

# BookStore ( 网上书店系统 )

2017 年 7 月 10 日

## 1.简介

BookStore 是一个简单版网上书店系统，用户能搜索、浏览系统中的图书，并添加到购物车，下单购买。管理员能通过管理平台对书店的数据进行浏览和管理。

## 2.部署方案

### 2.1 部署环境

项目打包为 WAR 包后，能部署在安装了 Java、Tomcat、MySQL 和 MongoDB 的所有主流操作系统(Windows/Linux/Mac)上。数据库服务器(MySQL/MongoDB)可以和 Web 服务器(Tomcat)部署在同一台机器上，也可以部署在不同的机器上，只需修改配置文件 `src/main/resources/applicationContext.xml` 中的相应配置信息即可。

### 2.2 软件版本要求

Java: 8+

Tomcat: 7.0+

MySQL: 5.6+

MongoDB: 3.4.6+

操作系统：建议更新至当前最新版

## 3.使用的框架、类库和工具

### 3.1 前端框架和类库

使用 jQuery 3.1.1 实现 DOM 操纵，发起 Ajax 请求。

使用 Bootstrap 3.3.7 搭建前端页面。

使用开源库 `date.format.js` (<http://blog.stevenlevithan.com/archives/date-time-format>) 格式化显示时间。

使用开源库 `SimpleAjaxUploader` (<https://github.com/LPology/Simple-Ajax-Uploader>) 实现 Ajax 文件上传。

### 3.2 后端框架和类库

使用 Hibernate 框架实现数据库访问的对象-关系映射(实现了业务逻辑与数据库模式之间的解耦)。

使用 Struts 2 实现 MVC 模式(实现了业务逻辑、流程控制和页面展现之间的解耦)。

使用 Spring Framework 实现依赖注入(实现了调用者与被调用者具体实现类的解耦，实现了代码与运行环境间的解耦)。

### 3.3 开发工具

使用 Eclipse 作为开发的 IDE。

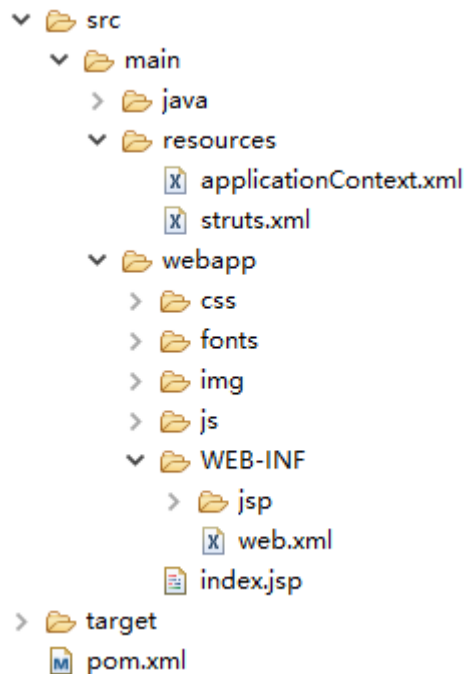
使用 Maven 管理项目使用的第三方依赖库。

使用 Git 进行软件版本控制管理。

## 4.项目结构

### 4.1 整体结构

项目的整体目录结构如下：



其中：

后端 Java 代码位于 src/main/java 目录中。

IoC 容器的配置信息写入 src/main/resources/applicationContext.xml 文件中。














前端静态资源（JavaScript 脚本、CSS 文件、字体和预置图片等）位于 src/main/webapp 目录中。

JSP 视图文件位于 src/main/webapp/WEB-INF/jsp 目录中。















此外，提交压缩包最外层目录附带的 bookstore.sql 文件包含了 MySQL 数据库的结构、样例数据及编写的存储过程，bookstore.json 文件包含了从 MongoDB 中导出的样例数据（MongoDB 数据库名为 bookstore，集合名为 userAddresses）。

