电子书架图书管理系统设计

文档信息

系统开发信息： 开发人员：朱丛启 学号：2008082134 单位：软件 081 班	文件类型：	软件开发用技术文档
	当前版本：	V1.0 Beta
	作　　者：	朱丛启
	写作时间：	2011 年 1 月 1 日 10 时-2011 年 1 月 2 日 3 时

软件信息

软件名称：	电子书架图书管理系统
版 本 号：	V1.0 Beta
语言类型：	简体中文
开发使用技术	JSP+JavaBean+Servlet+Struts2.0+MVC
授权类型：	免费软件/测试版
运行环境：	Windows、Linux 等
软件大小：	6.24 MB (6,549,634 bytes)
指导老师：	宋海玉、王存睿
软件作者：	朱丛启
参与测试：	开发人员 等
联系信箱：	zcq.power@gmail.com
版权所有：	Copyright©2010.Powered by Power
开发日期：	2010 年 12 月 22 日-2010 年 12 月 30 日
软件简介：	电子书架图书管理系统 主要功能模块：图书分类管理，图书评论管理、书架系统后台全局管理 面向使用者：个体用户

目录

一、序言

有云：多读书，读好书。面对现在社会上琳琅满目的图书，确实让我们可以大饱眼福。从中学习到得知识自然是相当的丰富，我们可以选择自己喜欢的领域的书籍，偶尔也可以去观摩其他领域的图书，拓展自己的知识。但是一个良好的读书习惯，比起去“为了读书而读书”更应该值得我们注意。有云：行动收获习惯，习惯收获性格，性格收获成功！所以，在做人做事的时候都要有一个好的习惯。能够养成科学的读书方法和习惯是值得我们好好注意的，也是我们应该去奋斗的。

这里参考了网友们对于读书的看法：

1）一个好的读书习惯是，不只是单纯的读，在读之前可能会去从大体的总括上了解这本书的内容；

2）读书要学会多思！看完了某一章节应该去思考这一章节都讲述了些什么；

3）读书要多记！从读书的开始的总括性阅读，到对每章节的阅读和思考，有一个很好的习惯应该伴着每一个环节，那就是对读过的章节的读书笔记。

其实，自己很多的时候也是，因为也比较喜欢读书，往往也就会去读各种的书结果很多的时候把自己曾经读过的书的很多的知识都给弄混了。后来，慢慢的养成了写读书笔记的习惯，这样一来很多的问题都解决了。

综上，电子书架管理系统的开发和设计是很符合现在需求的。

二、需求分析说明书

2.1 系统介绍

系统性质 MIS（管理信息系统）。本系统采用会员制管理，系统功能应包含用户权限控制，用户图书管理，用户图书评论管理，用户图书分类管理，系统后台管理，后台管理包括系统用户管理，管理用户的信息包括设置用户密码，设置用户权限，后台直接添加用户；系统用户的图书管理，图书评论信息管理等。系统界面力求简洁、易用，在标准化的基础上考虑界面的美观和新颖。

2.2 系统面向群体

系统面向有大量阅读量的用户，包含喜欢电子版阅读的和书本阅读的用户使用。

2.3 系统功能性需求

功能编号	功能名称	功能说明
1	用户图书分类添加	注册用户操作功能，可以添加自定义的图书分类
2	用户图书分类删除	注册用户可删除已有的图书分类
3	用户图书删除	注册用户可以在指定的图书分类中添加图书
4	用户图书删除	注册用户可以在指定的图书分类中删除图书
5	用户图书评论	注册用户评论图书
6	用户图书评论查看	注册用户可以查看图书的评论
7	管理员查看系统所有用户	从后台直接查看用户的书架
8	管理员设置系统所有用户	查看当前系统的全部用户，及其权限

	信息	
9	管理后台修改用户密码	在系统后台直接设置用户的密码信息
10	管理员后台直接添加用户	在系统后台直接添加用户
11	管理员后台管理用户图书	可以直接查看用户的所有图书，并可删除
12	管理员后台管理图书评论	可以直接查看用户的所有图书评论，并可删除

三、可行性分析报告

3.1 图书阅读者现实可行性

有人说过“渔民在阅读海洋，农民在阅读大地，医生在阅读病人，气象人员在阅读天空。凡是用眼睛看到的、用心去体会的，都是在阅读。

但是，在 21 世纪的今天，很多读书的方法和途径都有了很大的变化。

年轻人喜欢上网：在这个网络发达的年代，许多人早已掉下了手中的书本，畅游在网络的海洋当中，白纸黑字的吸引力显然不够，网络八卦、明星绯闻却成了年轻人饭后津津乐道的话题。现今，很多年轻人都表示，“现在网络这么发达，还需要买什么书呢？在网上百度一下不就行了。”对于读书，现在很多的人白天的时间都在东奔西走，到了晚上，一身的疲惫，不知该如何抽出时间来阅读。

正是这样的社会现状，我们可以把网络的图书信息提供社会更个行业的人士。或许您白天忙得很累，不过在您休息或者查阅一些信息的时候我们就能给您提供便捷的服务提供方案。将我们的系统发展成为不仅是图书管理，也会有图书浏览的服务方案。

3.2 技术可行性

系统性质为基于 WEB 的信息管理系统，数据库设计和操作是整个系统的核心。因为在大二学习过理论性的数据库知识，之后有过相关的数据库系统开发经验，具备一定的系统分析与设计能力，熟悉数据库的设计与操纵，因而该系统的实现在技术上是可行的。

四、开发环境与项目规划

4.1 开发环境

4.1.1 前台开发环境

开发操作系统平台：Windows seven ultimate Edtion

编码平台：MyEclipse 8.6 Edition

界面和图片处理：Adobe Firework CS4

页面编辑和优化：Adobe Dreamweaver CS4

程序流图：Edraw Max 5.6 、 Rational Rose

服务器解析容器：Apache tomcat 6.0.14 Edition

系统调测试浏览器平台：Mozilla Firefox 3.6.13 English Edition

系统调测试操作系统：windows seven ultimate Edition 、CentOS 5.5 final Edition

4.1.2 后台数据库环境

数据库系统：MYSQL sever 5.1 Edition

数据库管理工具：Navicat 8.0.28 Enterprise

4.2 环境选择原则

4.2.1 开发操作系统平台原则

Microsoft 的 windows seven 已经发布很长时间了，通过一段时间的使用之后，对于这个系统的性能和友好的界面得出结论，windows seven 提供的性能和功能能完全的满足 WEB 类软件系统的设计和开发。而对于开源的 Linux 平台也是一个不错的选择，但是因为考虑到目前在个人用户中还是 windows 在主流的位置，所以放弃了 Linux 平台开发的计划。不过会在系统完成之后跨平台来测试开发完成的系统。

4.2.2 编码平台原则

MyEclipse 是基于开源 Eclipse 项目的一个项目，这个项目能入 Eclipse 一样完成我们的开发工作，且较 Eclipse 提供了更多的使用的插件，是目前企业公司开发选用的首选的开发 IDE，选择 MyEclipse 给我们的开发提供便捷。而 MyEclipse 的最新稳定版本是 8.6，所以选用 MyEclipse 8.6 Edition 完成系统的开发工作。

4.2.3 服务器解析容器原则

Apache 开源项目里面的一个分支 tomcat，对 JAVAWEB 的发展提供了完整的后台服务器解析支撑，也目前 JAVAWEB 开发的首选服务器解析程序，所以这里也选择 tomcat 左右服务器的解析容器，完成 JAVAWEB 后台的业务操作。

4.2.4 系统调测试浏览器平台原则

因为 Firefox 是支持 W3CHTML 标准最好的一款浏览器，而且在全球用户群众 Firefox 也占有很大的份额。因为开发系统，利用 Firefox 3.6.13 English Edition 完成系统的测试。

