

**Web软件开发课程论文**

名称： 网上书店系统

姓名： 张 超

班级学号：111141-20141003836

**中国地质大学信息工程学院软件工程系**

**2016年 11 月**

目录

[1.系统概述 2](#_Toc226442949)

[1.1 系统概括 2](#_Toc226442950)

[1.2 系统目标 2](#_Toc226442951)

[1.2 人员计划 2](#_Toc226442952)

[1.3 开发环境与工具 3](#_Toc226442953)

[2.需求分析 3](#_Toc226442954)

[2.1系统需求概述 3](#_Toc226442955)

[2.2 功能性需求 3](#_Toc226442956)

[2.3 非功能性需求 3](#_Toc226442957)

[3.概要设计 4](#_Toc226442958)

[3.1 系统总体设计 5](#_Toc226442959)

[3.1.1逻辑结构设计 5](#_Toc226442960)

[3.1.2软件架构设计 5](#_Toc226442961)

[3.2 系统功能设计 6](#_Toc226442962)

[3.2.1总体结构 6](#_Toc226442963)

[3.2.1功能设计说明 6](#_Toc226442964)

[3.3 数据库设计 7](#_Toc226442965)

[3.3.1建库设计 7](#_Toc226442966)

[3.3.2数据库表存放关系 9](#_Toc226442967)

[3.4 界面设计 11](#_Toc226442968)

[3.4.1主界面 11](#_Toc226442969)

[3.4.2子界面A 11](#_Toc226442970)

[4.详细设计 13](#_Toc226442971)

[4.1 模块设计 13](#_Toc226442972)

[5.系统测试 15](#_Toc226442978)

[5.1 模块测试 15](#_Toc226442979)

[5.2 模块缺陷数量统计 24](#_Toc226442983)

[5.3 测试评价 24](#_Toc226442984)

[5.3.1 充分性评价 24](#_Toc226442985)

[5.3.2 缺陷与不足 24](#_Toc226442986)

[5.3.3 改进建议 24](#_Toc226442987)

[6.实习体会 24](#_Toc226442988)

[参考文献 25](#_Toc226442989)

# 1.系统概述

## 1.1 系统概括

【名称】

本系统的全称为《网上书店系统》，系统使用了当前流行的Java语言框架SSH，分别指Spring、Struts2和Hibernate框架。

【实用性】

随着internet的普及和电子商务的兴起，网上销售是internet电子商务在商品销售行业发展的必然结果，这种新型的商品销售形式，与传统销售的方式相比拥有许多优势：一是降低了销售成本；二是利用网络作为交易平台，改变传统的交易方式，使得交易活动不受空间和时间的限制；三是信息的传递更迅速灵活，交易在网上进行，大大提高了交易的效率。

【改进性】

1. 系统上：去年做过java web的课程实习，我完成的是一个网上书店。当时只是初入web，在整个系统上也就是“表单-服务器”模型，一个表单对应一个servelet的方法（doGet或者doPost），通过函数获取表单的数据，然后操作。今年学了更深入后，整个系统使用三大框架搭建，其中Spring用来管理javabean，层与层之间的对象实例化、依赖注入全写在Spring的核心配置文件applicationContext.xml中；Struts2管理所有的action，没有去年的servelet了，一个事件对应一个类的某方法，与去年的servelet有很大的不同；Hibernate是用于数据库持久层，做到了ORM的思想，一个类一张表，具体的细节都要依赖配置文件来完成。三个框架是整个系统的核心，在技术上，比去年学到的更多，也更深入了。
2. 模块上：由于本学期有软件架构的课程，有过相似的知识点的学习。本系统在模块上，从上到下按照action层、service层、dao层分层，一层一层的调用，为了减少每个层次的耦合性，使用了applicationContext.xml配置每层的对象依赖注入，实现了代码的良好复用性（移植功能不改代码，只改配置文件）。为了更好的体现ORM的思想，建立vo层，对每个实体封装一个类，也便于Hibernate框架的操作。
3. 功能上：去年的实习只是做到的简单的“增删查改”，今年的实习还增加了许多实用的功能。例如：用户注册时，增加了验证码校验的功能，当用户填写邮箱注册提交后，系统会发邮件到注册的邮箱，并让用户点击发送的链接以激活；去年只做了菜单的一级分类，在去年的基础上，增加了二级分类，在数据库中是一对多的关系；在后台管理上，也更完善了，增加了拦截器，防止用户不经过登录就访问后台；在数据库连接上，使用了c3p0数据库连接池+Hibernate实现，更有效率。还有很多新加的功能会在下面的报告中详细给出。

## 1.2 系统目标

|  |  |
| --- | --- |
| 序号 | 目标内容 |
| 1 | 前台用户的所有功能（在去年的基础上增加） |
| 2 | 后台管理员的所有的功能（在去年的基础上增加） |
| 3 | 三大框架的灵活使用，模块与模块之间的分层 |
| 4 | 使用Ajax功能，如用户注册时使用js提出请求查看用户名是否存在 |
| 5 | 前端使用JQuerymobile框架实现与手机的适配 |

|  |  |  |
| --- | --- | --- |
| 序号 | 姓名 | 承担角色和职责 |
| 1 | 张超 | 整个系统前台、后台的完成，三大框架的搭建，系统的模块设计与分层，数据库设计等。 |

## 1.2 人员计划

## 1.3 开发环境与工具

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 类别 | 具体需求 | 备注 |
| 1 | 硬件 | 联想G510 |  |
| 2 | 软件 | MySQL数据库，Java开发工具MyEclipse，谷歌浏览器，web容器tomcat6.0 |  |

# 2.需求分析

## 2.1系统需求概述

本网上书店系统采用的是典型的网上购物实践中最为普遍的模式（B2C模式），主要包括用户注册、用户登录、购物车和图书分类浏览等基本功能。此外，本系统也将实现在线图书销售系统的后台管理功能，包括图书的添加、删除和图书信息的编辑等功能。本系统基于SSH框架的搭建和MVC模式的应用，在系统的设计与开发过程中严格遵守软件工程的规范，运用软件设计模式，从而减少系统模块间的偶合，力求做到系统的稳定性、可重用性和可扩充性。

## 2.2 功能性需求

【前台部分】

1. 首页：本网上书店的首页即为index.jsp，访问首页必须经过index.action，通过执行action之中的方法，返回一个字符串“index”，再通过struts.xml定位到首页。在执行方法中，会从数据库中查出图书的一级分类、二级分类、热门商品和最新商品，保存到session或者值栈中，在页面上通过struts2的标签“s”配合迭代器的使用，把session中的东西展示出来。（此方法算是网上书店的通用方法，很多功能的实现都要依赖此流程）
2. 用户的注册：主要针对未注册的用户，完成注册功能使用，在注册过程中，需要进行数据的前台非空校验，及使用AJAX完成用户名是否存在的异步校验。在校验通过后即可完成注册，在后台同样需要进行数据校验及发送一封激活邮件。此功能在去年就实现过了，与去年不同的是：增加了额外的功能发送邮件激活和验证码，验证码图片是纯Java工具类画出来的，其值保存在session中；用户的信息的接收使用的是struts2的模型驱动接收，后面很多页面的数据传递给action都用到此方法。
3. 用户的激活：已经注册的用户，需要进入邮箱点击激活链接进行用户的激活。
4. 用户的登录：已经注册的用户，并且该用户已经激活的情况下，输入用户名和密码即可激活用户。
5. 用户的退出：针对已经登录的用户，退出该系统使用。主要是清空session中保存的用户数据然后通过action返回到首页。
6. 分类页面商品展示：根据某个分类的信息去展示商品。可以方便用户查找相应分类商品。利用封装PageBean类来实现，后面会详细介绍此方法。
7. 商品详情展示：点击某个具体商品的时候，可以显示商品的详情。以便用户可以方便的了解商品的具体信息。
8. 购物车：此功能为用户提供购物功能，用户可以根据自己需要，将自己喜欢的商品添加到购物车，也可以从购物车中移除商品，甚至清空购物车中的所有商品。
9. 订单：登录用户可以将自己喜欢商品进行购买，产生订单。以及可以根据用户信息，查询自己的订单。