### 4.2 包设计









项目共包含 8 个包，它们的内容和功能如下：

- ▼  bookstore.action
  - >  AdminBookAction.java
  - >  AdminCategoryAction.java
  - >  AdminOrderAction.java
  - >  AdminStatAction.java
  - >  AdminUserAction.java
  - >  AuthAction.java
  - >  BaseAction.java
  - >  CartAction.java
  - >  HomeAction.java
  - >  OrderAction.java
  - >  ProfileAction.java
  - >  UploadImageAction.java









包含所有控制器类，每个用例对应一个控制器。在 BaseAction 中定义了一些辅助方法，其他 Action 均继承 BaseAction。

- ▼  bookstore.model
  - >  Address.java
  - >  Book.java
  - >  BookCategory.java
  - >  Category.java
  - >  Order.java
  - >  OrderItem.java
  - >  User.java
  - >  Book.hbm.xml
  - >  BookCategory.hbm.xml
  - >  Category.hbm.xml
  - >  Order.hbm.xml
  - >  OrderItem.hbm.xml
  - >  User.hbm.xml





用 Java Bean 和 \*.hbm.xml 定义项目涉及的数据模型，用于与数据库建立对象-关系映射。

- ▼  bookstore.dao
  - >  BookCategoryDao.java
  - >  BookDao.java
  - >  CategoryDao.java
  - >  OrderDao.java
  - >  OrderItemDao.java
  - >  StatDao.java
  - >  UserDao.java











将对数据模型的 CRUD 操作封装成数据访问对象（DAO），定义 DAO 接口。

- ▼  bookstore.dao.impl
  - >  BookCategoryDaoImpl.java
  - >  BookDaoImpl.java
  - >  CategoryDaoImpl.java
  - >  OrderDaoImpl.java
  - >  OrderItemDaoImpl.java
  - >  StatDaoImpl.java
  - >  UserDaoImpl.java

DAO 的实现类，使用 getHibernateTemplate() 方法和 MongoTemplate 对象操作数据模型。

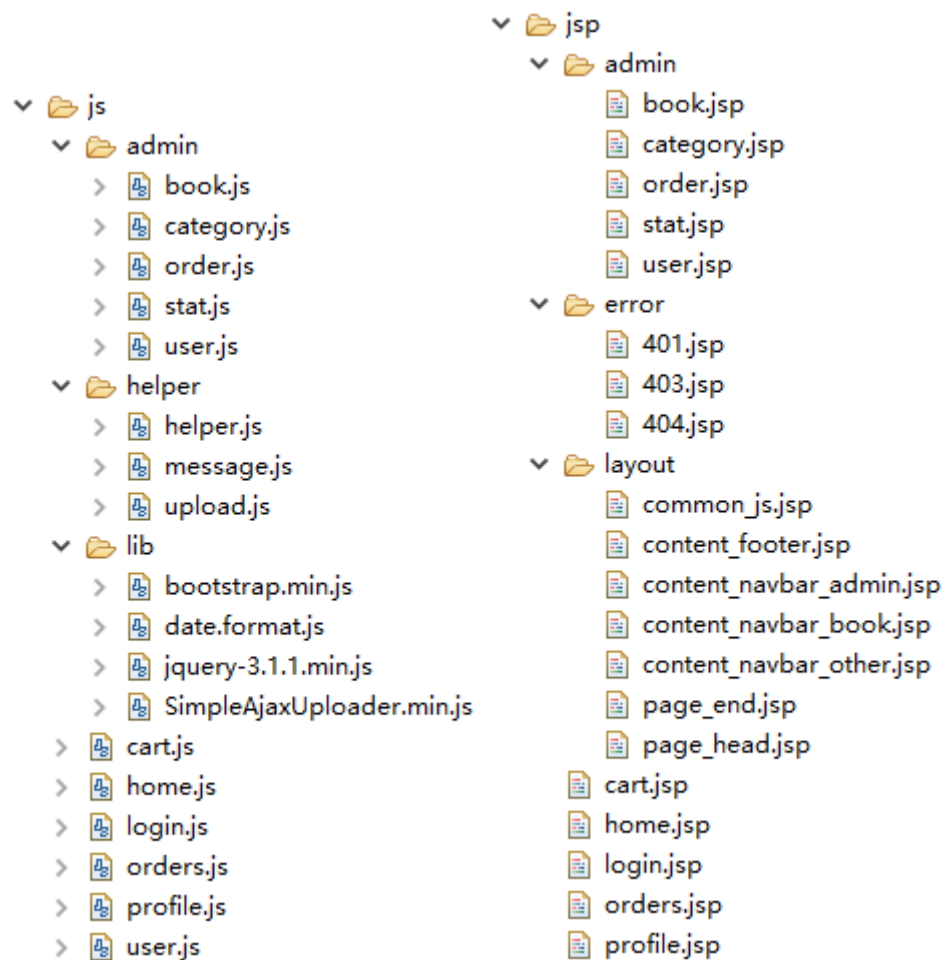
- ▼  bookstore.service
  - >  AppService.java
- ▼  bookstore.service.impl
  - >  AppServiceImpl.java