4.2.5 数据库系统原则

MYSQL 一个开源的数据库系统，在目前的应用很广泛，能满足很多的信息系统的数据管理应用，鉴于此选用 MYSQL 作为系统的数据存储仓库。

4.3 项目规划和管理

4.3.1 系统开发工作分配

系统开发工作有：系统前台的界面设计，使用 Adobe 的 FW 和 DW 完成协作完成。系统后台设计，使用 JAVA 语言和 JSP 等脚本完成。

4.3.2 系统开发进度安排

12 月 22 日：系统分析

12 月 23 日：系统建模，数据库系统设计

12 月 24 日：系统界面图片设计处理

12 月 25 日：系统迭代开发一期——>系统登录、注册模块设计开发

12 月 26 日：系统迭代开发二期——>系统主页模块设计开发

12 月 27 日：系统迭代开发三期——>用户书架主页模块设计开发

12 月 28 日：系统迭代开发四期——>用户书架图书管理模块设计开发

12 月 29 日：系统迭代开发五期——>系统后台全局管理模块设计开发

12 月 30 日：系统整合并测试

五、软件设计规范与标准

5.1 编写目的

制定界面设计标准规范的目的是为了规范和统一软件界面设计制定软件界面设计标准与规范。

5.2 界面设计思想

首先考虑标准化，在标准化的基础上进行界面的美工设计。在页面功能列表中使用导航条，便于用户的操作。如图：

图一 系统后台管理导航

图二 用户书架首页导航

5.3 界面设计原则

更加的切合主流的 WEB 界面，给用户提供更好的友好的操作。

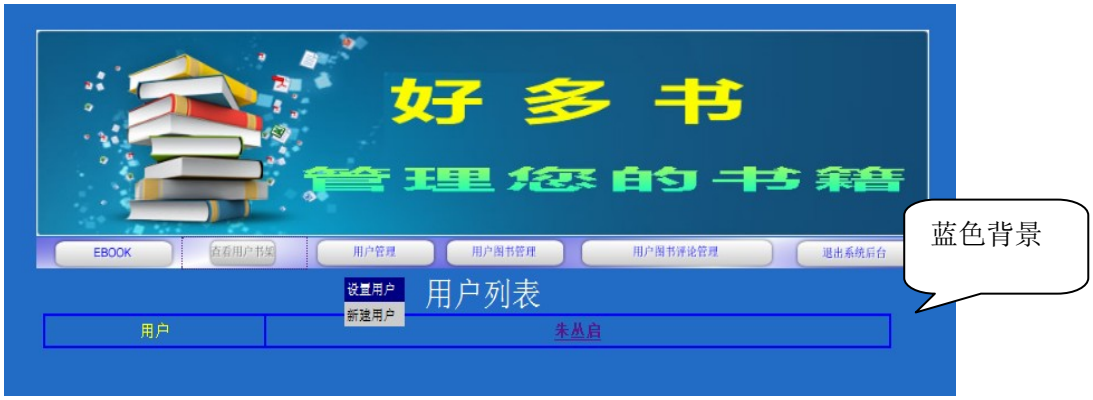
5.4 界面设计样式

主页面和书架主页均使用蓝底，蓝底是最能引起人的关注的，所以界面设计中很多

使用蓝色。而且在导航中也是结合出现的。如图：

图三 登录页面

图四 用户书架首页



图五 系统后台

5.5 常见提示信息样式

5.5.1 操作成功 JSP 页面提示：

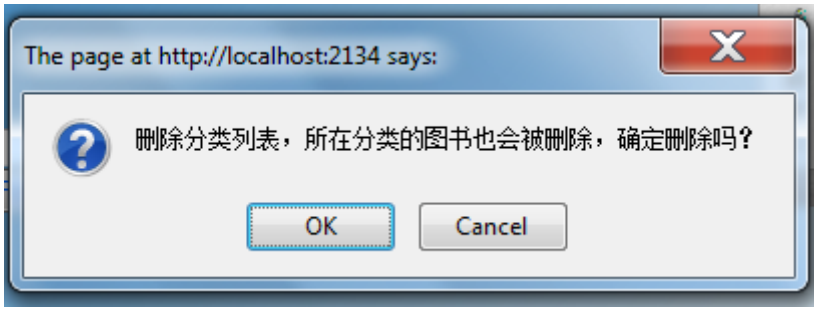
用户管理操作成功使用蓝色字大字体提示，如图：

图六 操作成功提示

后台管理操作成功使用白色大字体提示。

5.5.2 操作失败 JSP 页面提示：使用红色大字体提示

5.5.3 其他 Javascript 提示，如图：



图七 来自 Javascript 的提示信息框

六、软件编码规范与标准

6.1 控制模块对象命名

模块名称	路径
管理员添加用户	cn.ebook.admin.action.AdminAddUserAction
管理员删除图书	cn.ebook.admin.action.AdminDeleteBookAction
管理员删除评论	cn.ebook.admin.action.AdminDeleteDiscussionAction
管理员设置用户	cn.ebook.admin.action.SetUserAdminAction
管理员修改用户密码	cn.ebook.admin.action.SetUserPwAction
管理员删除用户	cn.ebook.admin.action.UserDeleteAction
用户添加图书分类信息	Cn.ebook.book.AddBookClassAction
用户添加图书	cn.ebook.book.AddBooksAction
用户添加图书评论信息	cn.ebook.book.AddEditContentAction
用户删除图书分类信息	cn.ebook.book.DeleteBookClassAction
用户删除图书	cn.ebook.book.DeleteBooksAction
获取图书分类信息	cn.ebook.book.GetBookClassAction
获取图书信息	cn.ebook.book.GetBookAction
获取图书分类信息	cn.ebook.book.GetPersonBookClassAction
获取图书信息	cn.ebook.book.GetPersonBooksAction

获取用户书架信息	cn.ebook.getdata.GetDataAction
查看用户数据信息	cn.ebook.getdata.ViewAction
用户登录控制	cn.ebook.login.LoginAction
用户注册控制	cn.ebook.signup.SignUpAction
用户图书管理操作	Cn.ebook.management.ManagementAction
管理员删除图书控制	cn.ebook.servlet.AdminDeleteBookServlet
管理员删除评论控制	Cn.ebook.servlet.AdminDeleteDiscussionServlet
管理员用户设置控制	cn.ebook.servlet.SetUserAdminServlet
管理员修改用户密码控制	cn.ebook.servlet.SetUserPwdServlet
管理员控制用户控制	cn.ebook.servlet.UserControlServlet
管理员用户删除控制	cn.ebook.servlet.UserDeleteServlet
图书模型	cn.ebook.book.BOOK
图书评论模型	cn.ebook.book.Discussion
图书分类模型	Cn.ebook.book.BookClass
图书用户模型	cn.ebook.login.User
数据库控制	Cn.ebook.db.EBOOKDAO

6.2 变量对象命名

变量名	数据类型	含义
BOOK.bookname	String	BOOK 模型图书名称
BOOK.Author	String	BOOK 模型图书作者
BOOK.Bookclass	String	BOOK 模型图书所在分类
BOOK.publishdate	String	BOOK 模型图书出版日期
BOOK.Description	String	BOOK 模型图书描述
BOOK.discussiontitle	String	BOOK 模型图书评论标题
BOOK.Discussiondate	String	BOOK 模型评论发表日期
BOOK.discussion	String	BOOK 模型评论内容
BOOK.username	String	BOOK 模型图书所属用户
BookClass.bookclass	String	BookClass 模型图书分类
BookClass.username	String	BookClass 模型图书所属用户
BookDiscussion.bookname	String	BookDiscussion 模型图书名称
BookDiscussion.discussiontitle	String	BookDiscussion 模型评论标题
BookDiscussion.discussiondate	String	BookDiscussion 模型评论时间
BookDiscussion.discussioncontent	String	BookDiscussion 模型评论内容
BookDiscussion.username	String	BookDiscussion 模型评论发布人

七、数据库分析与设计

7.1 数据库命名与规范

7.1.1 数据库和数据表的规范

名称	类型	命名约定
ebook	系统数据库	字母
表	基本表	字母
查询	查询	字母
字段	字段	字母

7.1.2 数据表详细设计

所属数据库表	名称	含义	类型	长度	命名约定
ebook.user	username	用户名	varchar	50	字母或汉字
ebook.user	password	用户密码	varchar	100	字母（MD5）
ebook.user	ADMIN	用户 admin 标记	varchar	20	字母
ebook.book	Bookname	图书名称	Varchar	100	字母
ebook.book	Author	图书作者	Varchar	100	字母
Ebook.book	Class	图书分类	Varchar	30	字母
Ebook.book	Publishdate	图书出版日期	Varchar	50	字母
Ebook.book	Description	图书描述	Text	-	字母和汉字
Ebook.book	Discussiontitle	评论标题	Varchar	50	字母和汉字
Ebook.book	Discussiondate	评论时间	Datetime	-	Datetime 数
Ebook.book	Discussion	评论内容	Longtext	-	字母和汉字
Ebook.book	Username	评论人	Varchar	50	字母和汉字
ebook.bookclass	Class	图书分类	Varchar	50	字母和汉字
Ebook.bookclass	Username	图书分类所属	Varchar	50	字母和汉字
bookdiscussion	Bookname	评论图书名	Varchar	100	字母和汉字
bookdiscussion	Discussiontitle	评论标题	Varchar	100	字母和汉字
Bookdiscussion	discussiondate	评论时间	Datetime	-	Datetime 数
Bookdiscussion	Discussioncontent	评论内容	Longtext	-	字母和汉字
Bookdiscussion	Username	评论人	Varchar	50	字母和汉字

7.2 数据库逻辑设计

7.2.1 数据表描述

USER

USERNAME PK //用户名

PASSWORD String(100) //用户密码

ADMIN String(30) //用户标识

BOOK

BOOKNAME PK //图书名称
AUTHOR String(100) //图书作者
CLASS String(30) //图书类别
PUBLISHDATE Date //图书出版时间
DESCRIPTION text(1000) //书目描述
DISCUSSIONTITLE String(50) //图书评论标题
DISCUSSIONDATE Date //图书评论时间
DISCUSSION String //图书评论内容
USERNAME FK(50) //图书的主人

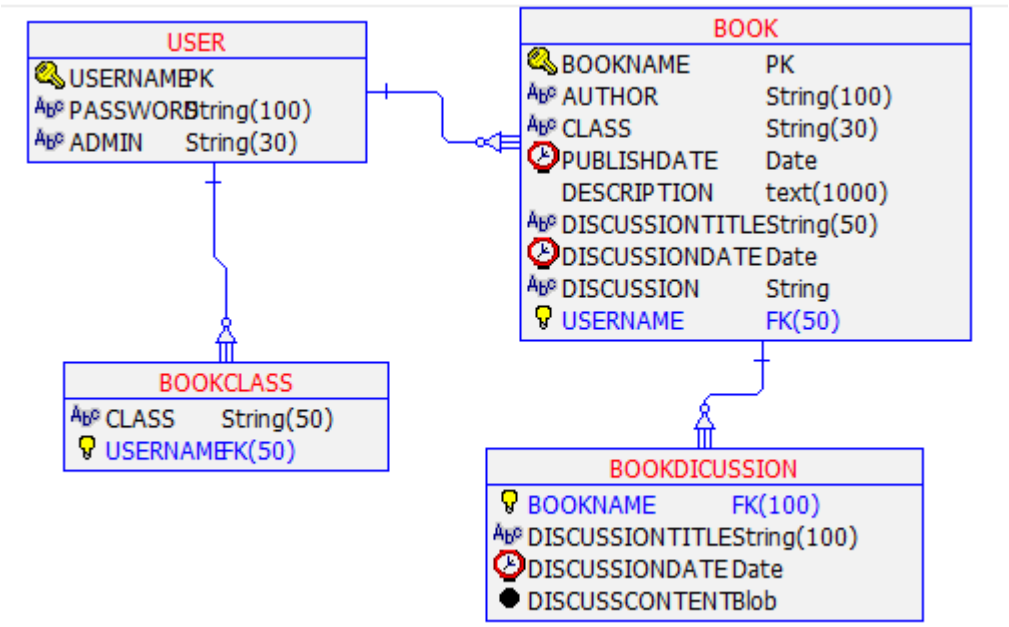
BOOKCLASS

CLASS String(50) //用户书目类别
USERNAME FK(50) //用户

BOOKDICUSSION

BOOKNAME FK(100) //图书名称
DISCUSSIONTITLE String(100) //评论标题
DISCUSSIONDATE Date //评论时间
DISCUSSCONTENT LONGTEXT //评论内容