【后台部分】

1. 管理员登录：管理员根据用户名和密码完成登录功能。后台使用了自定义拦截器的功能，若管理员未登录，直接通过链接访问页面，则无法访问。
2. 一级分类管理：管理员可以对前台显示的一级分类进行管理包括添加，修改，删除，查询的功能操作。
3. 二级分类管理：管理员可以对前台显示的二级分类进行管理包括添加，修改，删除，查询的功能操作。
4. 图书管理：管理员可以对前台上显示的图书进行管理包括添加，修改，删除，查询的功能操作同时要可以上传图书的图片。
5. 订单管理：管理员可以对所有的用户产生的订单进行管理包括未付款的，已经付款的，已经发货的，交易完成的订单。可以修改订单状态及异步加载订单项。
6. 用户管理：管理员可以对注册过的用户进行管理包括修改用户信息，删除用户，查询用户。

## 2.3 非功能性需求

【用户体验】

虽然老师说在界面上不作太多要求，但是作为一个网上书店系统，界面还是很重要的。在去年做基于servelet和jsp的网上书店时，我就自学了一些css布局和html标签的基础。在暑假时，也自学了一些基本的js操作实现动态网页，因此对于基本的操作，还是很熟悉的。本网上书店系统采用的B/S的模型，界面的编写主要是HTML+css+js，因为网页之类的东西，很多都是可以借鉴的，浏览器就能打开网页的源码，所以为了很好的效果，我也有参考当当网和淘宝的网页布局。另外，前台的基本校验也要做好，特别是用户注册时要及时的提示用户名是否存在，这里就用到了AJAX技术，也是在去年的基础上增加的。

【系统设计】

本网上书店用的是SSH框架，特别是Spring框架，对Java的开发有重大的帮助，Spring也连接了另外的两个框架，共同完成整个系统的搭建。三个框架的基本用途我在前面的“改进性”已经做了详细的说明。另外，本系统采用模块分层，前台主要有action层、service层、dao层，层与层之间的耦合性低，整个系统处处都是ORM的思想，本思想的核心就是一一映射，对数据库也好，对实体也好，都有Java类与其一一对应，思想的核心实现也是依赖于Java的反射机制（提供get和set方法）。软件工程的思想之一就是软件的复用性，本系统也做到了这一点，移植系统到不同的机器上时或者别的系统引用我的模块时，能做到“只动配置文件、数据库文件，而不动代码”。总之，在本次的实习中，无论是思想还是技术上，我都有很大的提高。

【系统性能】

网上开发最重要的是什么，有进程之间的并发要处理，数据库的访问效率要处理等。去年做实习开发，纯粹用的是JDBC的技术访问数据库，代码写的多且效率不好，今年改进之后，数据库的连接用到了c3p0数据库连接池，配合Hibernate框架使用，使持久层的操作效率大大提高；为了在一定程度上解决并发的问题，我在Spring的核心配置文件中加入了事务管理的配置，并在service层的类中都加了“@Transactional”注解，旨在提高实用性和系统性能。

【安全性】

增加了用户注册的安全性，利用Java中的util包里面的工具类，自己实现一个生成激活码的类，用户注册时需要填写邮箱，并在邮箱里面点击激活的链接才可以激活用户；注册时也加入了验证码，防止恶意软件刷用户注册，导致数据库存过多废数据；在登录权限上，利用session存放的键值对判断用户是否登录，结合struts2的标签，“s:if和s:else”，判断session里面的值，结合值显示不同的页面；后台结合struts2的过滤器机制，防止管理员未登录就做操作。总之，本系统的安全性就去年的来说有很大的改进。

# 3.概要设计

## 3.1 系统总体设计

## 3.1.1 SSH框架的搭建与设计

【Struts2框架】

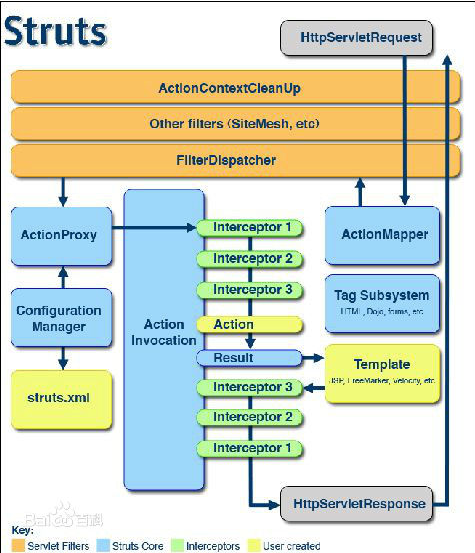
1. 介绍：

Struts2是一个具有很好实用价值的Web MVC框架，它减少了直接运用MVC模式来开发Web应用的周期。重要的是它是一个开源框架，方便开发者深入研究了解其机制，越来越广泛的接受和应用。而且它提供一个好的控制器和一套定制的标签库Taglib，着力在控制器和视图上的应用，完美的体现了MVC设计思想。提供集中统一的权限控制、国际化提示和消息、输入校验和日志记录等技术支持。

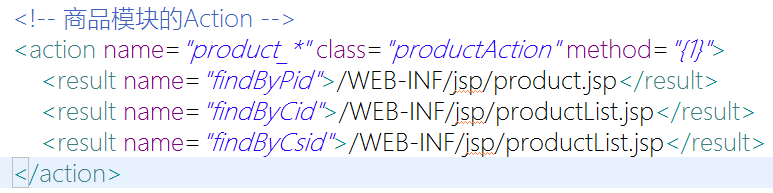
1. 核心实现：

Struts2主要是用来控制整个系统流程。它由三个部分组成，核心控制器FilterDispatcher、业务控制器和用户实现的业务逻辑组件。本系统采用的核心控制器为2.3.15.3版本，该控制器作为一个Filter运行在Web应用中，它负责拦截所有的用户请求，当用户请求到达时，该Filter会过滤用户请求。如果用户请求以action结尾，该请求将被转入Struts2框架处理。Struts2框架获得了\*.action请求后，将根据\*.action请求的前面部分决定调用哪个业务逻辑组件。