Service 层调用 DAO 层并封装所有的业务逻辑，它被 Action 调用。

- ▼  bookstore.model.result
  - >  BookDetail.java
  - >  FailureMessage.java
  - >  SuccessMessage.java
  - >  UserDetail.java
- ▼  bookstore.util
  - >  HashUtil.java
  - >  PasswordUtil.java
  - >  StringUtil.java
  - >  Validator.java

辅助性的数据模型及辅助函数。

### 4.3 前端资源组织



前端的 JavaScript 脚本和 JSP 视图文件按照功能和作用进行了模块化的拆分和组织。

## 5.考核点与实现细节

### 5.1 注册/登录/登出

在未登录状态下，可在导航栏右上方点击“注册”或“登录”按钮进行注册/登录。在已登录状态下，从右上方下拉框点击“登出”以登出系统。

登录/登出使用 HttpSession 的 `setAttribute()` 和 `removeAttribute()` 方法实现。需要浏览器 Cookie 和服务器 Session 的配合。

### 5.2 管理个人信息

在登录状态下任意页面的右上角下拉菜单中点击“管理个人信息”“管理收货地址”，即可管理相应内容。

## 我的个人信息



用户名

admin

密码

填写此栏以修改密码

确认

填写此栏以确认修改密码


昵称

系统管理员


余额

¥ 90520.00

身份

 管理员



 支持 JPG、PNG、BMP、GIF 图片格式，大小不超过 2M

更换我的头像

删除我的头像

保存

取消

## 我的收货地址



上海市闵行区东川路800号上海交通大学xx栋学生宿舍



上海市闵行区东川路800号上海交通大学软件学院x号楼xxxx



火星表面第42号坑洞



保存

取消

本系统中的用户信息包括用户名、密码、昵称、头像、余额、身份和收货地址。其中密码不可查看，只能重置。余额和身份仅管理员可修改。由于收货地址为个数不确定的数组，具有较高的非结构化特性，故选择将其存储在非关系型数据库 MongoDB 中，其他数据存储在关系型数据库 MySQL 中。