7.2.2 数据表关系图



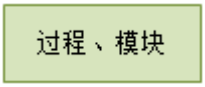
图八 数据库数据表关系图

八、软件体系结构设计说明书

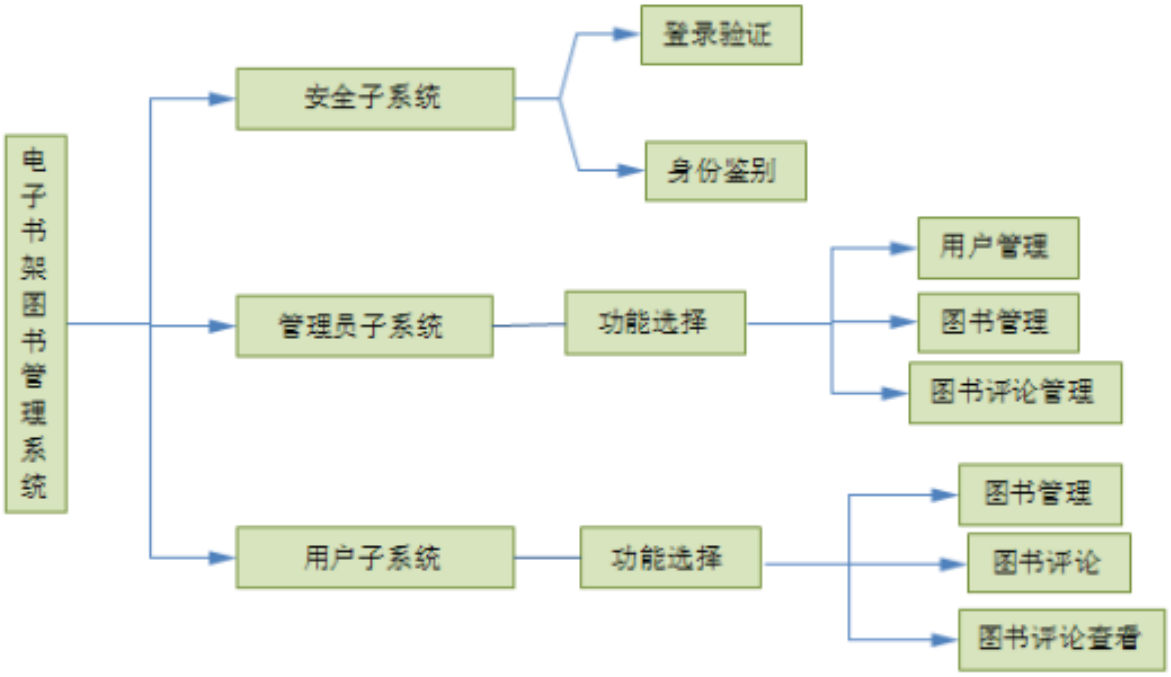
8.1 系统概要设计说明书

8.1.1 图列说明

1. 处理过程：

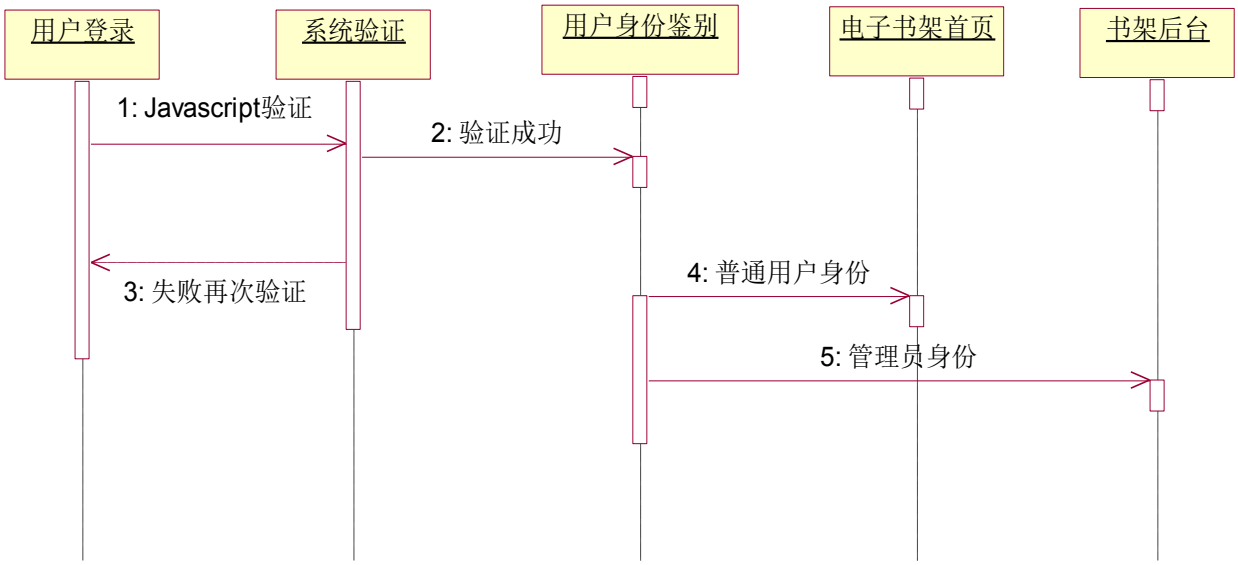


8.1.2 系统总体结构图



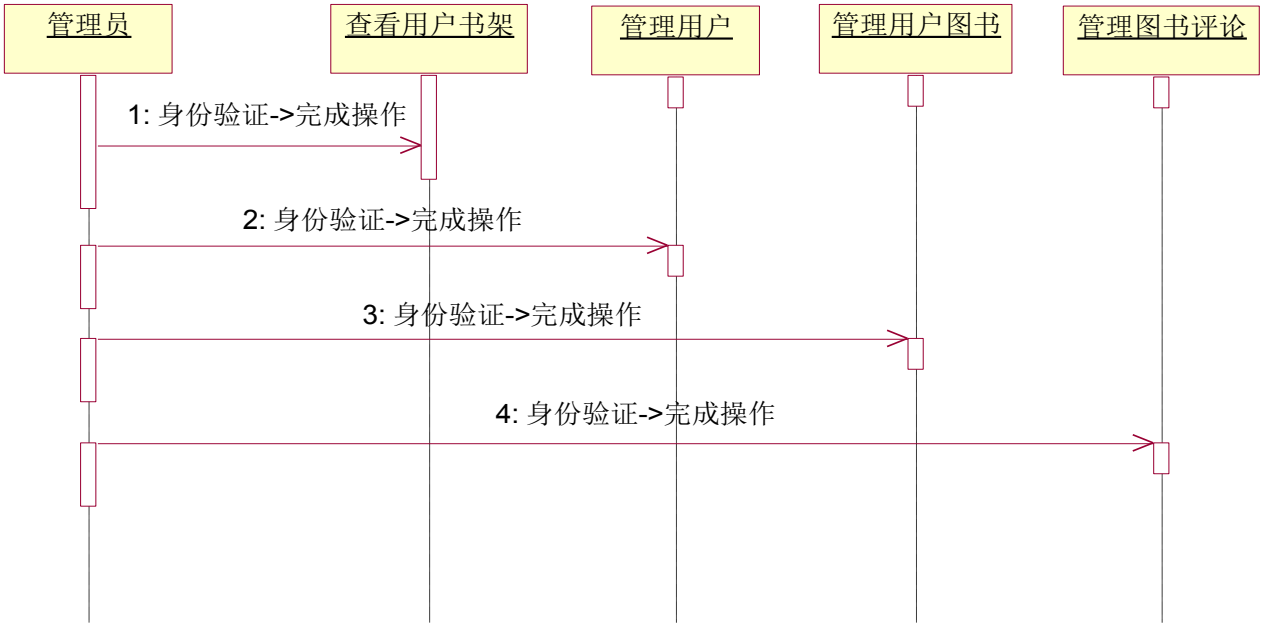
图九 系统结构图

8.1.3 安全子系统时序图



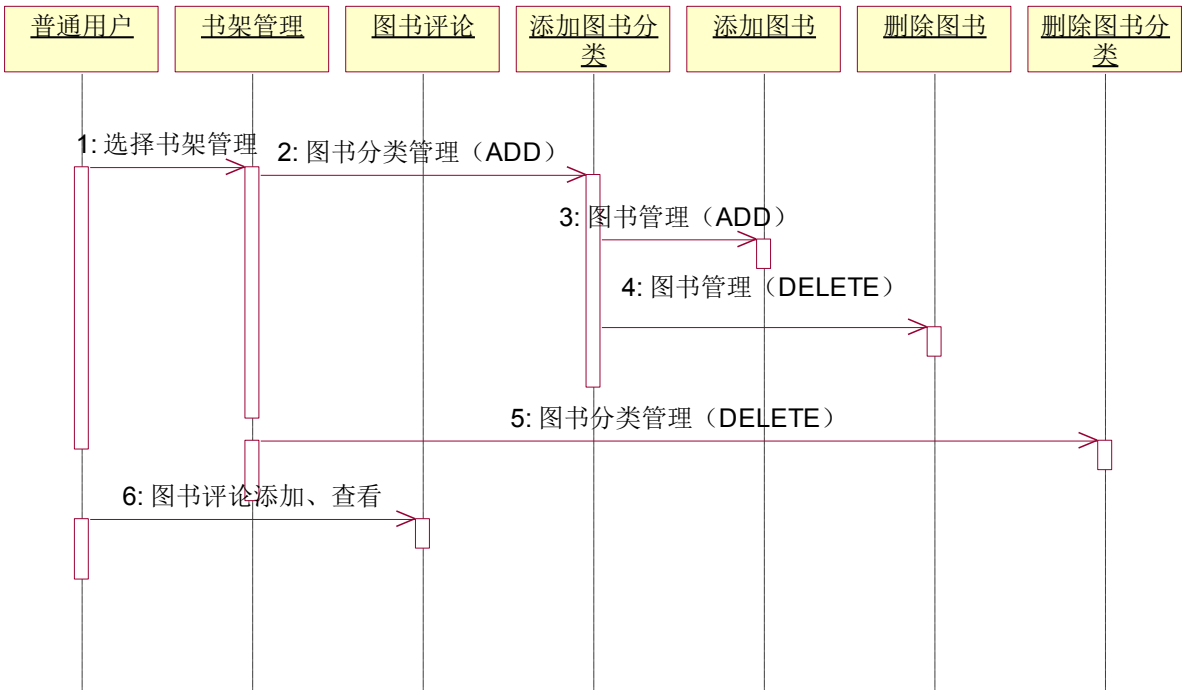
图十 安全子系统时序图

8.1.4 管理员子系统时序图



图十一 管理员子系统时序图

8.1.5 用户子系统时序图



图十二 用户子系统时序图

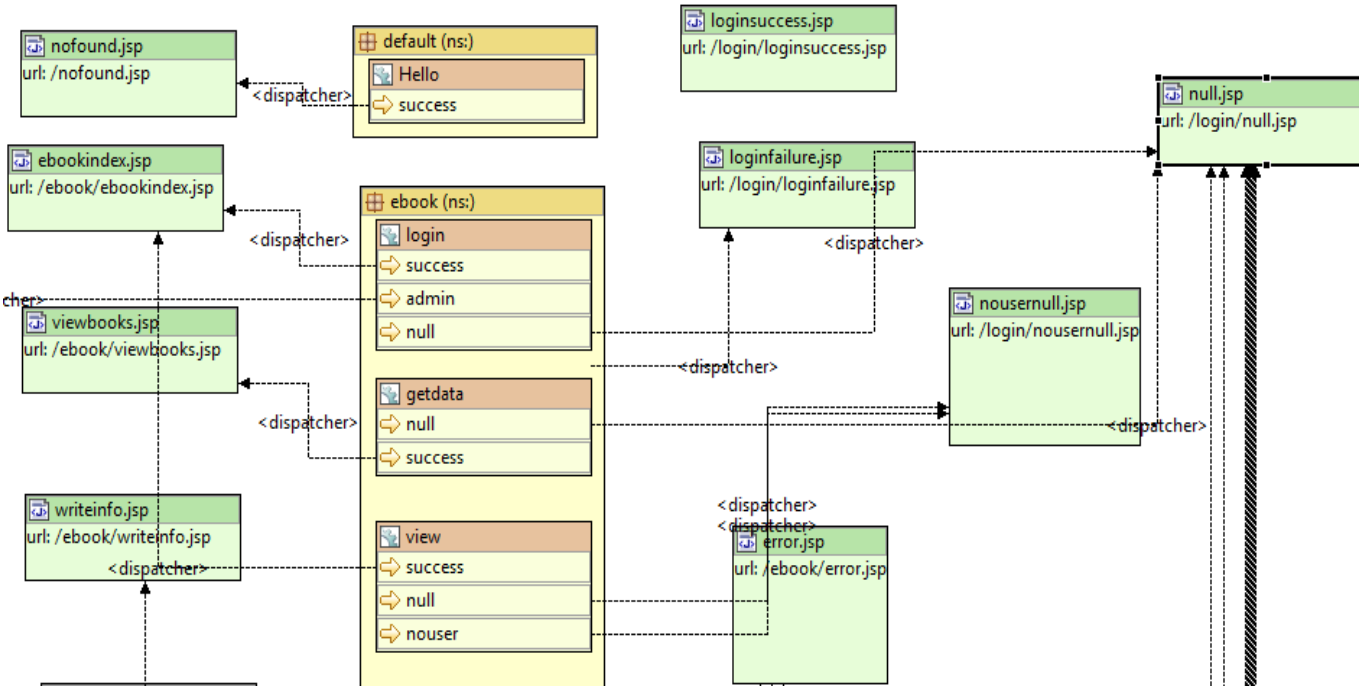
8.2 系统详细设计说明书

8.2.1 数据库操作模块汇总

模块名称	类型	原型或声明	说明
EBOOKDAO	子程序	public class BOOKDAO	数据库操作类
getDataSource	函数	public DataSource getDataSource()	获取 DataSource 对象
Login	函数	public int Login(User user)	用户登录
Signup	函数	public int Signup(User user	用户注册
addbookclass	函数	Public int addbookclass(BookClass bookclass)	添加图书类别
getbookclass	函数	public ArrayList getbookclass(String username)	获取图书类别
addbook	函数	public int addbook(BOOK book,String username)	添加用户图书
getBooks	函数	public ArrayList getBooks(String username)	获取图书
GetMoreBookinfo	函数	public ArrayList GetMoreBookinfo(String username)	获取更多的图书信息
DeleteBook	函数	public int DeleteBook(String username,String bookname)	删除图书
DeleteBookClass	函数	public int DeleteBookClass(String username,String bookclass	删除图书分类
AddDiscussion	函数	public int AddDiscussion(BookDiscussion bookds)	添加图书评论
DeleteUser	函数	public int DeleteUser(String user)	管理员删除用户
SetUserAdmin	函数	public int SetUserAdmin(String user)	管理员用户设置
SetUserPw	函数	public int SetUserPw(String user,String pw)	管理修改用户密码
DeleteDiscussion	函数	public int DeleteDiscussion(String user,String title)	管理员删除评论
closeResultSet	函数	public void closeResultSet(ResultSet rs)	关闭 ResultSet 对象
closeStatement	函数	public void closeStatement(Statement stmt)	关闭 Statement 对象
closeConnection	函数	public void closeConnection(Connection conn)	关闭 Connection 对象

8.2.2 系统核心模块设计详解

Struts2.0 模块，如图（图示威 struts 配置的部分）：



图十三 struts 配置文件 flow 视图

Struts 2.0配置信息：

```
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
    "http://struts.apache.org/dtds/struts-2.0.dtd">
<struts>
<constant name="struts.i18n.encoding" value="gb2312"></constant>
    <package name="default" extends="struts-default">
    <default-action-ref name="Hello"/>
    <action name="Hello">
    <result>/nofound.jsp</result>
    </action>
    </package>
    <package name="ebook" extends="struts-default">
    <!-- 用户登录 -->
        <action name="login" class="cn.ebook.login.LoginAction">

            <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result name="success">/ebook/ebookindex.jsp</result>
            <result name="admin">/ebookadmin/admin.jsp</result>
            <result name="null">/login/null.jsp</result>
            <result name="error">/login/loginfailure.jsp</result>
```

```
        </action>
        <!-- 主页到个人书架的切换，书架信息获取 -->
        <action name="view" class="cn.ebook.getdata.ViewAction">

            <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result name="success">/ebook/ebookindex.jsp</result>
            <result name="null">/login/nousernull.jsp</result>
            <result name="nouser">/login/nousernull.jsp</result>

        </action>
        <!-- 用户注册 -->
              <action                                    name="signup"
class="cn.ebook.signup.SignUpAction">
            <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result name="success">/login/success.jsp</result>
            <result name="null">/login/null.jsp</result>
            <result name="error">/login/failure.jsp</result>

        </action>
        <!-- 获取个人书架信息 -->
        <action                                        name="getdata"
class="cn.ebook.getdata.GetDataAction">
        <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result name="null">/login/null.jsp</result>
            <result name="success">/ebook/viewbooks.jsp</result>
        </action>
        <!-- 书架管理 -->
        <action                                    name="management"
class="cn.ebook.management.ManagementAction">
        <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result name="null">/login/null.jsp</result>
            <result name="success">/ebook/management.jsp</result>
        </action>
        <!-- 添加图书 -->
        <action                                      name="addbooks"
```

```
class="cn.ebook.book.AddBooksAction">
        <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result name="null">/ebook/error.jsp</result>
            <result name="success">/ebook/addok.jsp</result>
            <result name="error">/ebook/addno.jsp</result>
        </action>
        <!-- 添加图书分类 -->
        <action                                          name="addclass"
class="cn.ebook.book.AddBookClassAction">
        <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result name="null">/ebook/error.jsp</result>
            <result name="success">/ebook/addok.jsp</result>
            <result name="error">/ebook/addno.jsp</result>
        </action>
        <!-- 获取图书分类，在添加图书时获取分类 -->
        <action                                          name="getclass"
class="cn.ebook.book.GetBookClassAction">
        <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result
name="null">/ebook/ebookgetclasserror.jsp</result>
        <result name="success">/ebook/addbooks.jsp</result>
        <result name="error">/ebook/getclasserror.jsp</result>
        </action>
        <!-- 添加图书评论 -->
            <action                                          name="edit"
class="cn.ebook.book.AddEditContentAction">
        <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result name="null">/ebook/error.jsp</result>