1. 结构示意图（图片来自网络）：



1. 核心配置文件（以一个action为例）：



【Hibernate框架】

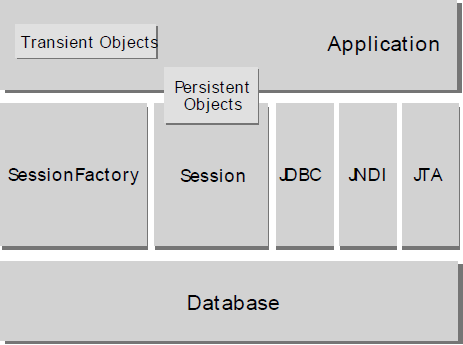
1. 介绍：

Hibernate是轻量级Java EE应用的持久层解决方案，是流行的ORM框架，它对JDBC进行了非常轻量级的对象封装，使得Java程序员可以面向对象的方式来操做数据库。Hibernate不仅管理Java类到数据库表的映射，还提供数据查询和获取数据的操作，可以减少人工使用SQL和JDBC处理数据的时间。

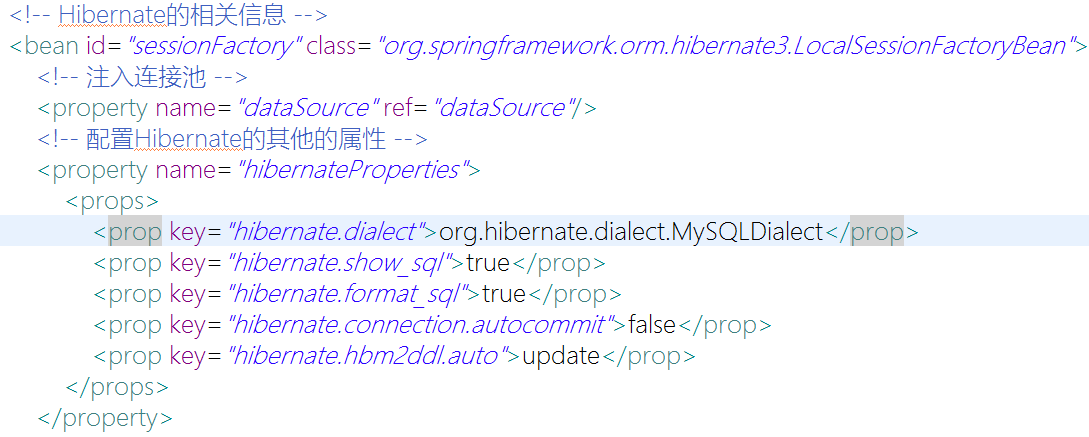
1. 核心实现：

Hibernate的核心类和接口有五个：SessionFactory接口是单个数据库映射关系经过编译后的内存镜像，产生session以操作数据库；Session接口是应用程序与持久储存层之间交互操作的一个单线程对象。所有的持久化操作都是在Session的基础上完成的；事务Transaction接口通过Transaction接口来达到允许应用系统通过一组一致的事务API来控制事务边界，从而可以封装系统底层的事务操作和访问；Query接口用来以面向对象的方式来实现对数据库的各种查询操作；Configuration类用来获取目前的配置并将这些配置加载到内存中，预启动Hibernate框架，最终达到创建SessionFactory对象的目的。

1. 结构示意图：



1. 核心配置文件（由Spring管理）：



【Spring框架】

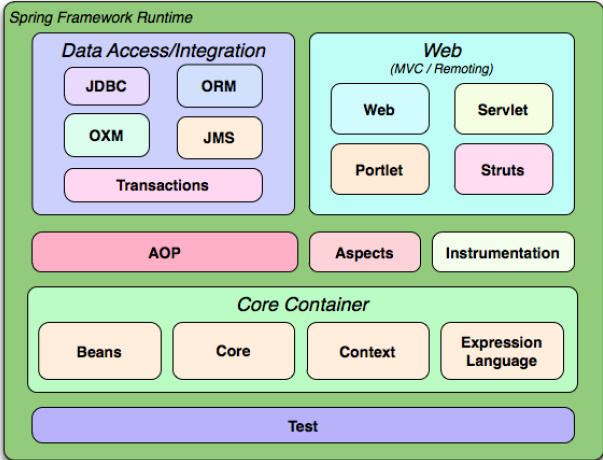
1. 介绍：

Spring是从实际开发中抽取出来的开源框架，为企业的开发提供一个轻量级的解决方案。该解决方案包括：基于Ioc(控制反转)的核心机制，以及AOP(面向切面编程)的思想，能与多种持久层技术的整合，是优秀的Web MVC框架等。Spring致力于Java EE应用各层的解决方案而不是仅仅专注于某一层的方案，它贯穿表现层、业务层、持久层，降低各层组件的耦合度，实现软件各层的解耦。

1. 核心功能：

Spring框架由7个定义良好的模块(组件)组成，各个模块可以独立存在，也可以联合使用：

1. Spring Core：核心容器提供了Spring的基本功能。核心容器的核心功能是用Ioc容器来管理类的依赖关系．Spring采用的模式是调用者不理会被调用者的实例的创建，由Spring容器负责被调用者实例的创建和维护，需要时注入给调用者。这是目前最优秀的解耦模式。
2. Spring AOP：Spring的AOP模块提供了面向切面编程的支持。SpringAOP采用的是纯Java实现。Spring AOP采用基于代理的AOP实现方案，AOP代理由Ioc容器负责生成、管理，依赖关系也一并由Ioc容器管理，尽管如此，Spring Ioc容器并不依赖于AOP，这样我们可以自由选择是否使用AOP。
3. Spring ORM：提供了与多个第三方持久层框架的良好整合。
4. Spring DAO： Spring进一步简化DAO开发步骤，能以一致的方式使用数据库访问技术，用统一的方式调用事务管理，避免具体的实现侵入业务逻辑层的代码中。
5. Spring Context：它是一个配置文件，为Spring提供上下文信息，提供了框架式的对象访问方法。Context为Spring提供了一些服务支持，如对国际化(i18n)、电子邮件、校验和调度功能。
6. Spring Web：提供了基础的针对Web开发的集成特性，例如多方文件上传，利用Servlet listeners进行IoC容器初始化和针对Web的applicationContext。
7. Spring MVC：提供了Web应用的MVC实现。Spring的MVC框架并不是仅仅提供一种传统的实现，它提供了一种清晰的分离模型，在领域模型代码和web form之间。并且，还可以借助Spring框架的其他特性。
8. 结构示意图：



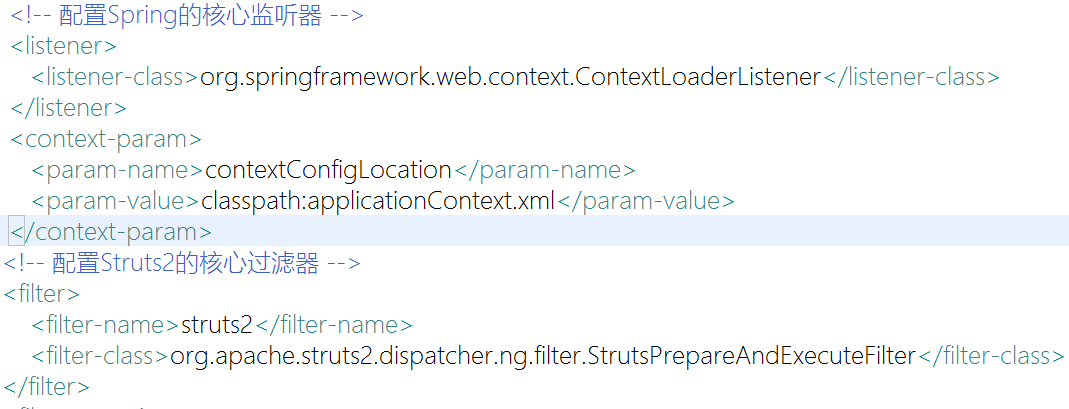
【框架的搭建】

在官网上分别下载三大框架的源码包，里面都有Demo例子，以某一个例子为例，找到文件WebRoot下面WEB-INF/lib文件夹，里面就是我们所需要的jar包。就我而言，可以找一个三大框架搭建的小项目，找到lib文件下下面的jar包，直接拿过来用即可。还有核心的web.xml文件、applicationContext.xml和struts.xml文件，按照里面的格式写我们自己的配置文件。

