了。原理流程图：

1.假设访问路径是 http://127.0.0.1:8080/tmall/admin\_category\_list  
2. 过滤器BackServletFilter进行拦截，判断访问的地址是否以/admin\_开头  
3. 如果是，那么做如下操作  
3.1 取出两个下划线之间的值 category  
3.2 取出最后一个下划线之后的值 list  
3.3 然后根据这个值，服务端跳转到categoryServlet，并且把list这个值传递过去  
4. categoryServlet 继承了BaseBackServlet，其service方法会被调用。 在service中，借助反射技术，根据传递过来的值 list，调用对应categoryServlet 中的方法list()  
5. 这样就实现了当访问的路径是 admin\_category\_list的时候，就会调用categoryServlet.list()方法这样一个效果

**步骤二：**代码讲解-BackServletFilter:

1. 首先在web.xml配置文件中，让所有的请求都会经过BackServletFilter

<url-pattern>/\*</url-pattern>

2. 还是假设访问的路径是：

<http://127.0.0.1:8080/tmall/admin_category_list>

3. 在BackServletFilter 中通过request.getRequestURI()取出访问的uri: /tmall/admin\_category\_list  
4. 然后截掉/tmall，得到路径/admin\_category\_list  
5. 判断其是否以/admin开头  
6. 如果是，那么就取出两个\_之间的字符串，category，并且拼接成/categoryServlet，通过服务端跳转到/categoryServlet  
7. 在跳转之前，还取出了list字符串，然后通过request.setAttribute的方式，借助服务端跳转，传递到categoryServlet里去

**步骤3**：代码讲解-CategoryServlet和BaseBackServlet

服务端跳转/categoryServlet就到了CategoryServlet这个类里  
1. 首先CategoryServlet继承了BaseBackServlet，而BaseBackServlet又继承了HttpServlet  
2. 服务端跳转过来之后，会访问CategoryServlet的doGet()或者doPost()方法  
3. 在访问doGet()或者doPost()之前，会访问service()方法  
4. BaseBackServlet中重写了service() 方法，所以流程就进入到了service()中  
5. 在service()方法中有三块内容  
5.1 第一块是获取分页信息  
5.2 第二块是根据[反射](http://how2j.cn/k/reflection/reflection-class/108.html)访问对应的方法  
5.3 第三块是根据对应方法的返回值，进行服务端跳转、客户端跳转、或者直接输出字符串。  
6. 第一块和第三块放在后面讲解，这里着重讲解第二块是根据反射访问对应的方法  
6.1 取到从BackServletFilter中request.setAttribute()传递过来的值 list  
6.2 根据这个值list，借助[反射机制](http://how2j.cn/k/reflection/reflection-method/109.html)调用CategoryServlet类中的list()方法  
这样就达到了CategoryServlet.list()方法被调用的效果

##### 7.2.4 JSP包含关系

步骤一：分类查询对应的JSP文件

分类查询对应的JSP文件是listCategory.jsp，用到了4个公共包含文件：

1. <%@include file="../include/admin/adminHeader.jsp"%>  
2. <%@include file="../include/admin/adminNavigator.jsp"%>  
3. <%@include file="../include/admin/adminPage.jsp"%>  
4. <%@include file="../include/admin/adminFooter.jsp"%>

步骤二：

每个后台页面都在开始使用了adminHeader.jsp

1. html5
2. jsp指令
3. 引入JSTL
4. 引入js和css
5. 预先定义一些判断输入框的js函数
6. 对于删除链接，需要进行确认操作

步骤三：adminNavigator.jsp

导航部分，包括一张图片和4个链接

##### 7.2.5 查询

步骤1：MVC设计思想

根据 MVC设计模式的思想，做J2EE web应用，从MVC的角度来看，就是把不同的数据显示在不同的页面上。数据就是 模型 ( bean, dao )、页面就是 视图 ( jsp )、控制不同的模型显示在不同的视图上，这件事，就是由控制器来完成 ( servlet )。

所以分类管理，从MVC的角度来看，就是把多条分类Category数据放在一个集合里， 让listCategory.jsp 这个视图去显示出来。

步骤2：list()方法

在list()方法里做的事情是，取出数据，并且交由jsp显示。

1. 通过categoryDAO取得数据集合 cs
2. 通过request.setAttribute 放在 “thecs" 这个key中，为后续服务端跳转到jsp之后使用

3. return "admin/listCategory.jsp";

步骤3：服务端跳转

list()方法跳转到jsp页面

步骤4：listCategory.jsp

作为视图，担当的角色是显示数据。所以关键就是从第44行开始，借助JSTL的c:forEach标签遍历从CategoryServlet的list() 的request.setAttribute("thecs", cs); 传递过来的分类集合。

##### 7.2.6 分页

步骤1：page.java

Page这个类专门为分页提供必要信息。

步骤2：获取分页参数

Service方法获取网页上传递来的开始位置，以及每页需要显示的数量。如果传递数据过来，开始位置取0，每页显示的数量取默认值：5.