            <result name="success">/ebook/addcontentok.jsp</result>
            <result name="error">/ebook/addcontentno.jsp</result>
        </action>
        <!-- 获取图书信息 ， 写书评时获取图书信息-->
        <action                                          name="getbooks"
class="cn.ebook.book.GetBooksAction">
            <exception-mapping
            result="null"
```

```
exception="java.lang.NullPointerException"/>
          <result name="null">/login/null.jsp</result>
       <result name="success">/ebook/writeinfo.jsp</result>
       <result name="error">/ebook/getbookserror.jsp</result>
       </action>
       <!-- 获取个人图书列表，删除图书时获取个人图书列表 -->
       <action                                    name="getpersonbooks"
class="cn.ebook.book.GetPersonBooksAction">
          <exception-mapping
          result="null"
exception="java.lang.NullPointerException"/>
          <result name="null">/login/null.jsp</result>
       <result name="success">/ebook/deletebookslist.jsp</result>
       <result name="error">/ebook/getbookslisterror.jsp</result>
       </action>
       <!-- 删除图书action -->
       <action                                        name="deletebook"
class="cn.ebook.book.DeleteBooksAction">
       <exception-mapping
          result="null"
exception="java.lang.NullPointerException"/>
          <result name="null">/login/null.jsp</result>
          <result name="success">/ebook/deletebookok.jsp</result>
          <result name="error">/ebook/deletebookerror.jsp</result>
       </action>
       <!-- 获取个人图书类别信息,删除图书类别时获取类别信息 -->
       <action                                    name="getpersonclass"
class="cn.ebook.book.GetPersonBookClassAction">
       <exception-mapping
          result="null"
exception="java.lang.NullPointerException"/>
          <result name="null">/login/null.jsp</result>
          <result
name="success">/ebook/deleteclasslist.jsp</result>
          <result
name="error">/ebook/getclasslisterror.jsp</result>
       </action>
       <!-- 删除图书类别 -->
       <action                                  name="deletebookclass"
class="cn.ebook.book.DeleteBookClassAction">
       <exception-mapping
          result="null"
exception="java.lang.NullPointerException"/>
          <result name="null">/login/null.jsp</result>
```

```
            <result name="success">/ebook/deletebookok.jsp</result>
            <result name="error">/ebook/deletebookerror.jsp</result>
        </action>
        <!-- 书架管理 -->
        <!-- 删除用户 -->
        <action                                          name="deleteuser"
class="cn.ebook.admin.Action.UserDeleteAction">
        <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result name="null">/login/null.jsp</result>
        <result
name="success">/ebookadmin/accountmanagement.jsp</result>
        <result name="error">/ebookadmin/deleerror.jsp</result>
        </action>
        <!-- 设置用户 -->
        <action                                            name="setuser"
class="cn.ebook.admin.Action.SetUserAdminAction">
        <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result name="null">/login/null.jsp</result>
        <result
name="success">/ebookadmin/accountmanagement.jsp</result>
        <result name="error">/ebookadmin/seterror.jsp</result>
        </action>

            <!-- 设置用户密码 -->
        <action                                          name="setuserpw"
class="cn.ebook.admin.Action.SetUserPwAction">
        <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result name="null">/login/null.jsp</result>
        <result name="success">/ebookadmin/chanagepwok.jsp</result>
        <result name="error">/ebookadmin/changeerror.jsp</result>
        </action>
        <!-- 管理员添加用户 -->
        <action                                            name="adduser"
class="cn.ebook.admin.Action.AdminAddUserAction">
        <exception-mapping
            result="null"
exception="java.lang.NullPointerException"/>
            <result name="null">/login/null.jsp</result>
```

```
        <result
  name="success">/ebookadmin/accountmanagement.jsp</result>
        <result name="error">/ebookadmin/seterror.jsp</result>
        </action>
        <!-- 删除图书 -->
        <action                                          name="bookdelete"
  class="cn.ebook.admin.Action.AdminDeleteBookAction">
        <exception-mapping
            result="null"
  exception="java.lang.NullPointerException"/>
            <result name="null">/login/null.jsp</result>
        <result
  name="success">/ebookadmin/bookmgmtbody.jsp</result>
        <result name="error">/ebookadmin/delebookerror.jsp</result>
        </action>
        <!-- 删除评论 -->
        <action                                  name="discussiondelete"
  class="cn.ebook.admin.Action.AdminDeleteDiscussionAction">
        <exception-mapping
            result="null"
  exception="java.lang.NullPointerException"/>
            <result name="null">/login/null.jsp</result>
        <result
  name="success">/ebookadmin/discussionmanagement.jsp</result>
        <result
  name="error">/ebookadmin/delediscussionerror.jsp</result>
        </action>
     </package>
  </struts>
```

数据库连接池 content 配置信息：

```
  <Context reloadable="true">
  <Resource name="jdbc/ebook" auth="Container"
     type="javax.sql.DataSource"
     driverClassName="com.mysql.jdbc.Driver"
     url="jdbc:mysql://localhost:3306/ebook"
     username="root" password="zcqadmin"
     maxActive="400" maxIdle="50"
     maxWait="-1"/>
</Context>
```