在进行三个框架整合开发时，可以由下往上的分层开发的原则，即：

1. 从底层开发，先开发领域对象，即一些实体类，这些领域对象会在各层之间进行传递，利用Hibernate做好领域对象和数据库表的映射关系。
2. DAO层的开发。它是专门独立出来对数据进行持久化，即处理数据的CRUD操作，也是利用Spring对Hibernate提供的DAO支持类HibernateDaoSupport来实现的。每个DAO组件对应一个数据库表。
3. Service层的开发。这层主要是具体业务组件开发，主要用于处理各类业务逻辑，并且能隔离事务和对其他资源调度。
4. Action层开发。主要是处理Web请求，由Spring进行管理。

【web.xml配置】



### 3.1.2 MVC模式分析

【模型Model】

模型是应用程序的主体部分。模型表示业务数据。通常来说，你的模型类将包含取出、插入、更新、删除你的数据库信息的功能。

该层主要是用来存实体Bean类和Hibernate影射文件。通过该层开发者很方便地对整个工程下的Bean进行管理，同时也非常方便地对Hibernate配置文件进行影射，极大的提高了开发的效率，同时对以后系统的维护也非常方便。

【视图View】

视图是应用程序中用户界面相关的部分，是用户看到并与之交互的界面。一个视图通常是一个网页，但是在 CI中，一个视图也可以是一个页面片段，如页眉、页尾。

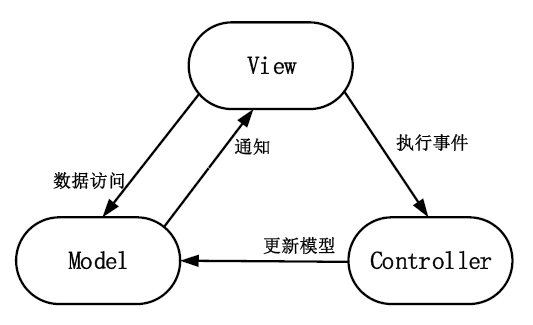
View层在本系统中主要是用来数据展示，本系统中V层主要是采用JSP技术。通过JSP技术可以很方便地把数据展示在web浏览器上。在JSP技术基础上，对其进行了改进，把html语言和java语言进行分离，这样既达到了解耦的目的，也使后续系统的维护变得很方便。把流程控制都放到Action层，把业务逻辑处理放在service层中，充分利用Struts2技术，在html页面中使用Struts2标签来展示数据，尽量减少在html页面上写java语句。

【控制器Controller】

控制器工作就是根据用户的输入，控制用户界面数据显示和更新model对象状态。是模型、视图以及其它任何处理 HTTP 请求所必须的资源之间的中介，并生成网页。

C层主要是用来做流程和逻辑控制，也是整个系统的核心层。所有进出系统的访问都要经过这一层的控制。本系统对C层时行了一些改进，我们把控制层又细分为Action层，service层和Dao层。Action层只管流程控制，Service层只管逻辑控制，Dao层只管与数据打交道，各施其职，每个层次只管自己的事情，这样能使整个系统的层次非常分明，不会出现功能的混淆。让开发者和维护者在编写代码的时候非常清晰。

【MVC原理图】

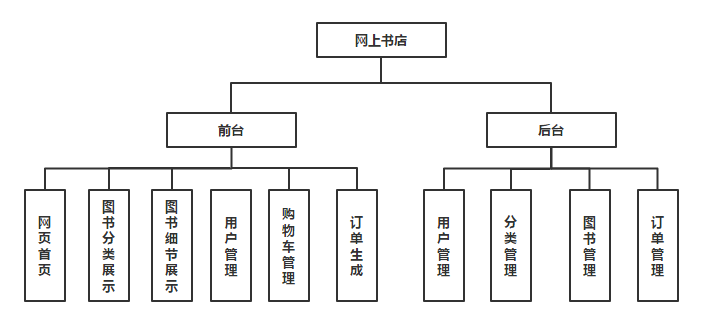


## 3.2 系统功能设计

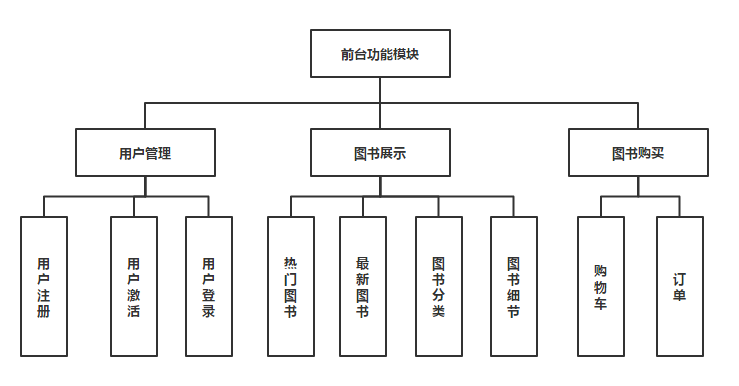
### 3.2.1总体结构

注：所有结构图均采用网上在线画图工具ProcessOn，网址链接：<http://www.processon.com>

【系统结构图】

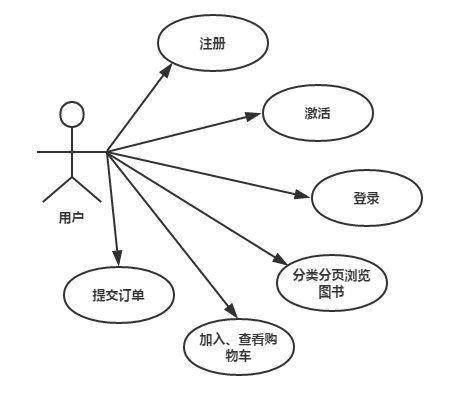


【前台功能图】

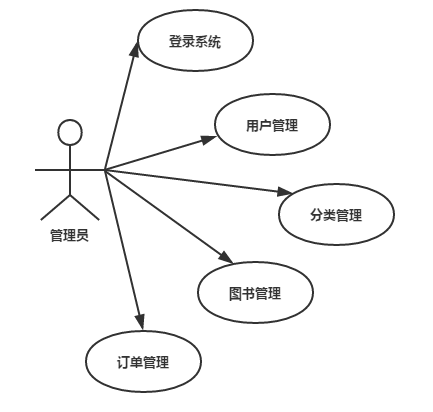


### 3.2.1功能设计说明

【用户功能用例图】



【管理员用例图】



## 3.3 数据库设计

### 3.3.1建库设计

信息系统的主要任务是通过大量数据获得管理所需要的信息，这就要求系统本身能够存储和管理大量的数据，而这一功能的实现必须借助大型数据库系统。本系统的开发选择MySQL作为后台数据库开发工具。本网上书店系统中，通过对功能的分析，设计出数据库shop，该数据库中的表如下：