步骤3：基于分页参数，获取数据

通过反射调用list()方法，list调用DAO获取分页数据。

步骤4： 为page对象设置总数

在分页显示的时候，依据这个总数来判断一共有多少页面，最后一页是多少。

##### 7.2.7 增加

步骤1：增加的部分是坐在查询结果页面的。

步骤2：提交数据会导致add()方法被调用。

步骤3：接收数据并处理：parseUpload方法获取文件流

步骤4：ImageUtil工具类：转换文件格式为jpg，改变文件大小

步骤5：中文问题，在过滤器中对提交的数据进行统一编码

步骤6：客户端跳转，add()方法返回@admin\_category\_list

##### 7.2.8 删除

步骤1：用于删除的超链，指向地址admin\_category\_delete,并且会传递当前分类对象的id过去。

步骤2：在adminHeader.jsp中对所有的删除连接都进行了监听。

步骤 3 : 删除调用CategoryServlet.delete()方法。

##### 7.2.9 编辑

步骤1 : 编辑超链，用于编辑的超链，指向地址admin\_category\_edit,并且会传递当前分类对象的id过去。

步骤2 : 调用CategoryServlet.edit()方法。

步骤3：服务端跳转到editCategory.jsp，在editCategory.jsp页面里，获取由CategoryServlet.edit() 通过request传递过来的Category对象，获取name和id。

##### 7.2.10 修改

步骤1：编辑页面提交数据。

步骤2：CategoryServlet.update() 方法和CategoryServlet.add()方法的处理很类似，有所不同之处在于增加操作一定会提交图片，而修改不一定提交图片。