系统 web.xml 配置信息：

```
  <?xml version="1.0" encoding="UTF-8"?>
  <web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
```

```
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
 <filter>
  <filter-name>struts2</filter-name>
<filter-class>org.apache.struts2.dispatcher.FilterDispatcher</fil
ter-class>
  <init-param>
   <param-name>encoding</param-name>
   <param-value>gb2312</param-value>
  </init-param>
 </filter>
 <filter-mapping>
  <filter-name>struts2</filter-name>
  <url-pattern>/*</url-pattern>
 </filter-mapping>
  <servlet>
    <description>This    is    the    description    of    my    J2EE
component</description>
    <display-name>This    is    the    display    name    of    my    J2EE
component</display-name>
    <servlet-name>UserControlServlet</servlet-name>

<servlet-class>cn.ebook.servlet.UserControlServlet</servlet-class
>
  </servlet>
  <servlet>
    <description>This    is    the    description    of    my    J2EE
component</description>
    <display-name>This    is    the    display    name    of    my    J2EE
component</display-name>
    <servlet-name>UserDeleteServlet</servlet-name>
<servlet-class>cn.ebook.servlet.UserDeleteServlet</servlet-class>
  </servlet>
  <servlet>
    <description>This    is    the    description    of    my    J2EE
component</description>
    <display-name>This    is    the    display    name    of    my    J2EE
component</display-name>
    <servlet-name>SetUserAdminServlet</servlet-name>
<servlet-class>cn.ebook.servlet.SetUserAdminServlet</servlet-clas
s>
  </servlet>
  <servlet>
    <description>This    is    the    description    of    my    J2EE
component</description>
```

```
    <display-name>This is the display name of my J2EE
component</display-name>
    <servlet-name>SetUserPwdServlet</servlet-name>
<servlet-class>cn.ebook.servlet.SetUserPwdServlet</servlet-class>
  </servlet>
  <servlet>
    <description>This is the description of my J2EE
component</description>
    <display-name>This is the display name of my J2EE
component</display-name>
    <servlet-name>AdminDeleteBookServlet</servlet-name>
<servlet-class>cn.ebook.servlet.AdminDeleteBookServlet</servlet-c
lass>
  </servlet>
  <servlet>
    <description>This is the description of my J2EE
component</description>
    <display-name>This is the display name of my J2EE
component</display-name>
    <servlet-name>AdminDeleteDiscussionServlet</servlet-name>
<servlet-class>cn.ebook.servlet.AdminDeleteDiscussionServlet</ser
vlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>UserControlServlet</servlet-name>
    <url-pattern>/UserControlServlet</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
    <servlet-name>UserDeleteServlet</servlet-name>
    <url-pattern>/UserDeleteServlet</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
    <servlet-name>SetUserAdminServlet</servlet-name>
    <url-pattern>/SetUserAdminServlet</url-pattern>
  </servlet-mapping>
   <servlet-mapping>
    <servlet-name>SetUserPwdServlet</servlet-name>
    <url-pattern>/SetUserPwdServlet</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
    <servlet-name>AdminDeleteBookServlet</servlet-name>
    <url-pattern>/AdminDeleteBookServlet</url-pattern>
  </servlet-mapping>
  <servlet-mapping>
```

```
    <servlet-name>AdminDeleteDiscussionServlet</servlet-name>
    <url-pattern>/AdminDeleteDiscussionServlet</url-pattern>
  </servlet-mapping>
 <welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
 </welcome-file-list>
 <error-page>
  <error-code>404</error-code>
  <location>/error404.jsp</location>
 </error-page>
  <error-page>
  <error-code>500</error-code>
  <location>/error500.jsp</location>
 </error-page>
 <login-config>
  <auth-method>BASIC</auth-method>
 </login-config>
</web-app>
```

数据库存储过程：（利用Navicat创建）

```
#存储过程
#deletedicussion
#call deletedicussion(BNAME VARCHAR(100),UNAME VARCHAR(50))
BEGIN
delete from bookdiscussion where username =  username and bookname =
bname;
END
#deletebook
#call deletebook(bclass varchar(50),uname varchar(50))
begin
declare  bname varchar(100);
select bookname into bname from book where class = bclass and username
= uname;
delete from bookdiscussion where username = username and bookname =
bname;
delete from book where class = bclass and username = uname;
end
#deleteuser
#call deleteuser(user varchar(50))
begin
delete from bookdiscussion where username = user;
delete from book where username = user;
delete from bookclass where username = user;
delete from user where username = user;
```

End

数据库DataSource对象获取：

```
    //初始化数据源对象
    public EBOOKDAO(){
        InitialContext ctx;
        try {
            ctx= new InitialContext();
            datasource =
(DataSource)ctx.lookup("java:comp/env/jdbc/ebook");
        } catch (NamingException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
    //datasource数据源获取
    public DataSource getDataSource(){
        return datasource;
    }
```

数据库调用存储过程部分：

```
    //删除图书
    public int DeleteBook(String username,String bookname){
        int result = 0;
            try {
            conn = getDataSource().getConnection();
            stmt = conn.createStatement();
            String sql = "delete from book where ( username =
'"+username+"' and bookname = '"+bookname+"')";
            result = stmt.executeUpdate(sql);
            String callsql = "call
deletedicussion('"+bookname+"','"+username+"')";
            stmt.executeUpdate(callsql);
            } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        finally{
            closeResultSet(rs);
            closeStatement(stmt);
            closeConnection(conn);
        }
        return result;
    }
    //删除图书类别
    public int DeleteBookClass(String username,String bookclass){
```

```
        int result = 0;
            try {
            conn = getDataSource().getConnection();
            stmt = conn.createStatement();
            String sql = "delete from bookclass where ( username =
'"+username+"' and class = '"+bookclass+"')";
            result = stmt.executeUpdate(sql);
            String            callsql            =            "call
deletebook('"+bookclass+"',md5('"+username+"'))";
            stmt.executeUpdate(callsql);
            } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        finally{
            closeResultSet(rs);
            closeStatement(stmt);
            closeConnection(conn);
        }
        return result;
    }
//删除用户
    public int DeleteUser(String user){
        int result = 0;
            try {
            conn = getDataSource().getConnection();
            stmt = conn.createStatement();
            //String sql = "delete from user where username =
'"+user+"'";
            String sql = "call deleteuser('"+user+"')";
            result = stmt.executeUpdate(sql);
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        finally{
            closeResultSet(rs);
            closeStatement(stmt);
            closeConnection(conn);
        }
        return result;
    }
```

九、软件测试分析报告

9.1 测试范围与主要内容

测试范围	主要内容	简要说明
系统登陆验证	验证用户身份，进行权限控制	功能性测试
图书信息获取功能测试	测试数据库数据信息到 JSP 页面的完全显示	功能性测试
图书分类统计测试	测试用户自定义图书类别信息的正确性	功能性测试
数据完整性测试	测试后台整个系统的数据完整性	功能性测试
核心代码复检	检验核心代码的质量	逻辑检错性测试

9.2 测试方法

功能性测试：黑盒测试　逻辑检错性测试：白盒测试

9.3 测试报告

测试项目	测试目的	测试数据	测试结果	改进方案	修改状态
用户权限控制验证	测试后台用户设置的正确性	用户名：朱丛启 密码：zcq	系统响应正确，权限控制正常	---	---
图书分类信息获取	测试用户自定义图书分类的完整获取	用户朱丛启的图书分类信息	系统响应正确，数据完整获取	---	---
数据完整性	后台完整获取系统用户信息	当前所有的用户信息和用户图书及图书评论信息	系统响应正确，数据完整获取	---	---

9.4 开发过程中遇到的问题及解决方案

1、 Mysql 建表错误 errorno：150

错误原因：有约束关系的两个表之间的关联字段数据类型不匹配。

解决方案：修改关联表的关联字段的数据类型

2、 JSP 页面自动跳转 url 个链接 url 不一致

尚未解决。目前系统采取的措施：自动跳转和链接 url 使用不同的写法。

3、 注册页面的图片有时候无法正确显示

错误原因：FW 版本不一致，之前时候的是 FW8.0 做的，后来用 FWCS4 所有出现这个问题。

解决方案：使用 FWCS4 重新设计

4、 图书的添加中的操作 sql 语句错误

5、 数据库设计不合理，重新设计或者添加数据信息

6、 MYSQL 1303 ERROR -工具原因

错误原因：书写存储过程的时候，在 Navicat 中书写完整的存储过程出错，因为在 Navicat 里面有一部分是不需要写了，比如 create procedure procedurename(...)是不用写的。

解决方案：在 Navicat 中写存储过程的核心部分和参数列表

7、 MYSQL 1604 ERROR

错误原因：因为不会写存储过程，需要现学，就上网参考网络的列子，但是在参考的时候我粗心看错了

解决方案：按照正确的书写方法重写存储过程

8、 存储过程 DEDISCUSSION（删除图书时一并删除图书评论）删除操作不全面- 图书删除之后评论信息还存在

错误原因：存储过程书写不完善

解决方案：重新写完善的存储过程

9、 存储过程 DEBOOK（删除图书分类时一并删除图书及其评论）删除不全面 - 图书类别删除之后评论信息还存在

错误原因：存储过程书写不完善

解决方案：重新写完善的存储过程

10、 管理员识别有错

错误原因：struts Action 到 JSP session 属性名设置同名冲突

解决方案：重新设置 session 的属性名

十、软件使用说明书

详请查看附录系统配置方法

参考资料

[1] 梁立新《项目实践精解:基于 Struts -Spring-Hibernate 的 Java 应用开发》（第二版） 电子工业出版社 2008-7-1

[2] 孙鑫 《Struts 2 深入详解》 电子工业出版社 ， 2008-7-1

[3] 唐汉明 等 《深入理解 MySQL 核心技术》人民邮电出版社 2008-4-1

[4]《JSP 2.0 大学教程》

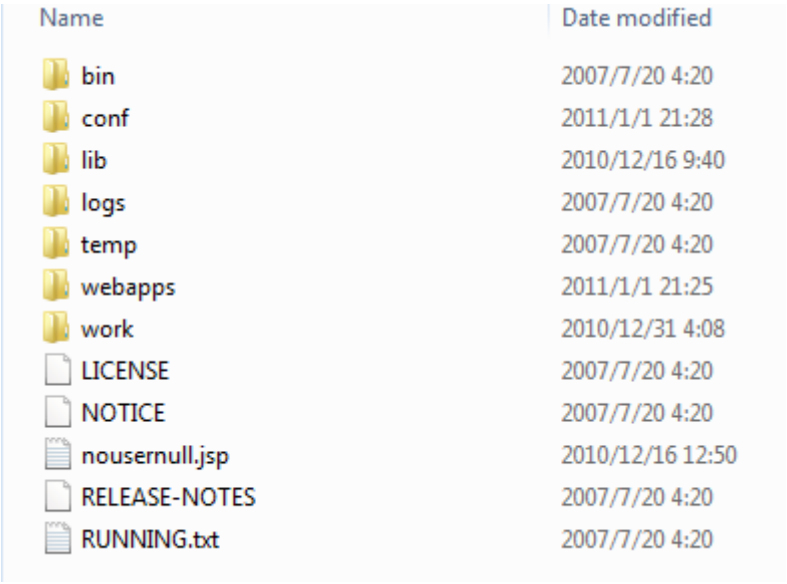
http://www.jjxw.cn/1081/2010/11/01/381@1386206.htm