【adminuser】：存放后台管理员的信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 类型 | 长度 | 说明 |
| 1 | uid | int | 11 | 主键（自增） |
| 2 | username | varchar | 255 | 管理员姓名 |
| 3 | password | varchar | 255 | 管理员密码 |

【product】：存放书籍信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 类型 | 长度 | 说明 |
| 1 | pid | int | 11 | 主键（自增） |
| 2 | pname | varchar | 255 | 书名 |
| 3 | market\_price | double |  | 图书原价格 |
| 4 | shop\_price | double |  | 图书现价格 |
| 5 | image | varchar | 255 | 图书图片地址 |
| 6 | pdesc | varchar | 255 | 图书介绍 |
| 7 | is\_hot | int | 11 | 是否热门 |
| 8 | pdate | varchar | date | 图书上市时间 |
| 9 | csid | int | 11 | 外键（图书所属二级分类） |

【category】：存放一级分类信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 类型 | 长度 | 说明 |
| 1 | cid | int | 11 | 主键（分类号） |
| 2 | cname | varchar | 255 | 分类名 |

【categorysecond】：存放二级分类信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 类型 | 长度 | 说明 |
| 1 | csid | int | 11 | 主键（二级分类号） |
| 2 | csname | varchar | 255 | 二级分类名称 |
| 3 | cid | int | 11 | 外键（所属的一级分类） |

【orders】：存放订单信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 类型 | 长度 | 说明 |
| 1 | oid | int | 11 | 主键（订单号） |
| 2 | total | double |  | 订单总钱数 |
| 3 | ordertime | date |  | 订单时间 |
| 4 | state | int | 11 | 订单状态 |
| 5 | name | varchar | 20 | 收货人姓名 |
| 6 | phone | varchar | 20 | 收货人电话 |
| 7 | addr | varchar | 50 | 收货人地址 |
| 8 | uid | int | 11 | 外键（用户id） |

【orderitem】：订单项，一个订单包含多个订单项

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 类型 | 长度 | 说明 |
| 1 | itemid | int | 11 | 主键（订单项号） |
| 2 | count | int | 11 | 选购图书数量 |
| 3 | subtotal | double |  | 订单项小计 |
| 4 | pid | int | 11 | 外键（商品id） |
| 5 | oid | int | 11 | 外键（订单id） |

【user】：用户基本信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段名称 | 类型 | 长度 | 说明 |
| 1 | uid | int | 11 | 主键（用户编号） |
| 2 | username | varchar | 255 | 用户名 |
| 3 | password | varchar | 255 | 用户密码 |
| 4 | name | varchar | 255 | 姓名 |
| 5 | email | varchar | 255 | 邮件地址 |
| 6 | phone | varchar | 255 | 电话 |
| 7 | addr | varchar | 255 | 用户地址 |
| 8 | state | int | 11 | 状态（是否激活） |
| 9 | code | varchar | 64 | 激活码（已经激活的此处为空） |

### 3.3.2数据库表存放关系

本网上书城中包含的基本概念有实体和联系。

　　1.实体 (entity)：

　　客观存在并可相互区别的事物称为实体。实体可以是具体的人、事、物，也可以是抽象的概念或联系。例如：一个学生、一门课、一个供应商、一个部门、一本书、一位读者等都是实体。

　　2.联系 (relationship)：

在现实世界中，事物内部以及事物之间是有联系的，这些联系在信息世界中反映为实体内部的联系和实体之间的联系。实体内部的联系通常是组成实体的各属性之间的联系。两个实体型之间的联系可以分为3类，一对一联系，(1:1)；一对多联系(1 : n)；多对多联系(m : n)。