### 7.3 后台-其他管理

7.3.1概述

后台的其他管理页面与分类管理用到的设计思路类似。

7.3.2 属性管理

7.3.3 产品管理

7.3.4 产品图片管理

7.3.5 产品属性管理

7.3.6 用户管理

7.3.7 订单管理

# **八.前台**

### **8.1原型-前端**

##### 8.1.1概述

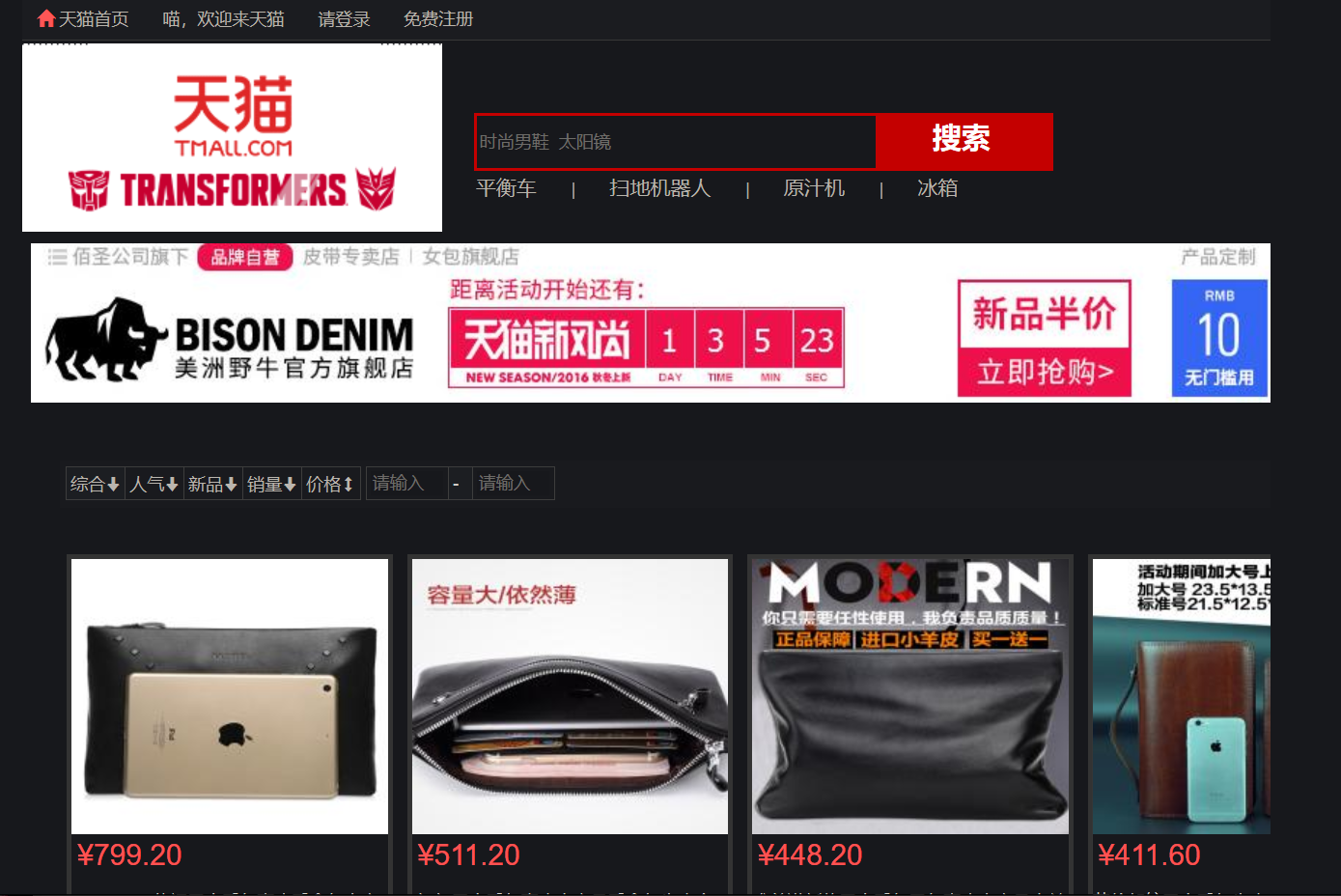
在开发功能之前，都会先有原型界面。 通常原型界面的数据不是真实的，是临时写在页面上，为用户提供一个直观的界面感受的。