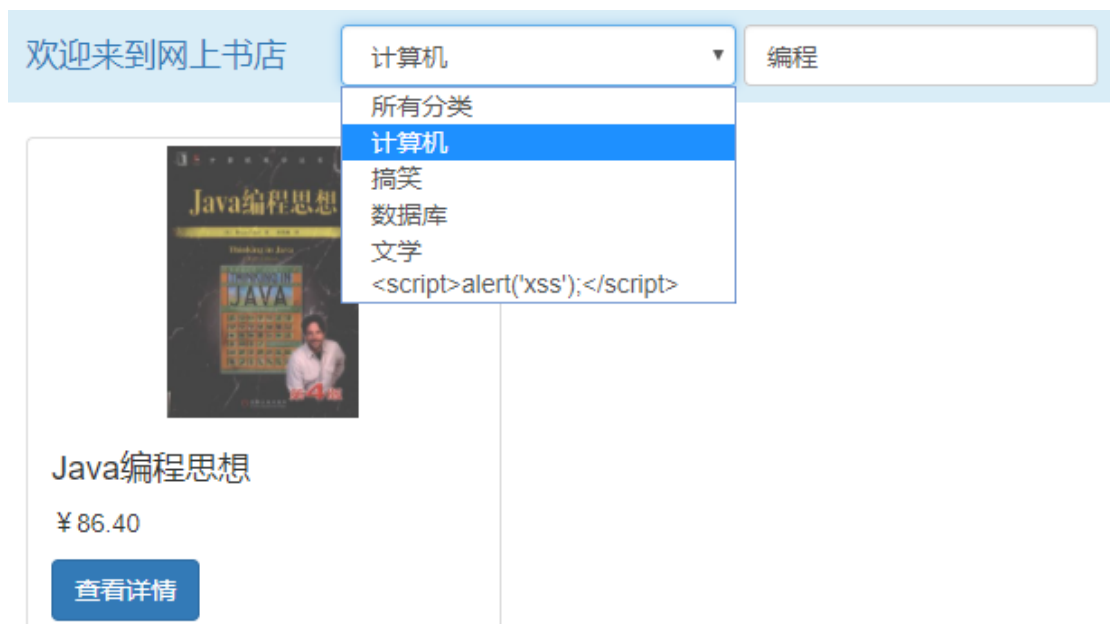
```
C:\WINDOWS\system32\cmd.exe - mongo
> use bookstore;
switched to db bookstore
> db.userAddresses.find({})
{ "_id" : ObjectId("5960cee9d02ecd177b20087a"), "class" : "bookstore.model.Address", "userId" : 1, "address" : [ "上海市闵行区东川路800号上海交通大学xx栋学生宿舍", "上海市闵行区东川路800号上海交通大学软件学院x号楼xxxx", "火星表面第42号坑洞" ] }
{ "_id" : ObjectId("5960cf07d02ecd177b20087b"), "class" : "bookstore.model.Address", "userId" : 2, "address" : [ "找不到的", "不存在的" ] }
{ "_id" : ObjectId("5961ee15d02e0d3a5573523f"), "class" : "bookstore.model.Address", "userId" : 13, "address" : [ "<script>alert('xss');</script>" ] }
{ "_id" : ObjectId("596213f2d02e05c8a56cd620"), "class" : "bookstore.model.Address", "userId" : 14, "address" : [ "" or 1=1 --" ] }
{ "_id" : ObjectId("596213f6d02e05c8a56cd621"), "class" : "bookstore.model.Address", "userId" : 15, "address" : [ ] }
>
```

（头像作为图片，其二进制数据本身是非结构化的，但由于一个用户对应于一个头像，且我们不需要从头像二进制数据中解析其中的某部分数据，所以也可认为是一个整体数据块，是用户的一个属性，相当于结构化的数据。实现上将头像存入文件，然后在 MySQL 相应字段存储头像在文件系统中的路径。这样的实现有助于服务器和浏览器对静态资源进行缓存，且比存入 MongoDB 中要简便。）

在项目后续扩展中，如果要给用户增加一些新的不能完全预知的，或者其他非结构化的属性，也可存入非关系型数据库 MongoDB 中。

### 5.3 浏览、搜索书籍

在首页可浏览书籍，在顶端导航栏可按分类筛选、按关键词搜索书籍。



关键词搜索的实现方式：使用 SQL/HQL 的 LIKE %keyword% 查询语句进行简单的子串匹配，细节上考虑到了“%”和“\_”等特殊符号的转义问题。

### 5.4 书籍详情查看、添加书籍到购物车

在首页点击“书籍详情”按钮后，弹出模态框显示书籍详情，点击“加入购物车”按钮将该书籍加入到购物车。可以填写购买数量。

书籍详情

## MySQL：从删库到跑路

作者：暂无作者信息    出版社：暂无出版社信息

¥ 65.30

分类：计算机 搞笑 数据库

听说最近数据库界忽然流传起来一本书，叫《MySQL：从删库到跑路》。我看了前几章节，大概介绍的暗黑界的程序员怎么写坑，怎么悄无声息用锅给老实忠厚的程序员，还有最后怎么删除数据库之后遁入尘世不见踪影，手法之巧妙，策略之精巧，简直就是删库于无形之中。记得第一章有段序是这样写的“来呀 快活呀 反正有大把bug”，我看后吓出一身冷汗。