【全局的E-R图】



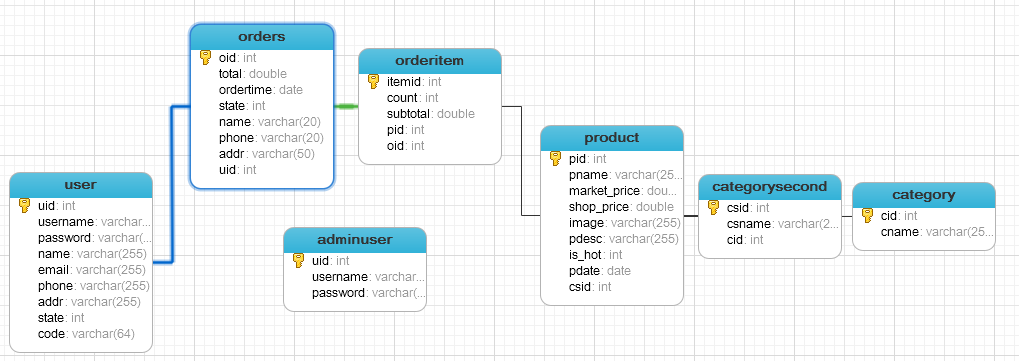
【用户实体】



【图书实体】

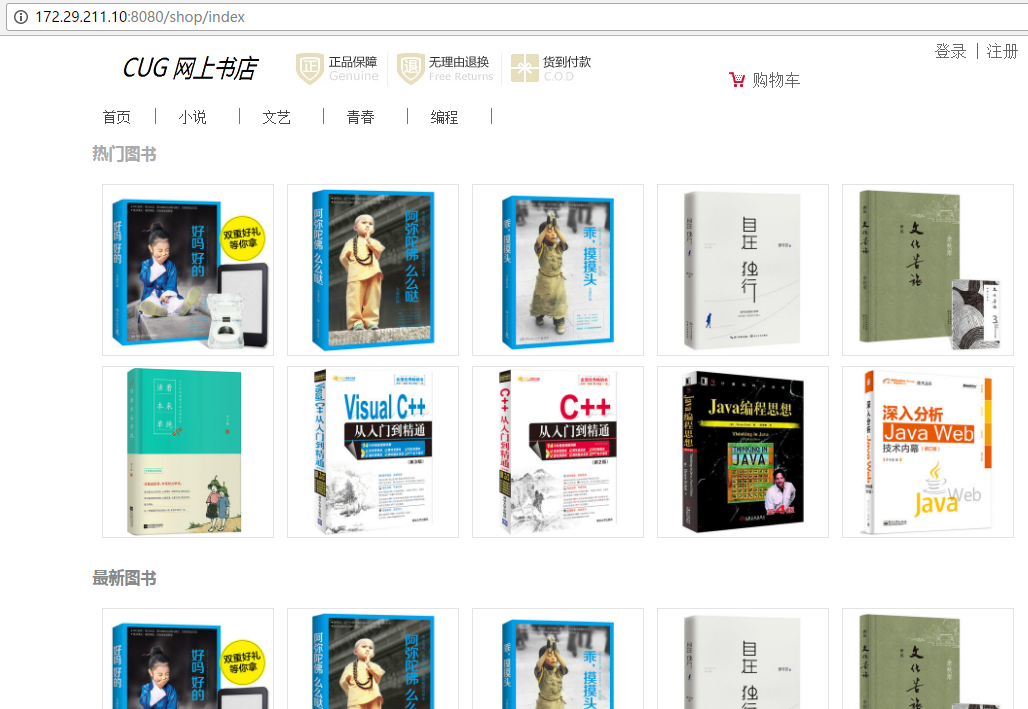


【数据库表的依赖关系图】



## 3.4 界面设计

### 3.4.1主界面

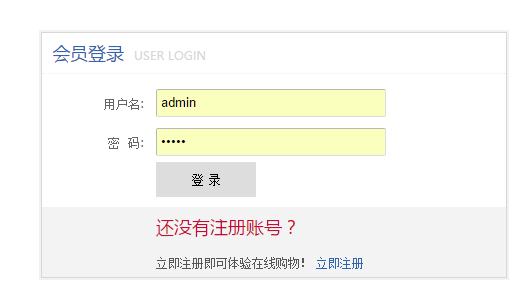


### 3.4.2子界面

【注册界面】



【登录界面】



【图书细节浏览】



【购物车页面】



【订单页面】



【管理图书主界面】



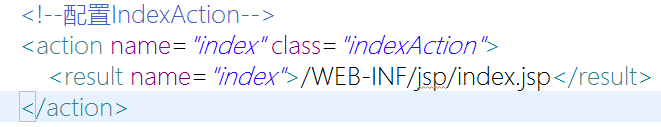
# 详细设计（核心技术概要）

【首页】

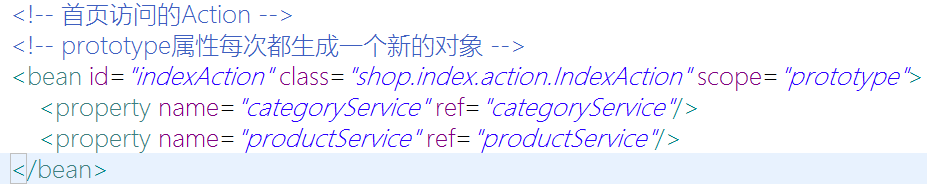
设计思路：

访问index.action后，在action的方法里面查询出所有的一级分类、最热图书（数据库中标记is\_hot为1）、最新图书（数据库中按图书上市时间查找），并保存在值栈中，在index.jsp中利用struts2的标签<s:iterator />迭代打印出来（返回值都是某某实体的List）。

struts.xml配置：



Spring配置，注入categoryService和productService：



注：所有的action都是这个思路，在struts.xml中配置好相应的action，并交给Spring的核心配置文件管理。在applicationContext.xml中，若xx.action调用了xx service的对象，则在配置文件中通过ref注入，然后有service层调用了dao层，dao层调用了sessionFactory，也是一样，按照此方法一层一层的注入。达到了良好的解耦合的目的！

【注册】

当用户填写好自己的信息后，首先在前台进行初步的校验（包括AJAX校验用户是否已经存在、填写信息不能为空等），action的配置和index.action一样，就不再累赘说明。当用户点击提交，若通过了前台校验，则使用模型驱动接收User的对象（也就是封装好了的用户实体对象）。模型驱动接收截图：



注意：必须提供set方法，就和我想要在action类里面拿到url传递过来的参数一样，必须写一个和参数类型、命名一样的变量，然后提供set方法，那么struts2框架的特性就能通过反射机制拿到变量的值！

在函数里面用service对象调用dao层的东西（已经在Spring的配置文件里面注入了），完成逻辑处理后（例如判断用户提交的验证码是否和session里面的一样），返回一个String的字符串，然后通过struts.xml找result的字符串，再通过字符串定位到某一个页面。

整个流程机制都是如此！

【订单】

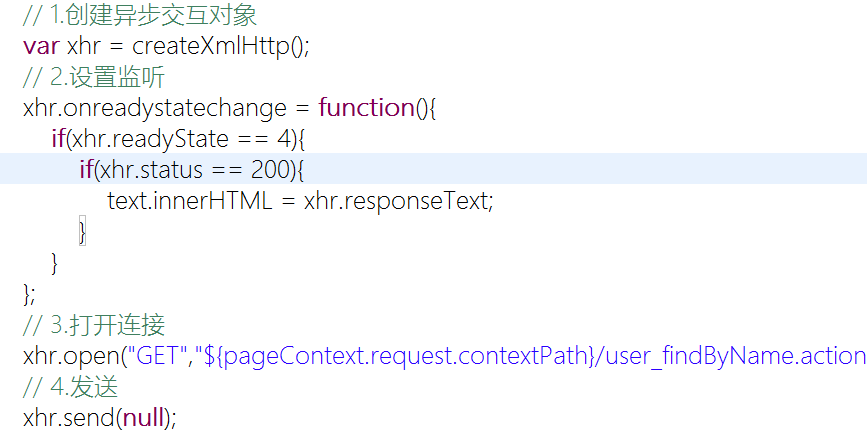
思路与首页的一样，当用户提交订单时，就把当前购物车的实体从session中拿出来，生成订单的对象并保存到数据库中。当用户查询订单时，就从数据库里面的订单查出来，保存在session中，通过s标签在jsp中显示出来。

【购物车】

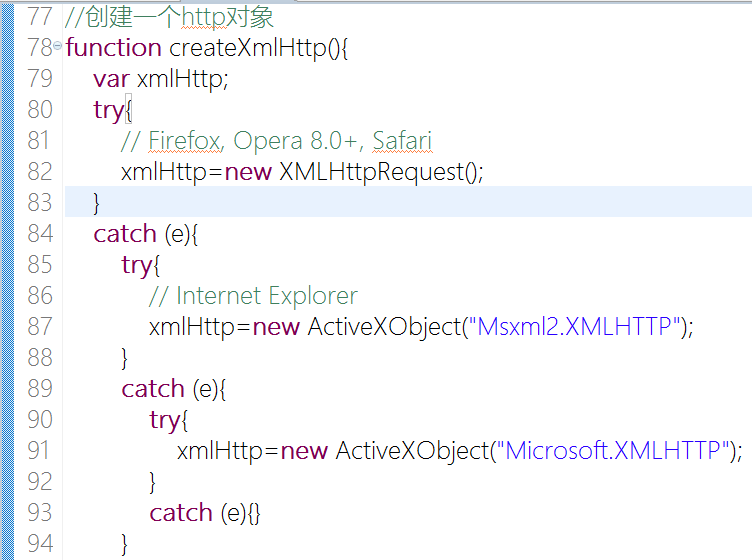
因为没有没有购物车的表，所以购物车也是在session中保存的，一个购物车对应多个购物项，添加购物车、删除购物车里面的图书，都是依赖于Map集合里面的put方法、remove方法等，真正做到了实体与类的一一对应。