关于前端的关注点，着重落在如何获取服务端的数据，显示在页面上，以及前端页面与服务端的同步/异步数据交互上。

##### 8.1.2 首页



##### 8.1.3 分类页



##### 8.1.4 产品页面



##### 8.1.5购物车页面



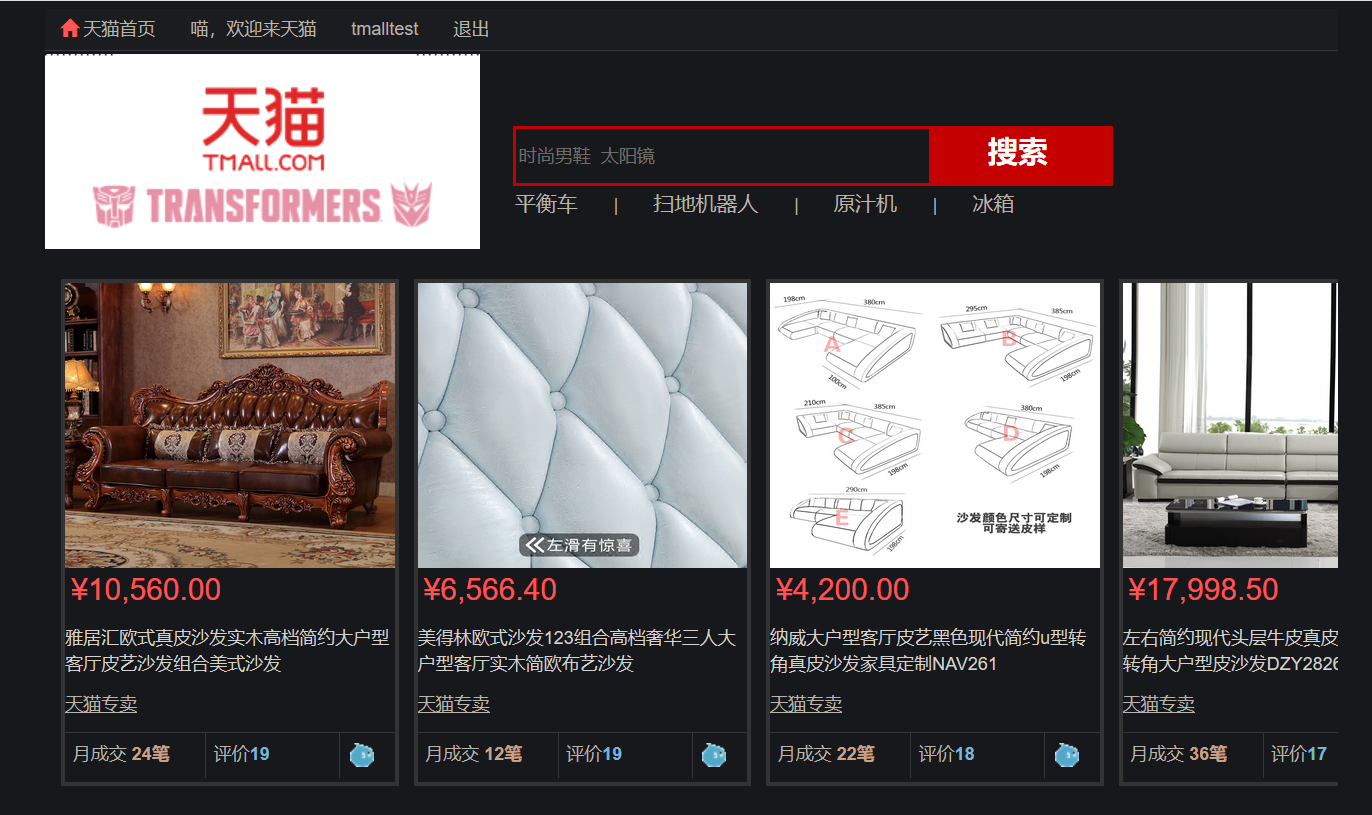
##### 8.1.6 我的订单页面



##### 8.1.7 结算页面



##### 8.1.8 查询结果页面



##### 8.1.9 支付页面



##### 8.1.10 支付成功页面



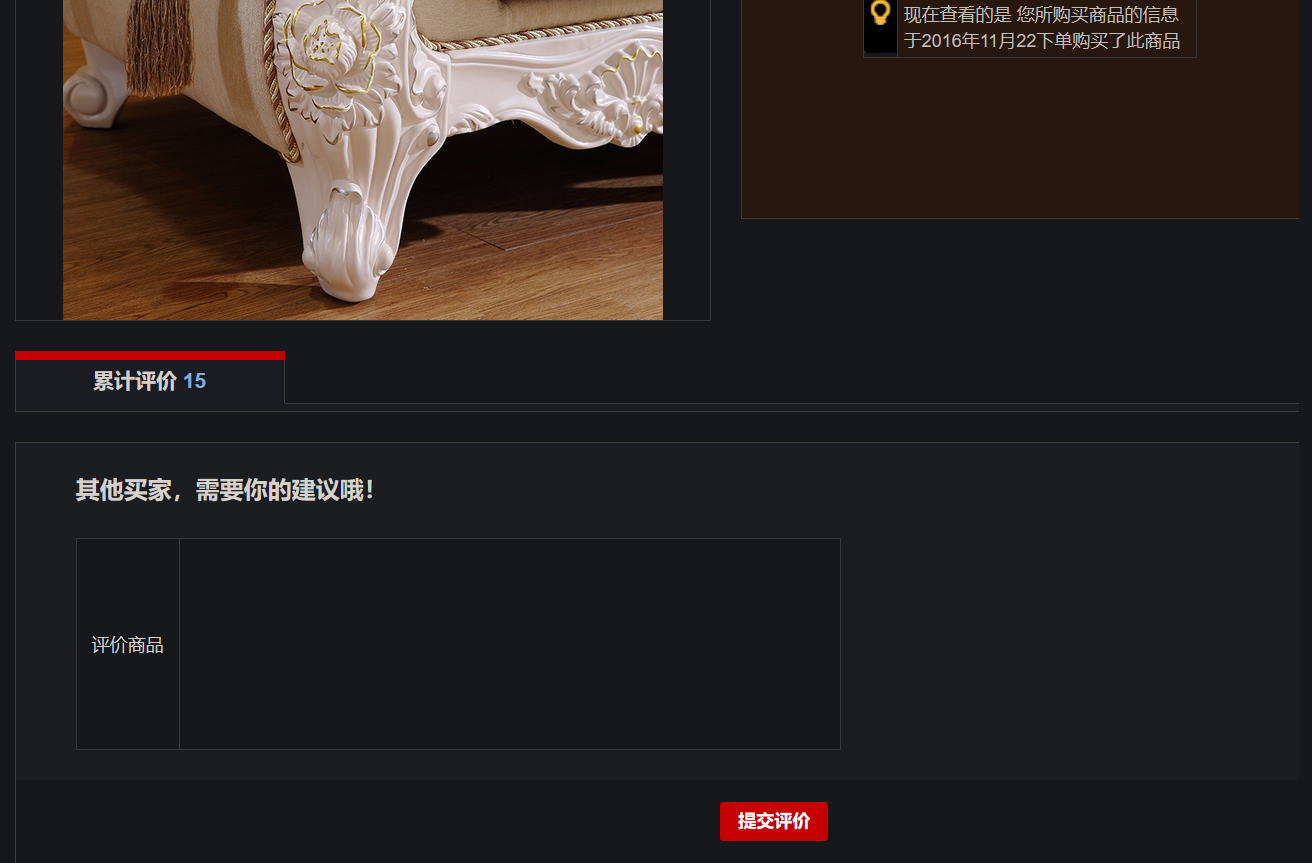
##### 8.1.11 确认收货页面



##### 8.1.12 收货成功页面



##### 8.1.13 评价页面



##### 8.1.14 登录页面



### 8.2 前台-首页

##### 8.2.1 概述

1. 前台无需登录

从前台模块之间的依赖性，以及开发顺序的合理性来考虑，把前台功能分为了 无需登录 即可使用的功能，和需要登录 才能访问的功能。 建立在前一步前台-首页的基础之上，开始进行一系列的无需登录功能开发。

2. 前台需要登录

接着是需要登录的前台功能。 这部分功能基本上都是和购物相关的。 因此，一开始先把购物流程 单独拿出来捋清楚，其中还特别注明了购物流程环节与表关系，这样能够更好的建立对前端购物功能的理解。随着这部分功能的开发，就会进入订单生成部分，在此之前，先准备了一个 订单状态图，在理解了这个图之后，可以更好的进行订单相关功能的开发

##### 8.2.2 静态资源

步骤1：图片资源

Site目录，轮播图片，分类图片，产品图片

步骤2：css,js

##### 8.2.3 Filter配合Servlet

步骤1：ForeServletFilter

与BackServletFilter相似，如果访问路径以/fore而不是以/foreServlet开头，就跳转到foreServlet。

步骤2：ForeServlet和BaseForeServlet