数量

加入购物车

返回

书籍详情通过一次 Ajax 请求从服务器异步地获取，使用 JSON 作为数据交换格式。（实际上，此项目中绝大部分数据都是用 Ajax 获取和发送的，在同一页面内类似于一个单页应用，前后端使用类似 RESTful 格式的 API 进行数据交互。）

购物车存储在关系型数据库 MySQL 中，每个用户对应于一个购物车（即一笔未支付的订单），所以添加书籍到购物车必须先登录。

### 5.5 下单购买

在“我的购物车”页面可查看当前购物车状态，修改某一书籍的购买数量，删除某个项目，或选择结算购物车。结算时会弹出确认框。

欢迎来到网上书店

购物车 2 系统管理员

## 我的购物车

书籍	单价	数量	小计	操作
MySQL：从删库到跑路	65.30	3	195.90	<div>+</div> <div>-</div>
黑客攻防：从入门到入狱	88.60	1	88.60	<div>+</div> <div>-</div>

总价：¥ 284.50

结算



## 结算



总项目数 2

总数量 4

总价 ¥284.50

确认支付

返回

支付时会检查各书籍库存是否充足，用户余额是否足够，均满足条件才能支付成功，否则失败（以上操作构成一个数据库事务）。支付成功的订单可以在“我的历史订单”页面查看详情。

### 5.6 管理员管理书籍和用户

可以在管理平台中通过表格、按钮和模态框等 UI 界面对书籍和用户进行增删改查操作。

（仅展示“书籍管理”相关界面，“用户管理”同理）

网上书店管理系统 书籍管理 分类管理 用户管理 订单查询 销售统计 系统管理员

### 书籍管理 +

编号	书名	单价	库存量	创建时间	修改时间	操作
1	MySQL：从删库到跑路	65.30	5	2017-05-02 16:28:00	2017-07-09 19:16:38	...
2	黑客攻防：从入门到入狱	88.60	2	2017-05-02 16:30:22	2017-06-01 19:55:11	...
4	Java编程思想	86.40	97	2017-05-02 16:37:57	2017-06-01 11:35:44	...
5	深入理解计算机系统（原书第3版）	111.20	199	2017-05-02 16:41:20	2017-07-09 16:46:02	...
6	活着	10.00	25	2017-05-02 16:43:29	2017-06-01 11:36:04	...
7	津巴多普通心理学（原书第7版）	89.30	145	2017-05-02 16:46:03	2017-06-01 11:36:14	...
8	<script>alert('xss!')</script>	0.10	0	2017-05-02 16:47:36	2017-07-09 16:59:51	...
9	"; drop database bookstore --	0.20	0	2017-05-02 16:48:14	2017-07-09 16:59:51	...

## 添加书籍

书名	<input type="text" value="书籍名称"/>	<div>暂无图片</div> <div>支持 JPG、PNG、BMP、GIF 图片格式，大小不超过 2M</div> <div>上传书籍图片</div>
作者	<input type="text" value="书籍作者"/>	
出版社	<input type="text" value="书籍出版社"/>	
单价	<input type="text" value="¥ 书籍单价"/>	
库存	<input type="text" value="库存量"/> 本	
分类	请至“分类管理”面板为此书籍添加分类信息。	

写一些关于书籍的简要介绍

添加

取消

## 书籍详情 - 编号：4

书名	<input type="text" value="Java编程思想"/>	 <div>支持 JPG、PNG、BMP、GIF 图片格式，大小不超过 2M</div> <div>更换书籍图片</div> <div>删除书籍图片</div>
作者	<input type="text" value="Bruce Eckel 著 陈昊鹏 译"/>	
出版社	<input type="text" value="机械工业出版社"/>	
单价	<input type="text" value="¥ 86.40"/>	
库存	<input type="text" value="97"/> 本	
分类	<input type="text" value="计算机"/>	