【AJAX异步访问技术】

当用户注册时，当鼠标移出输入框时，触发了js的onblur的事件，在此事件中开始异步访问：一共是四个步骤：

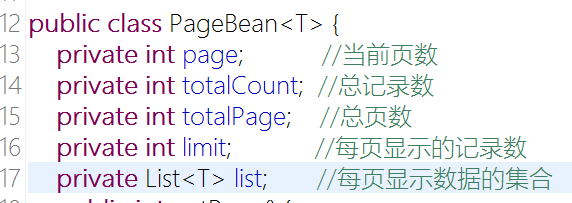


而产生异步对象时，为了考虑浏览器的兼容性，应该：

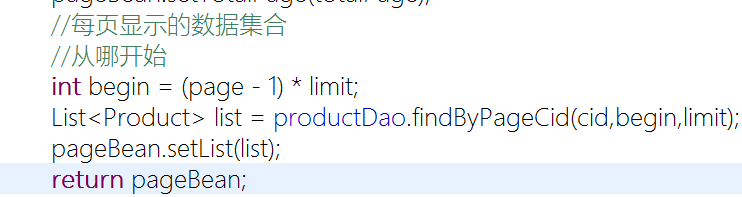


【分页技术】

前台、后台也好，有很多地方都用到了分页的技术，处理方法是：封装一个PageBean的类，采用泛型<T>，封装的属性有：

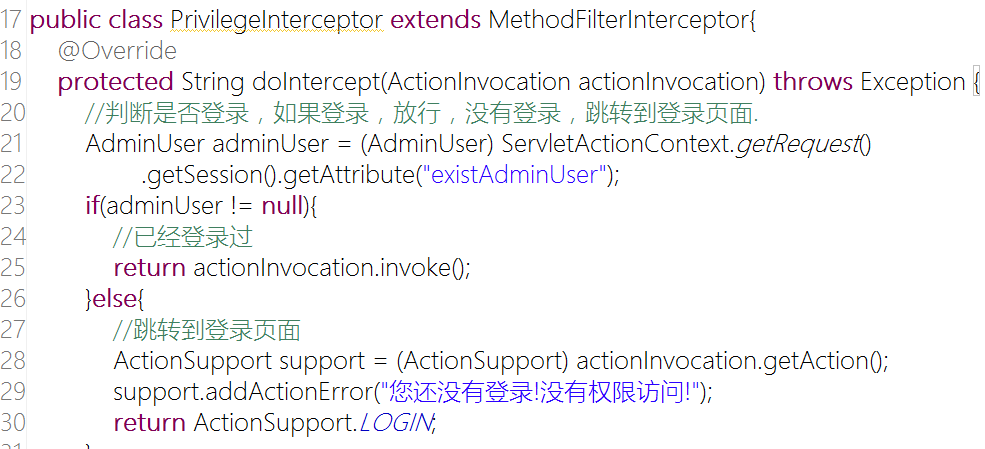


因为本身Hibernate访问数据库时是支持分页查询的，核心方法：

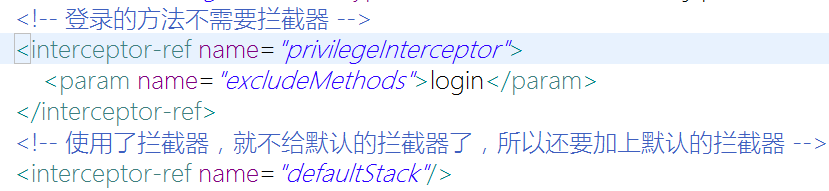


【后台过滤器】

当后台管理员未登录，直接通过url链接访问后台，然后执行查询等操作，就会被拦截器拦下。自定义拦截器：

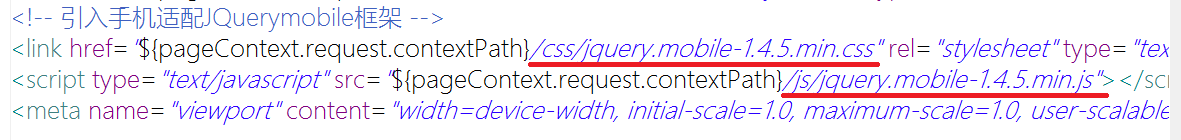


在struts.xml中使用拦截器：



【手机适配技术】

用到了开源的框架jquery.mobile，在官网上下载最新版本的源代码包，用到的只是其中的几个文件，比如我用到了其中的文件jquery.mobile-1.4.5.min.js和jquery.mobile-1.4.5.min.css，首先在html头引入这些文件：



其次，在jsp布局时，不要用绝对的布局，例如指定绝对的width为xx px，而要用百分比的宽度代替。截图：



# 5.系统测试

## 5.1 模块测试

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*前台部分

【注册用户】

1. 信息填写不完整：



1. AJAX异步访问测试：





1. 两次输入密码不同、正则表达式校验邮箱：



1. 注册成功，邮箱验证：

思路：

在本机上安装好了邮箱服务器并注册了几个邮箱备用，封装一个发送邮件的工具类，主要是用到了mail.jar包中的函数，主要是连接本地邮箱服务器的函数和发邮件的sendMail()函数。用户在注册时会生成一个激活码（java.util包中有此类），激活码会保存在User表中。激活连接会传一个参数，仅当此参数跟数据库中的相等时，才算用户激活。



邮件的接收、发送用的foxmail。

【用户登陆】（与之前模块功能相同的就不再测试）

1. 用户尚未注册或者密码错误：

核心方法：

Action中return login.jsp，并在struts2的标签中加入错误的信息，通过<s:actionerror />标签，在合适的位置把错误信息显示出来。



1. 登录成功：

思路：

登录成功后会显示“用户名”、“我的订单”和“退出”，因为登录成功后用户信息已经存在session中了，前台jsp显示时有判断，若session中有用户的值，则显示不同的菜单。



【分类浏览】

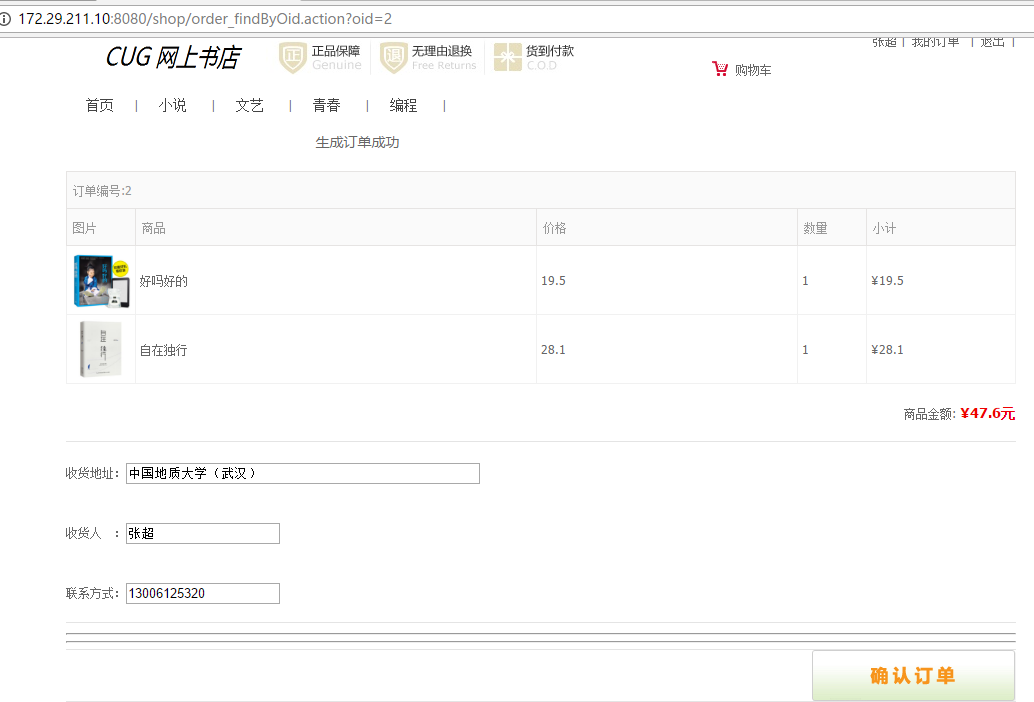
包括一级分类、二级分类，这些前台展示的分类都是与Java类实体、数据库表一一对应的，都是从数据库中读出后，再在页面中一一迭代出来的。