附录： 系统配置方法（图示）

一、配置 tomcat 服务器：

1.1 修改配置文件

查看 tomcat 的目录文件信息



图十四 tomcat 目录树

打开 conf 文件夹，找到 server.xml 文件打开编辑 server.xml 文件，找到如下的信息：

```
<Connector port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />
    <!-- A "Connector" using the shared thread pool-->
    <!--
    <Connector executor="tomcatThreadPool"
               port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />
    -->
    <!-- Define a SSL HTTP/1.1 Connector on port 8443
         This connector uses the JSSE configuration, when using APR, the
         connector should be using the OpenSSL style configuration
         described in the APR documentation -->
    <!--
    <Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
               maxThreads="150" scheme="https" secure="true"
               clientAuth="false" sslProtocol="TLS" />
    -->

    <!-- Define an AJP 1.3 Connector on port 8009 -->
```

<Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />

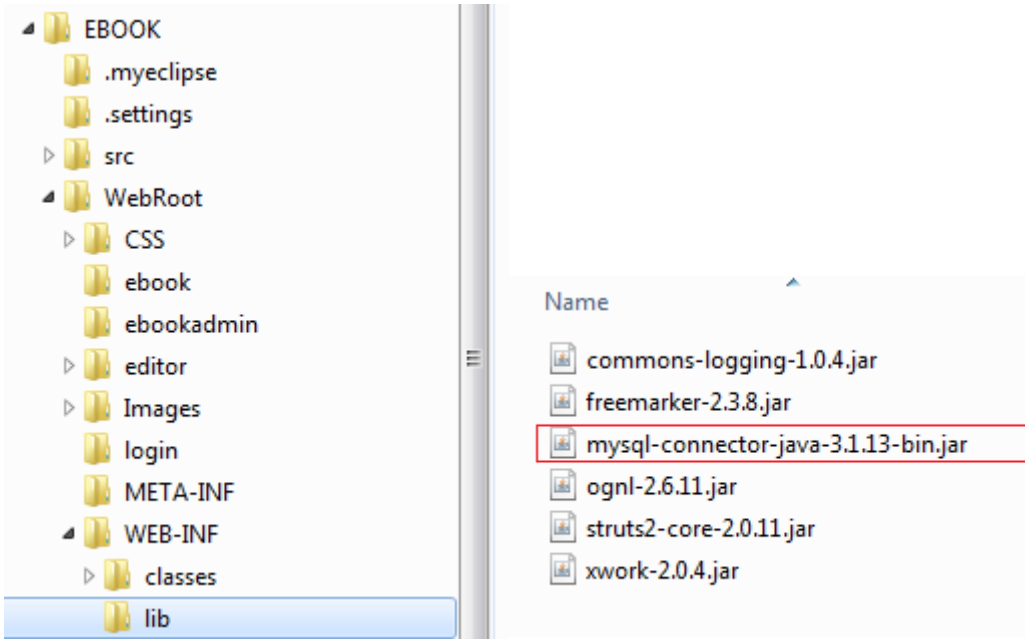
将其修改为：

```
<Connector port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" URIEncoding="gb2312"/>
    <!-- A "Connector" using the shared thread pool-->
    <!--
    <Connector executor="tomcatThreadPool"
               port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />
    -->
    <!-- Define a SSL HTTP/1.1 Connector on port 8443
         This connector uses the JSSE configuration, when using APR, the
         connector should be using the OpenSSL style configuration
         described in the APR documentation -->
    <!--
    <Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
               maxThreads="150" scheme="https" secure="true"
               clientAuth="false" sslProtocol="TLS" />
    -->

    <!-- Define an AJP 1.3 Connector on port 8009 -->
    <Connector port="8009" protocol="AJP/1.3" redirectPort="8443"
URIEncoding="gb2312"/>
```

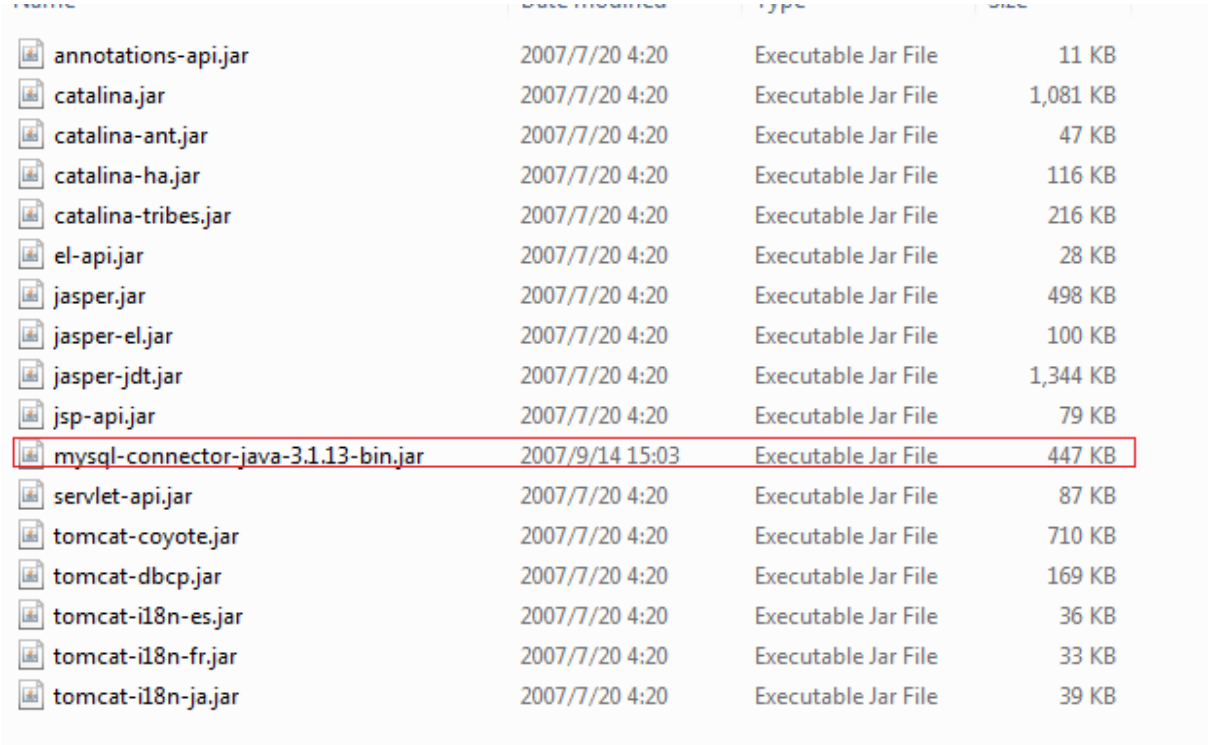
1.2 拷贝 MYSQL 数据库驱动

查看系统的目录树，如图：



图十五 系统目录树

拷贝 mysql-connector-java-3.1.13-bin.jar，放置到 tomcat 目录中的 lib 文件夹中

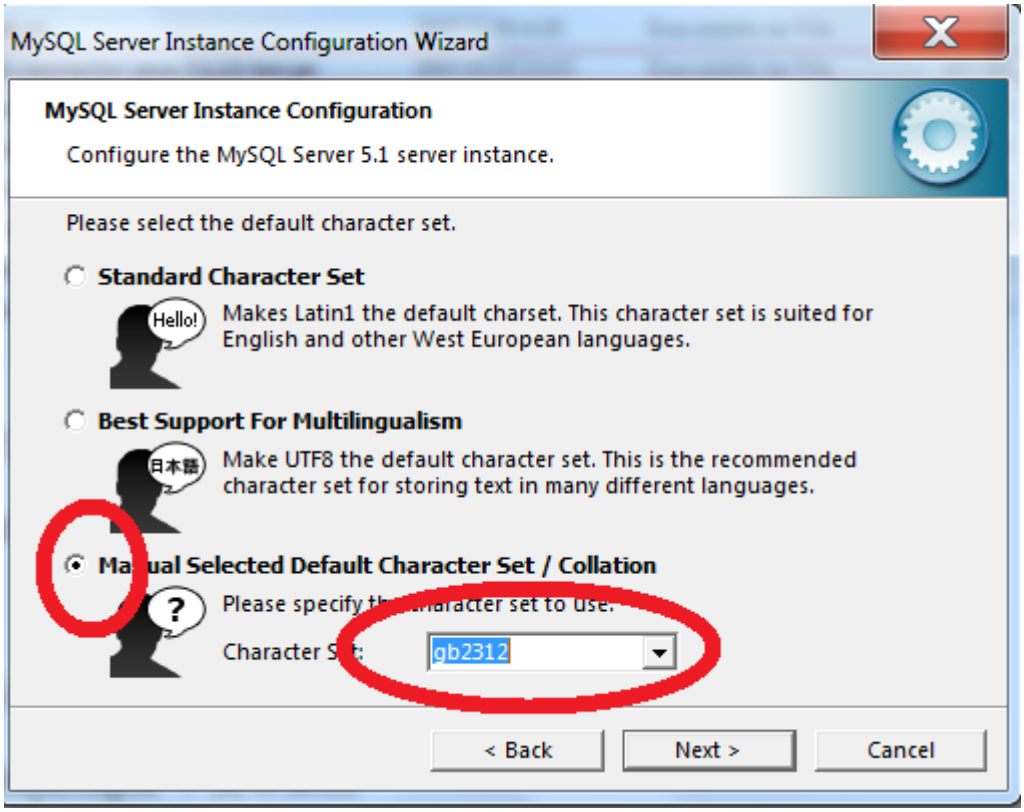


annotations-api.jar	2007/7/20 4:20	Executable Jar File	11 KB
catalina.jar	2007/7/20 4:20	Executable Jar File	1,081 KB
catalina-ant.jar	2007/7/20 4:20	Executable Jar File	47 KB
catalina-ha.jar	2007/7/20 4:20	Executable Jar File	116 KB
catalina-tribes.jar	2007/7/20 4:20	Executable Jar File	216 KB
el-api.jar	2007/7/20 4:20	Executable Jar File	28 KB
jasper.jar	2007/7/20 4:20	Executable Jar File	498 KB
jasper-el.jar	2007/7/20 4:20	Executable Jar File	100 KB
jasper-jdt.jar	2007/7/20 4:20	Executable Jar File	1,344 KB
jsp-api.jar	2007/7/20 4:20	Executable Jar File	79 KB
mysql-connector-java-3.1.13-bin.jar	2007/9/14 15:03	Executable Jar File	447 KB
servlet-api.jar	2007/7/20 4:20	Executable Jar File	87 KB
tomcat-coyote.jar	2007/7/20 4:20	Executable Jar File	710 KB
tomcat-dbcp.jar	2007/7/20 4:20	Executable Jar File	169 KB
tomcat-i18n-es.jar	2007/7/20 4:20	Executable Jar File	36 KB
tomcat-i18n-fr.jar	2007/7/20 4:20	Executable Jar File	33 KB
tomcat-i18n-ja.jar	2007/7/20 4:20	Executable Jar File	39 KB