请至“分类管理”面板管理此书籍的分类信息。

本书赢得了全球程序员的广泛赞誉，即使是最晦涩的概念，在Bruce Eckel的文字亲和力和小而直接的编程示例面前也会化解于无形。从Java的基础语法到最高级特性（深入的面向对象概念、多线程、自动项目构建、单元测试和调试等），本书都能逐步指导你轻松掌握。  
从本书获得的各项大奖以及来自世界各地的读者评论中，不难看出这是一本经典之

创建时间：2017-05-02 16:37:57

修改时间：2017-06-01 11:35:44

保存

取消

## 5.7 管理员查看销售统计

在管理平台的“销售统计”页面中，管理员可按照书籍、书籍分类、用户和起止时间对销售数据进行筛选统计。

### 销售统计

按分类筛选

所有分类

2017/05/01

-

2017/07/10

筛选

购买人数：2    总销量：18    总金额：¥ 1279.10

按书籍筛选

1

2017/05/01

-

2017/07/10

筛选

购买人数：2    总销量：3    总金额：¥ 195.90

按用户筛选

testuser

2017/05/01

-

2017/07/10

筛选

订单数：1    总订单项数：2    总购买数量：4    总金额：¥ 331.10

由于订单数据总量可能很大，设计上采用调用数据库存储过程的方式进行筛选和统计值计算，封装复杂的业务逻辑，将计算交给 DBMS 去做，传输统计值而不是传输大量数据，这样能显著提升应用的性能。

```
DROP PROCEDURE IF EXISTS statAll;
DROP PROCEDURE IF EXISTS statCategory;
DROP PROCEDURE IF EXISTS statBook;
DROP PROCEDURE IF EXISTS statUser;

DELIMITER $$

CREATE PROCEDURE statAll(IN start_time TIMESTAMP, IN end_time TIMESTAMP,
    OUT stat_person INT UNSIGNED, OUT stat_quantity INT UNSIGNED, OUT stat_price INT UNSIGNED)
BEGIN

    SELECT COUNT(DISTINCT user_id), SUM(quantity), SUM(quantity * price)
    INTO stat_person, stat_quantity, stat_price
    FROM `Order` JOIN OrderItem USING (order_id)
    WHERE `Order`.is_paid = 1 AND `Order`.updated_at >= start_time AND `Order`.updated_at <= end_time;

END $$

CREATE PROCEDURE statCategory(IN stat_category INT(11) UNSIGNED, IN start_time TIMESTAMP, IN end_time TIMESTAMP,
    OUT stat_person INT UNSIGNED, OUT stat_quantity INT UNSIGNED, OUT stat_price INT UNSIGNED)
BEGIN

    SELECT COUNT(DISTINCT user_id), SUM(quantity), SUM(quantity * price)
    INTO stat_person, stat_quantity, stat_price
    FROM `Order` JOIN OrderItem USING (order_id)
    WHERE `Order`.is_paid = 1 AND `Order`.updated_at >= start_time AND `Order`.updated_at <= end_time
    AND (
        SELECT book_id
        FROM BookCategory
        WHERE category_id = stat_category
    );

END $$
```

### 5.8 其他设计亮点

- 1. 存储时对密码进行哈希处理，防止数据库被攻击时能直接获取到用户的密码，提升了应用的安全性。（实现上采用了 SHA-256 算法进行哈希，可改进为使用 BCrypt 算法。）
- 2. 上传文件采用了哈希值、时间戳和随机数三者结合的命名方案，尽可能避免文件重名冲突。
- 3. 在后端对所有的用户输入内容做了细致的有效性验证，提升了应用的健壮性。
- 4. 在前端处理了 Ajax 异常，并对空值有适当的占位符显示（如书籍图片、作者或出版社未设置的情形），提升了用户友好性。

书籍详情

暂无图片

无字天书

作者：暂无作者信息    出版社：暂无出版社信息

¥ 0.00

分类：暂无分类信息

暂无该书籍的介绍信息

数量1

加入购物车

返回