在ForeServlet类里，就提供了所有前端需要用到的业务方法。

BaseForeServlet与BaseBackServlet相似，不过前台没有用到分页，所以service主要是根据反射调用ForeServlet的home()方法以及根据home()的返回值进行跳转。

##### 8.2.4 home方法

步骤1：首页展示的数据

1. 在横向导航栏上提供4个分类连接   
 2. 在纵向导航栏上提供全部17个分类连接   
 3. 当鼠标移动到某一个纵向分类连接的时候，显示这个分类下的推荐商品   
 4. 按照每种分类显示5个商品的方式，显示所有17种分类  
步骤2：home方法

获取所有分类，为分类填充产品，推荐产品。

##### 8.2.5 home.jsp

步骤1：包含页面

根据前台页面的特点，我们把home.jsp设计成如代码所示。  
1. header.jsp  
 引入标准标签库，js,css，自定义javascript函数等  
2. top.jsp  
 置顶导航  
3. search.jsp  
 搜索栏  
4. homePage.jsp  
 首页业务页面  
5. footer.jsp  
 页脚

##### 8.2.6 homePage.jsp

步骤1：homepage.jsp所包含的页面关系

1. categoryAndCarousel.jsp，分类和轮播
   1. categoryMenu.jsp，竖状分类菜单
   2. productsAsideCategorys.jsp，竖状分类菜单右侧的推荐产品
   3. carousel.jsp，轮播。
2. homepageCategoryProducts.jsp

主题的17种分类以及每种分类对应的5个产品

步骤2：categoryAndCarousel.jsp

categoryAndcarousel.jsp 页面利用ForeServlet传递过来的数据，天猫国际几个字旁边显示4个分类超链，另外包含了其他3个页面：  
categoryMenu.jsp、productsAsideCategorys.jsp、carousel.jsp