图十六 tomcat 的 lib 文件夹中文件视图

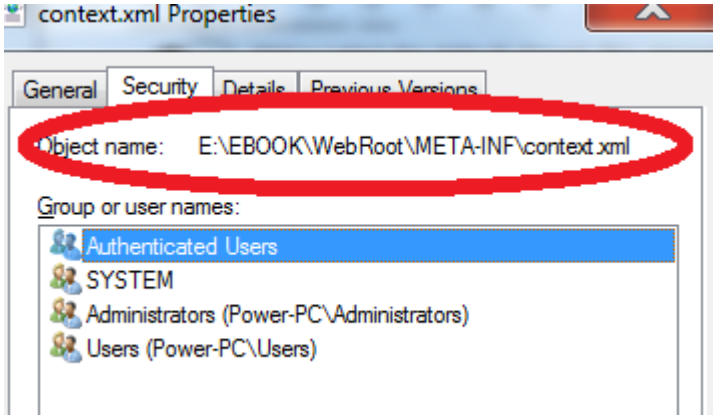
二、安装配置 MYSQL 数据库

2.1. MYSQL 安装或配置时选择正确的编码格式，如图：



2.2 MYSQL 数据密码和系统密码设置

找到配置文件如图所示（例系统存放在 E 盘下）



打开配置文件，如下：

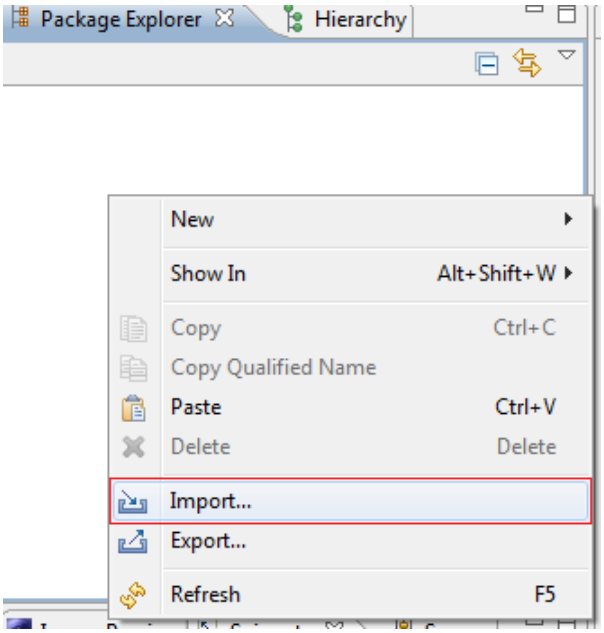
```
<Context reloadable="true">
    <Resource name="jdbc/ebook" auth="Container"
              type="javax.sql.DataSource" driverClassName="com.mysql.jdbc.Driver"
              url="jdbc:mysql://localhost:3306/ebook"
              username="root" password="zcqadmin" maxActive="200" maxIdle="30"
              maxWait="-1"/>
</Context>
```

