

Bookstore 开发文档

项目简介

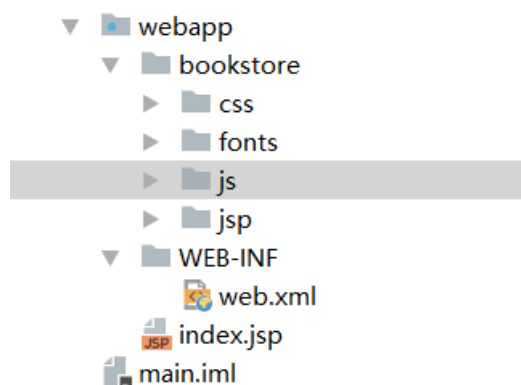
该 web 应用是采用 SSH 框架开发的一个简易在线书店系统，对于书店用户实现了登录、购书以及修改个人信息的功能，对于管理员实现了对用户、书籍以及订单信息的管理功能。

部署环境

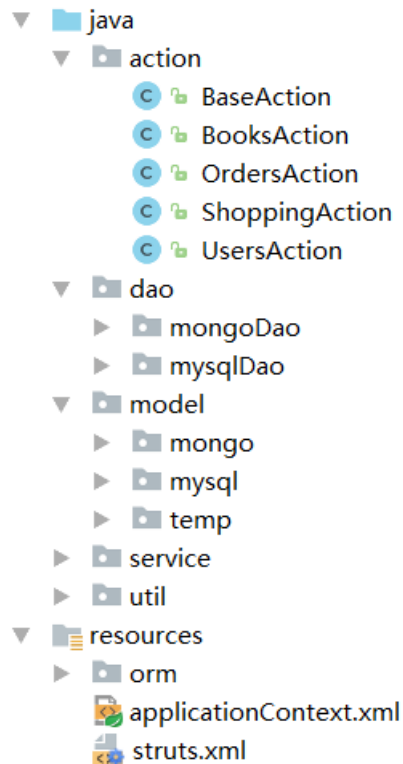
- 开发平台 ： IntelliJ IDEA 2017.1
- 应用服务器 ： Tomcat 9.0.0.M19
- 数据库 ： MySQL ， MongoDB

项目结构

- 前端页面结构如下图所示：



- 后端结构如下图



Action 层用于控制页面跳转；Dao 层用于与数据库进行交互，按数据源分为了 mongo DB 的 Dao 和 MySQL 的 Dao；Model 层用于表示应用中用到的一些实体类，mongo 包与 mysql 包下的 model 分别对应要保存到对应数据库中的持久化类，temp 包中是一些在应用运行过程中产生的临时类；Service 层用于实现业务逻辑；Util 层中是一些其他的类，其中 Service 层与 Dao 层的类均采用了接口与实现分离的设计。

在 resources 中，orm 包中是 hibernate 的配置文件，用于将 MySQL 中的表与实体类相对应。

功能实现情况

1 用户功能

1.1 用户注册与登录

1.1.1 注册

注册页面如图所示

Please sign up

Username

Password

Confirm Password

Phone

Address

用户提交表单后,Struts 将表单交给 UsersAction 中的 signup 方法处理,

```
public String signup() throws Exception {
    int signStat = userService.signup(username, password, password2, phone, address);
    if(signStat == 2) {
        this.addActionError( anErrorMessage: "Two passwords differs!");
        return ERROR;
    }
    else if (signStat == 1) {
        this.addActionError( anErrorMessage: "Username already exists!");
        return ERROR;
    }
    else {
        return SUCCESS;
    }
}
```

之后 userService 会判断两个密码是否相等, 以及用户名是否存在, 若密码相等且用户名不存在则成功注册, 并返回相应状态。随后 struts 由注册是否成功选择跳转到登录页面还是留在注册页面。

1.1.2 登录

用户输入用户名和密码经后台验证成功即可登录,在登录后, 用户的信息将被置于 Session 中保存起来, 如图所示:

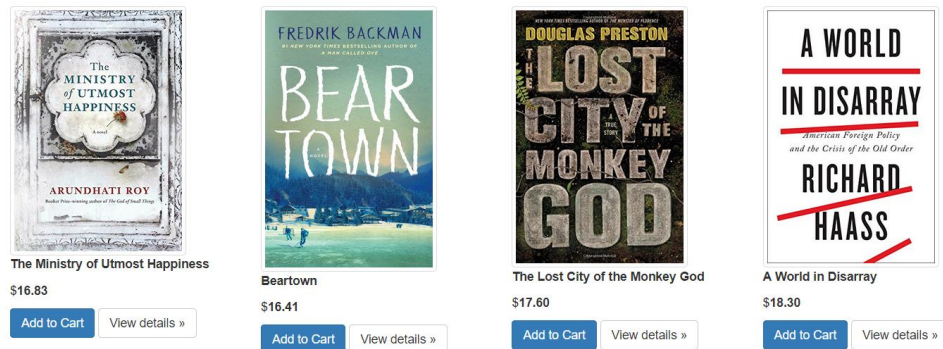
```
public boolean login(String username, String password) {
    User user = getUserByName(username);
    if (user != null && user.getPassword().equals(password)) {
        Map<String, Object> session = ActionContext.getContext().getSession();
        session.put("user", user);
        return true;
    }
    return false;
}
```