### 8.3 前台-无需登录

##### 8.3.1 概述

我们把前台功能分为两部分：  
1. 无需登录   
 有一些功能，无需登录也可以使用的，比如登录，注册本身，分类页面，查询，产品页面等   
2. 需要登录   
 有一些功能，必须建立在已经登录的基础之上，比如购买，加入购物车，查看购物车，结算页面，订单页面等等。

##### 8.3.2 注册

与首页的home.jsp相仿，register.jsp也包含了header.jsp,top.jsp,footer.jsp等公共页面。

不同的是，没有使用首页的search.jsp,而是用的简单一点的 simpleSearch.jsp。

##### 8.3.3 登录

与register.jsp相仿，login.jsp也包含了header.jsp,footer.jsp等公共页面。中间是登录业务页面loginPage.jsp

##### 8.3.4 退出

清除session中的user

##### 8.3.5 产品页

product()方法用来展示产品。

productPage.jsp 又由3个页面组成  
1. imgAndInfo.jsp：单个图片和基本信息

2. productReview.jsp：评价信息

3. productDetail.jsp：详情图片

##### 8.3.6 模态登录

在未登录状态，点击购买或者加入购物车就会弹出这个模态对话框，在这个模态对话框上可以进行登录操作.

立即购买和加入购物车这两个按钮的监听是放在imgAndInfo.jsp页面中

这两个按钮都会通过JQuery的.get方法，用异步ajax的方式访问forecheckLogin，获取当前是否登录状态。如果返回的不是"success" 即表明是未登录状态，那么就会打开登录的模态窗口。

模态窗口modal.jsp被footer.jsp所包含。

##### 8.3.7 分类页

步骤1：5个Comparator比较器

分类这个页面有排序功能，所以ForeServlet之前，先准备5个Comparator比较器:

1. ProductAllComparator 综合比较器，把 销量\*评价 高的放前面

2. ProductReviewComparator 人气比较器，把 评价数量多的放前面

3. ProductDateComparator 新品比较器，把 创建日期晚的放前面

4. ProductSaleCountComparator 销量比较器，把 销量高的放前面

5. ProductPriceComparator 价格比较器，把 价格低的放前面

步骤2：ForeServlet.category()

根据cid获取分类c，为c填充产品，为产品填充销量和评价。获取参数sort，进行排序。

步骤3：categoryPage.jsp

categoryPage.jsp 里有3个内容：

1. 显示当前分类图片

<img src="img/category/${c.id}.jpg">

2. 排序条 sortBar.jsp

3. 产品列表 productsByCategory.jsp

步骤4：sortBar.jsp

sortBar.jsp即排序条，作用如下：

1. 根据sort参数判断那个排序按钮高亮
2. 每个排序按钮提交到本页面，即/forecategory,并带上参数sort

步骤5：productsByCategory.jsp

productsByCategory.jsp显示当前分类下的所有产品，通过forEach遍历c.products集合里的每个产品，并把产品标题，价格，图片，评价数，成交数打印出来。

categorycount是用于测试的，在访问地址的时候加这个参数可以控制显示的产品数量。

##### 8.3.8 搜索

Search()方法进行模糊查询，将结果返回给jsp，最终在productBySearch.jsp页面中显示出来。

### 8.4 前台-需要登录

##### 8.4.1 购物流程

步骤1：购买的业务流程

1. 登录 2. 访问产品页 3. 立即购买 4. 进入结算页面

5. 加入购物车 6. 查看购物车 7. 选中购物车中的商品

8. 又到了第4步的结算页面 9. 在结算页面生成订单

10. 付款 11. 确认收货 12. 评价

步骤2：购物流程环节与表观系

围绕购物流程最重要的两个表是OrderItem 和 Order表，关于OrderItem的业务行为：  
1. 立即购买 —— 新增 OrderItem  
2. 加入购物车 —— 新增 OrderItem  
3. 查看购物车 —— 显示未和Order关联的OrderItem  
4. 选中购物车中的商品 —— 选中OrderItem  
5. 结算页面 —— 显示选中的OrderItem  
  
6. 生成订单 —— 新增Order  
7 .付款 —— 修改Order状态  
8. 我的订单 —— 显示Order  
9. 确认收货 —— 修改Order状态

##### 8.4.2 立即购买

步骤1：buyone()

1. 获取p和user
2. 增加订单项OrderItem或调整数量
3. 跳转到结算页面/forebuy

##### 8.4.3 结算页面

步骤1：buy()

1. getParamValues获取多个oiid
2. 根据oiid将订单项放入集合ois中
3. 计算ois的价格总数total
4. ois放入session，total放入request
5. 服务端跳转到buy.jsp