将 password 修改为自定义的密码即可。

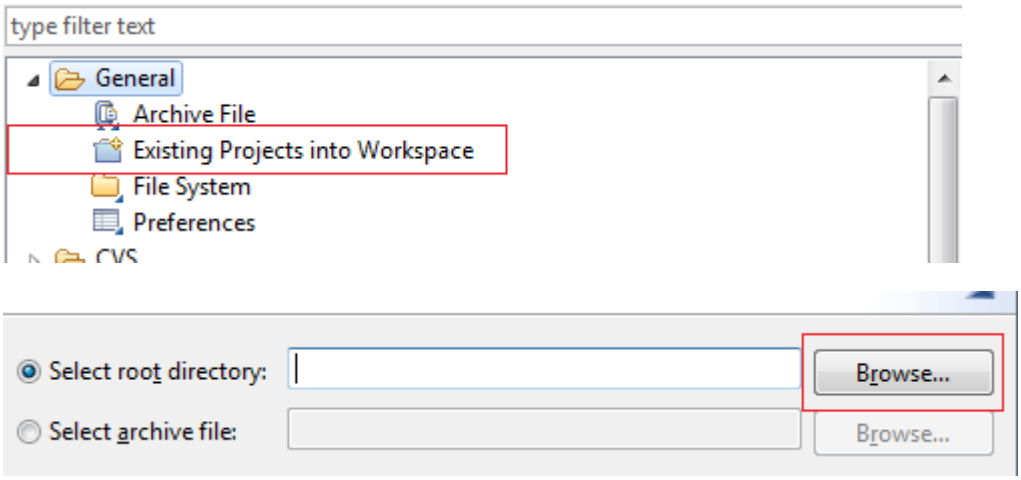
三、系统发布

3.1 在 MyEclipse 中发布系统

3.1.1 导入系统



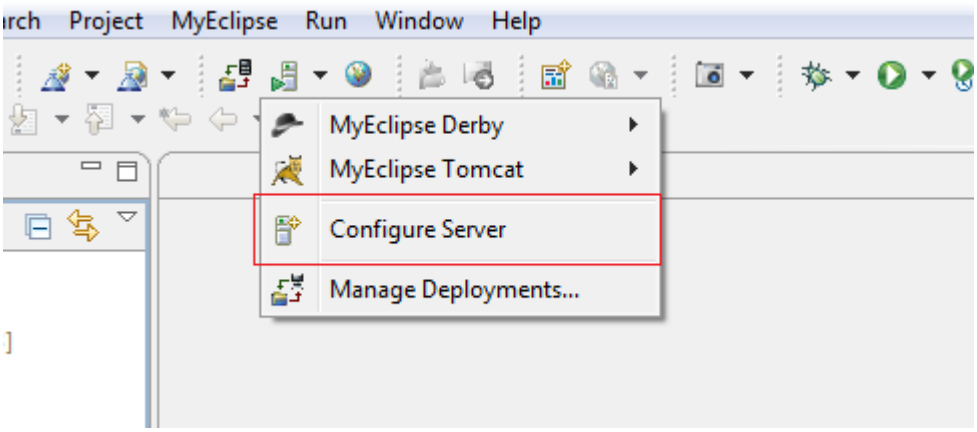
图十七 Import 系统



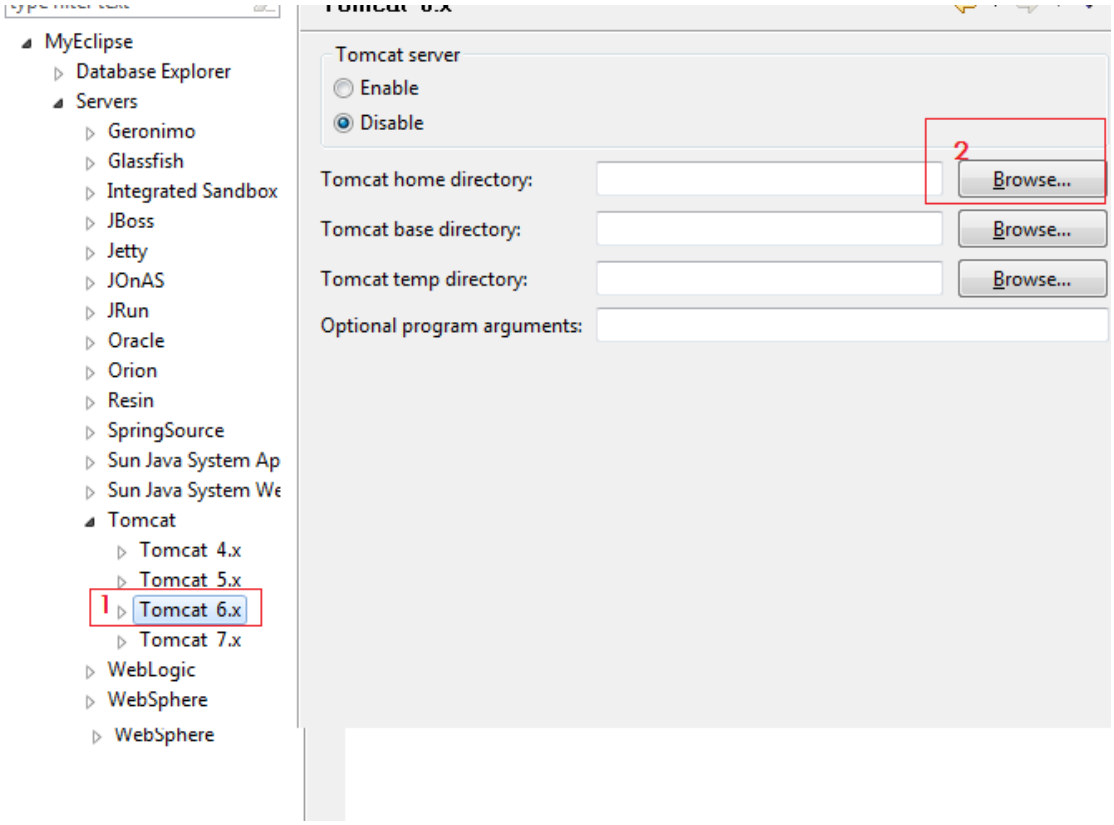
图十八 选择 Existing Projects into Workspace

点击 finish 完成系统导入。

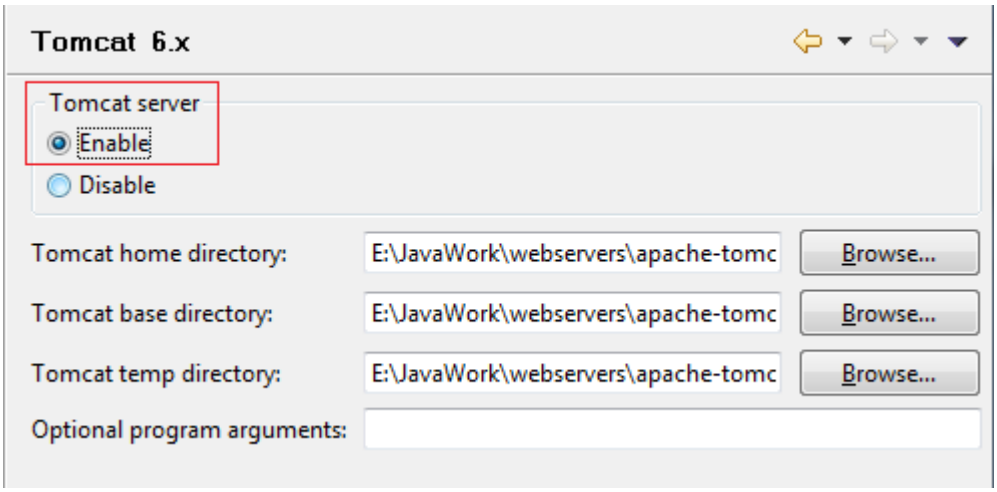
3.1.2 设置 tomcat 服务器



图十九 选择 Configure Server 设置服务器



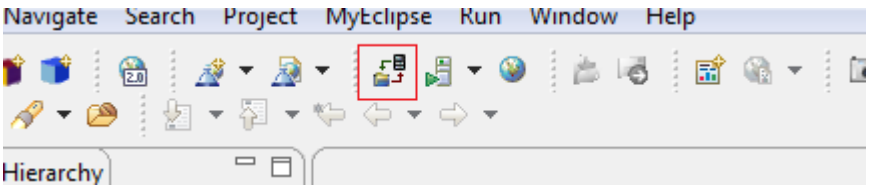
图二十 Browse 找到 tomcat 所在目录



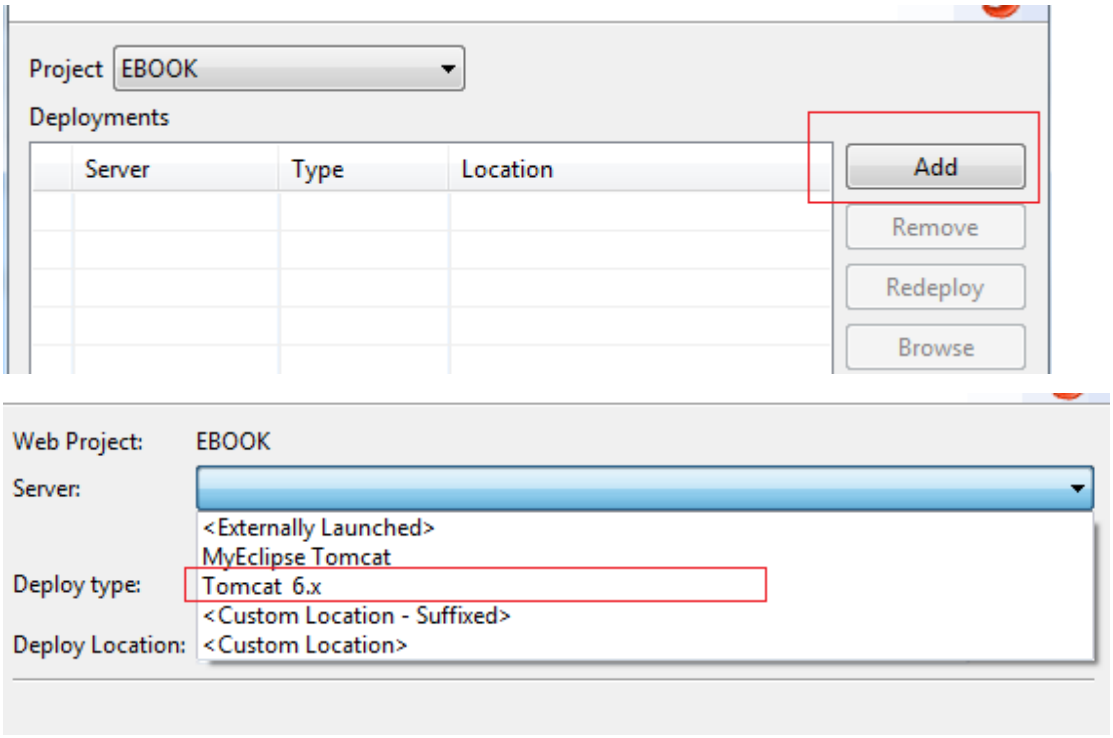
图二十一 点击 Enable 让刚才设置的 tomcat 服务器在 MyEclipse 中为等待工作状态

点击 ok 完成 tomcat 导入 MyEcilpse 设置。

3.1.3 发布系统到 tomcat 的工作目录

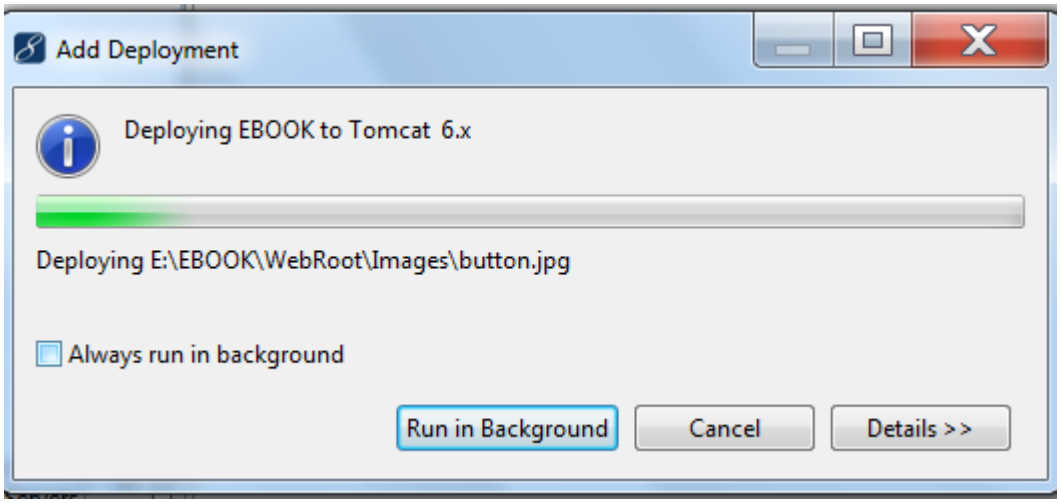


图二十二 点击图示按钮开发发布系统



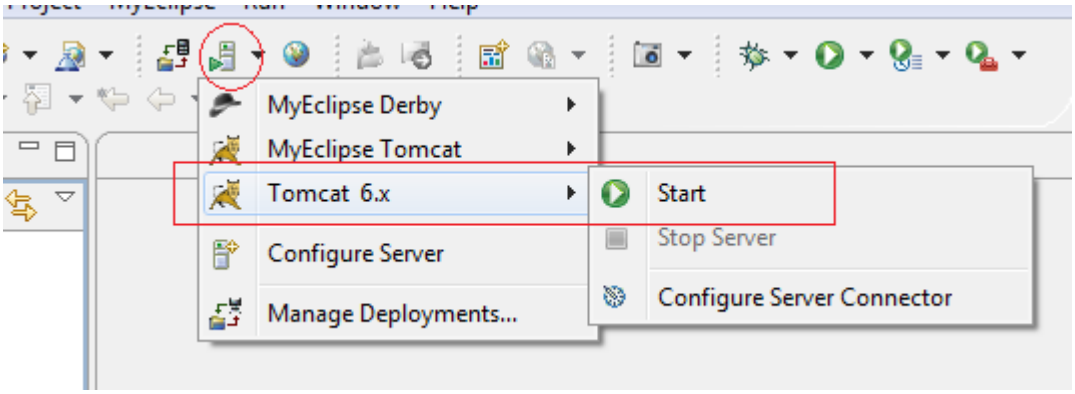
图二十三 下拉选择 Tomcat 6.x

选择好了点击 finnish 开发发布系统文件到 tomcat 的工作目录下，在点击 ok 完成发布工作，此时会看到如图的文件拷贝工作：



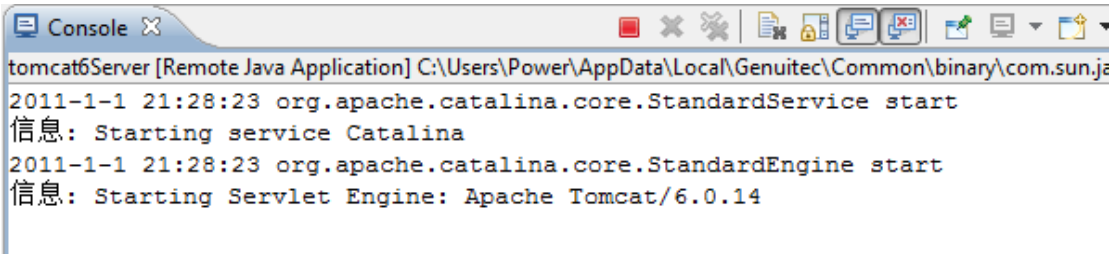
图二十四 开发拷贝系统文件

3.1.4 启动 tomcat 服务器



图二十五 启动服务器

开始启动 tomcat：



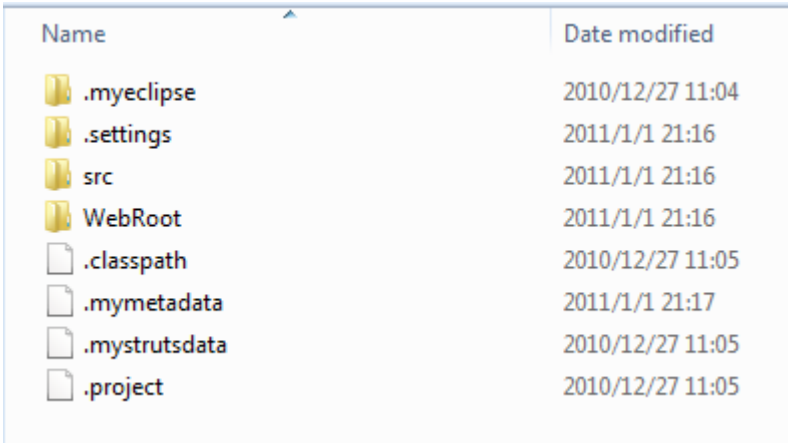
图二十六 Console 显示的服务器启动信息

打开 firefox，在地址栏中输入 http://localhost:8080/EBOOK/开始使用系统，系统工作：

图二十七 系统成功运行-系统主页

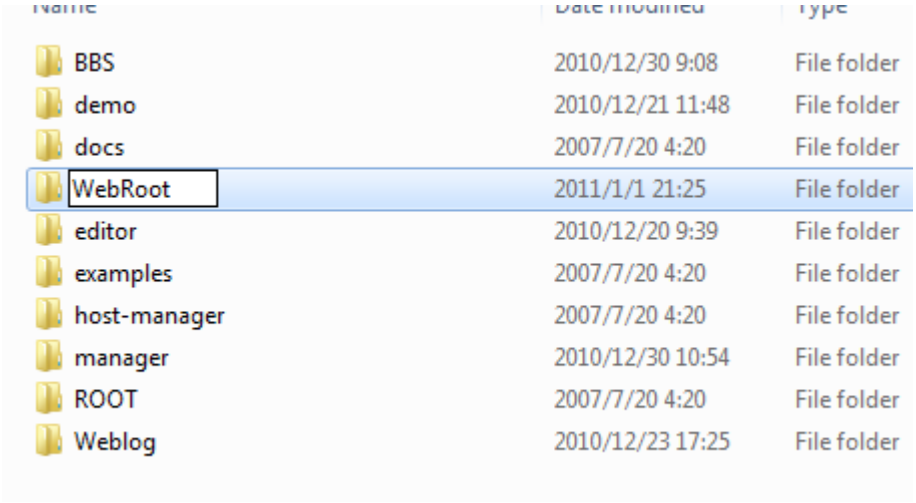
3.2 直接将系统 COPY 到 tomcat 的工作目录下

查看系统文件树，如图：



图二十八 系统文件树

拷贝 WebRoot 到 tomcat 的工作目录 webapps 中，如图是拷贝完成后的图示：



图二十九 tomcat webapps 目录下的系统文件夹

修改 WebRoot 为名 EBOOK，完成拷贝工作。启动 tomcat 服务器就可以允许系统了。