【购物车结算】

注：

结算的表单提交的action为空，因为后面没法再写了，后面就是把信息转到某支付平台支付了。



//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*后台部分

注：

后台的整体思路跟前台是一样的，只是对数据库的操作权限不同，前台只能查询表，而后台可以更新表、删除表。后台的分类用到了类似“树”型的结构，是用到了一个开源的dtree.js，便于更美观、直白的显示。

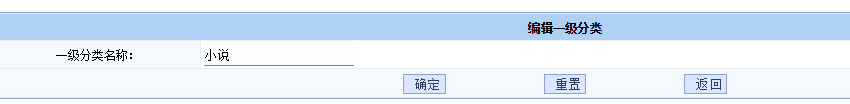
其它的后台管理，无非就是对表的“增删查改”，这里都用的很熟悉了。

【用户管理】





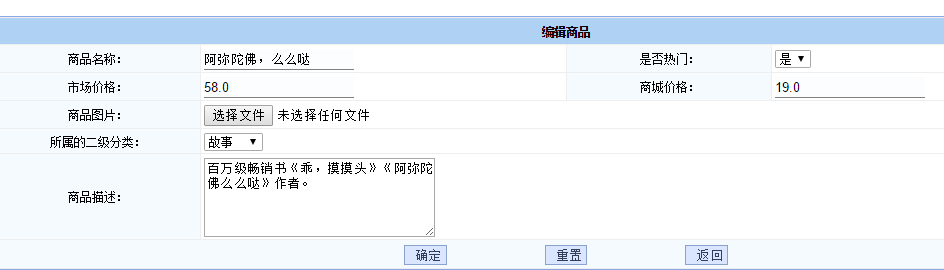
【一级分类管理】



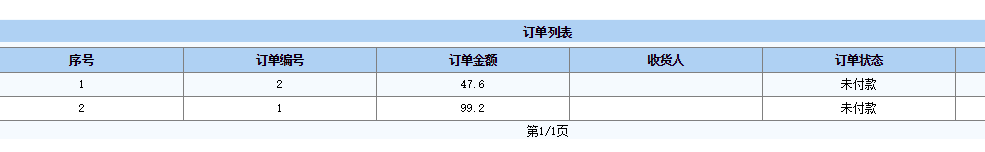
【二级分类管理】



【图书管理】



【订单管理】



## 5.2 模块缺陷数量统计

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 功能模块 | 1级缺陷 | 2级缺陷 | 3级缺陷 | 4级缺陷 | 缺陷总数 |
| 功能模块1 |  |  |  |  |  |
| 功能模块2 |  |  |  |  |  |
| 功能模块3 |  |  |  |  |  |
| 功能模块4 |  |  |  |  |  |
| 功能模块5 |  |  |  |  |  |
| ………….. |  |  |  |  |  |
| 合计 | 0 | 0 | 0 | 0 | 0 |

## 5.3 测试评价

### 5.3.1 充分性评价

1. 三大框架的分工明确，自己能够利用框架完成基本的功能。自己熟悉框架的配置，完成网上书店的基本功能，在本机上测试都没什么问题，因为没有放到服务器上，不知道很多用户同时访问时的效果怎么样。

1. 整个系统分模块来完成，每个模块之前的联系都写在Spring的配置文件中，各模块直接的耦合性低，也增加了代码的可读性、复用性。相比去年来说有很大的提高。
2. 在功能上比去年也多了很多，比如有发邮件校验，验证码校验等。还有很重要的一点就是实体--Java类—数据库的对应关系，使整个系统更加强健，不杂乱。前台页面的展示都是基于数据库的，实体类也设计的恰到好处。
3. 还有就是前端用了jquery.mobile框架，能适应手机屏幕大小，这一点也是去年没有的。

### 5.3.2 缺陷与不足

界面上有带提高，基本就是白底黑字的那种，因为本次实习主要的经历都放在后台的框架和action、数据库的处理上，其实前端的页面好多都是可以拿来用的，因为现在是主要做后端，前端的设计和知识还有待提高。

### 5.3.3 改进建议

本网上书店没有用到接口，比如dao层，更规范的写法是：写一个dao层的接口，再在接口的基础上，写一个实现类，这样实现会更降低耦合性。还有就是配置文件有很多类似key—value的键值对，不用写死在配置文件，可以抽出来，写成. properties文件，这样以后改动起来也更方便一些。

# 6.实习体会

本次实习真是一次很大的综合性的实习。无论是在模块设计、框架运用、数据库分析、实体对应上，都有很好的实习体会。本次实习最后的图书的图片测试、数据均来自当当网上书店。本次实习最大的收获就是弄懂了Spring框架的原理和基本的作用，和三大框架结合起来使用。在暑假的时候，我就对几个框架有过初步的了解，但都只是做过每个框架的小Demo，真正把几个框架都结合起来使用并运用到web项目的，就是本次大实习了。让我感受最大的就是Spring框架了，之前做过Hibernate的小demo，也对其有一定的了解，在课堂上我也讲过Hibernate的基本原理，当Spring与Hibernate结合使用的时候，Hibernate的一个核心配置就省了，都可以交给Spring来管理，当加入struts2后，也可以交给Spring来管理。所以说，Spring是企业级的管理Java Bean的框架，后来我也听过SSM框架，就是struts2和Hibernate都可以被替代，但是Spring的地位是不能动摇的。

还有我印象比较深的，也是web开发必要的知识，就是后台与前台的数据交互，我的网上书店中，大部分都是用session和值栈来传递数据的，在jsp页面上就必不可少用struts2的标签辅助我们拿数据、展示数据。这就有一个问题了，有时候数据交互的过大，tomcat有点崩了，有一次我运行就发现web项目特别的卡，后来看了报错，发现是值栈崩了，因为是存了过多的数据，后来在百度上找到了解决的办法。Struts2的反射机制也很好用，特别是用于接收参数，当我请求一个action，大多数情况下都是带参数的，现在我要在action里面把参数拿出来，只需要在类里面定义一个跟参数一样的属性，提供一下set方法，就可以拿到参数了。接收参数的模型驱动也很好用，直接拿一个实例对象，对象里面都封装好了我们需要的值，这样操作起来就更简洁了！

功能上、模块设计上、系统架构上，我都比去年Java实习有了很多的提高，在前面的报告中我也有提到。特别是模块之间的解耦性，Spring的功能就体现出来了。模块之间是一层一层的调用的，比如action层调用service层，service层调用dao层，各层之间的依赖注入都写在配置文件中。跟我们以前的实习相比，以前的实习，基本没有接触到配置文件，很多变量都写死在代码里面，有了配置文件，就好比C语言的宏定义一样，可变的东西最好都写在配置文件里面，这样可以不动代码只动配置文件。

还有本次实习对实体、ORM的思想理解的更为深刻了。比如Hibernate框架，Java类到表的映射，写好了映射文件，数据库的表都可以自动生成。Java类和实体的对应，能把常用的抽出来一个类，更方便我们逻辑计算，也是一种能力吧。还有就是ORM的实现，都是离不开Java的反射机制的，这也是Java语言一个核心的机制吧，三个框架里面每个都必须依赖于Java反射机制。

总之，本次实习是一个很好的练习过程，我相信对我的web开发能力有很大的提高！