步骤2：buyPage.jsp

1. 遍历出订单项集合ois中的订单数据，从立即购买跳转到结算页面只会有一件产品。
2. 显示总金额
3. 提交订单项到订单

##### 8.4.4 加入购物车

步骤1：登录之后，点击加入购物车，会使用Ajax提交数据到addCart()

步骤2：addCart()

和buyone()一样：

1. 获取pid，num
2. 从session中获取user
3. 新增订单项或只增加数量

##### 8.4.5 查看购物车页面

步骤1：cart()

1. 获取当前用户
2. 获取用户的ois
3. 跳转到cart.jsp

##### 8.4.6 登录状态Filter

准备一个过滤器，当访问那些需要登录才能做的页面的时候，进行是否登录的判断，如果不通过，那么就跳转到login.jsp页面去，提示用户登录。

新建过滤器ForeAuthFilter，

1．不需要登陆的页面

注册，登录，产品，首页，分类，查询

2.需要登陆的页面

购买行为，加入购物车，查看购物车，查看订单

##### 8.4.7 购物车页面操作

步骤1.调整订单数量，通过ajax访问/forechangOrderItem

1. 判断用户是否登录
2. 获取pid和number
3. 遍历OrderItem
4. 根据pid找到OrderItem，更新OrderItem
5. 返回success

步骤2.删除订单项，通过ajax访问/foredeleteOrderItem

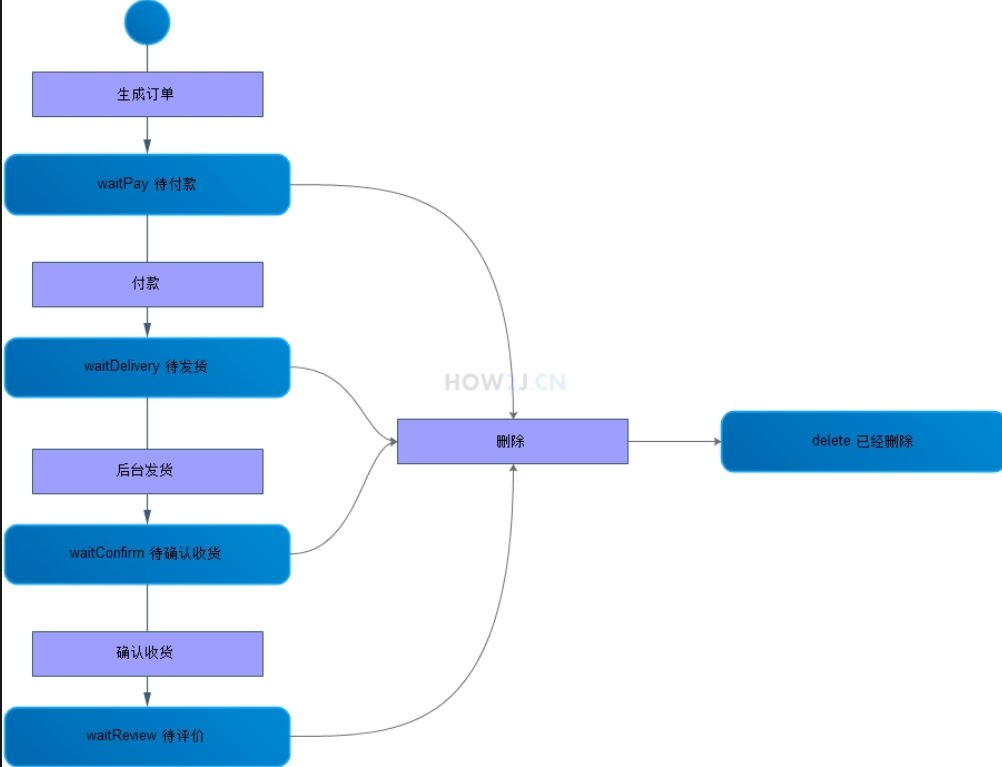
1. 判断用户是否登录

2. 获取oiid

3. 删除oiid对应的OrderItem数据

4. 返回字符串"success"

##### 8.4.8 订单状态图



1. 首先是创建订单，刚创建好之后，订单处于waitPay 待付款状态

2. 接着是付款，付款后，订单处于waitDelivery 待发货状态

3. 前两部都是前台用户操作导致的，接下来需要到后台做发货操作，发货后，订单处于waitConfirm 待确认收货状态

4. 接着又是前台用户进行确认收货操作，操作之后，订单处于waitReview 待评价状态

5. 最后进行评价，评价之后，订单处于finish 完成状态

比较特殊的是，无论当前订单处于哪个状态，都可以进行删除操作。 像订单这样极其重要的业务数据，实际上是不运行真正从数据库中删除掉的，而是把状态标记为删除，以表示其被删掉了，所以在删除之后，订单处于 delete 已删除状态

##### 8.4.9 生成订单

步骤1.提交订单调用createOrder()方法

1.获取参数创建订单对象

2.更新订单项

3.跳转到确认支付页forealipay，并带上id和总金额

步骤2：确认支付页

显示总金额，支付后跳转到forepayed页面，带上id和总金额

步骤3：支付成功页

显示订单的基本信息

##### 8.4.10 我的订单页

bought方法获取user中不是delete状态的订单集合os

bought.jsp显示订单的日期、金额等信息，显示每个订单项对应的产品

##### 8.4.11 我的订单操作

步骤1：在我的订单页面，根据订单的不同状态，可以做出如下不同的操作：

1. 付款 —— 已经生成，但是未付款

2. 确认收货 —— 通过后台发货后

3. 评价 —— 确认收货后, 进行评价

4. 删除 —— 任意状态下

步骤2：确认收货/foreconfirmPay

1. 为订单对象o填充订单项，
2. confirmPay.jsp显示订单的创建的时间等信息

步骤3：确认收货成功/foreorderConfirmed

1. 修改订单o的状态为等待评价，修改其确认支付时间
2. 更新到数据库
3. orderConfirmPage.jsp显示成功信息

步骤4：评价

步骤5：删除/foredeleteOrder

1. 修改状态，更新
2. 返回字符串success
3. boughtPage.jsp的js代码隐藏当前订单

##### 8.4.12 评价产品

步骤1：评价产品页面/foreview

1. 为订单o填充订单项
2. 获取第一个订单项的产品，显示其图片
3. 获取这个产品的评价集合
4. 为产品设置数量和销量
5. 把产品，订单和评价放在request上
6. 在reviewPage.jsp中显示产品图片，产品标题，价格，产品销量，产品评价数量，以及订单信息等。 同时还显示出了该产品所有的评价，但是默认是隐藏的

步骤2：提交评价/foredoreview

1. 获取订单o，修改状态，更新到数据库
2. 获取产品对象p，获取参数content(评价信息)，对评价信息进行转义
3. 获取当前用户，获取评价对象review
4. 为review设置评价信息等，提交到数据库
5. 跳转到/foreview：参数showonly=true，显示评价信息

# 九．总结

### 9.1 项目总结

##### 9.1.1 项目结构

1. 项目名称 tmall

2. java源代码包结构

tmall.bean 实体类

tmall.comparator 比较器

tmall.dao DAO类

tmall.filter 过滤器

tmall.servlet servlet

tmall.test 测试类

tmall.util 工具类

3. web目录

css css文件

img 图片资源

js js文件

admin 后台管理用到的jsp文件

include 被包含的jsp文件

##### 9.1.2 典型场景

1.购物车

2.订单状态流转

3.CRUD

4.分页

5.一类产品多属性配置

6．一款产品多图片维护

7. 产品展示

8.搜索查询

9.登录、注册

10.登录验证

##### 9.1.3 设计模式

1. MVC

MVC设计模式贯穿于整个后台与前台功能开发始末

2. Filter+Servlet+反射

采用Filter+Servlet+反射的设计模式，

把原本后台需要20多个Servlet的经典Servlet设计方式，精简到了7个。

把原本前台需要20多个Servlet的经典Servlet设计方式，精简到了2个。

web.xml的配置文件也大大减少，降低了开发和维护的工作量，减少了出错的几率。

3. 统一的分页查询简化开发

所有的后台都使用同一个分页机制，仅仅需要一份简化的adminPage.jsp即满足了各种分页功能的需求，简化了开发，提升了开发速度。

4. 模块化JSP设计

从大的JSP文件中，通过JSP包含关系抽象出多个公共文件，并把业务JSP按照功能，设计为多个小的JSP文件，便于维护和理解

### 9.2 改进练习

还有些功能可以继续完善。   
1. 后台管理员登陆模块   
2. 产品图片排序   
3. 前台分类下显示产品，提供分页功能   
等等