1.2 浏览、查找书籍

1.2.1 浏览书籍

书店主页页面如图所示

Books



在加载主页时，会调用一个 action 以获取所有书籍

```
<s:action name="books" executeResult="true"/>
```

该 action 会调用 BooksAction 中的 allBookInfos()方法，将 BooksAction 中的变量 bookInfos 置为所需的结果值，这样前端就可以直接调用 BooksAction 的 getBookInfos()方法获取到包含所有书信息的列表。

```
public String allBookInfos() {
    bookInfos = bookService.getAllBookInfos();
    return SUCCESS;
}
```

之后在前端用 <s:iterator value="bookInfos"> 可对该列表进行迭代并用

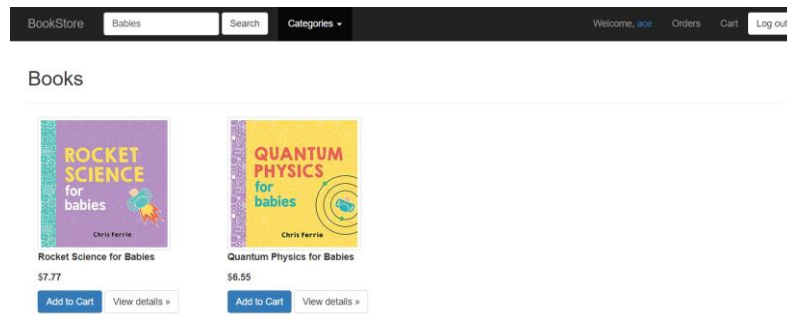
<s:property value="bookId"/> 这样的方法获取每个 bookInfo 对象的属性，从而将主页中的书籍信息构建出来。

1.2.2 查找书籍

有两种方式可用于查找书籍，第一种是在搜索框中输入书籍所包含的关键词，提交关键词之后，后台会查找所有书名中包含该关键词的书籍，之后将这些书放到列表中，采用和上面类似的方法返回给前端。

```
public List<BookInfo> searchBookFor(String keyword) {
    List<BookInfo> resultBooks = new ArrayList<>();
    List<BookInfo> allBooks = this.getAllBookInfos();
    for (BookInfo book : allBooks) {
        if (book.getTitle().contains(keyword)) {
            resultBooks.add(book);
        }
    }
    return resultBooks;
}
```

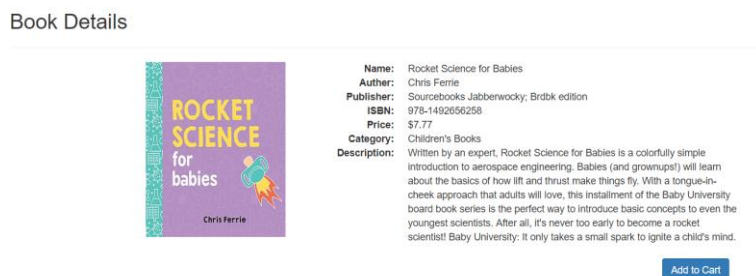
实际效果如下图



第二种是选择对应的书籍种类进行查找，提交种类后，后台将所有该种类书籍放到列表中返回给前端。

1.2.3 查看书籍详细信息

点击书籍图片即可进入详情页面,如图



进入详情页对应的 action 会将该书对应的 book 和 bookDetail 对象放到 BooksAction 的成员变量中去，以便前端能取到相应的信息。

```
public String getMoreInfo() {
    book = bookService.getBookById(bookId);
    bookDetail = bookService.getBookDetailById(bookId);
    return SUCCESS;
}
```

1.3 购物车

购物车是用保存在 Session 中的一个 Cart 对象来实现的，Cart 对象的成员变量是一个 Orderitem 的数组，每个 Orderitem 对象包含的成员变量有 recordId, bookId,





Tilte, amount, unitPrice，其中除 recordId 是在结账之后生成的之外，其他的属性都是对象在初始化时指定的。控制新增书籍到购物车的 action 如图所示

```
public String addToCart() throws Exception {
    Map<String, Object> session = ActionContext.getContext().getSession();
    Cart cart = (Cart) session.get("cart");
    if (cart == null) {
        cart = new Cart(new ArrayList<Orderitem>());
    }
    cart = cartService.addBook(cart, bookId);
    session.put("cart", cart);
    return SUCCESS;
}
```

cartService 会根据 bookId 生成一个 orderitem 放到 cart 的 orderitem 数组中，之后再增加该商品数量只需改变 orderitem 的 amount 属性即可，而移除商品也就是将对应的 orderitem 从数组中移出。

用户可在购物车页面看到当前购物车中的商品以及总价状况，如图

Cart

Name	Price	Amount	Operation
Beartown	\$16.41	<input type="text" value="1"/>	 
The Lost City of the Monkey God	\$17.60	<input type="text" value="3"/>	 
Total Price : \$69.21			
<button>Checkout</button>			

在给页面进行的操作会实时更新 session 中 cart 的状态。

1.4 结账

在购物车页面点击 checkout 即可结账，orderService 会生成一个新的 Order 到 MySQL 数据库中，然后将 cart 中的所有 orderitem 以 JSON 字符串的形式保存到 mongo DB 中。


```
public void checkout(User user, Cart cart) {  
    Order order = new Order();  
  
    order.setUserId(user.getUserId());  
    order.setDate(new Timestamp(System.currentTimeMillis()));  
  
    int orderId = addOrder(order);  
    ArrayList<Orderitem> items = cart.getItems();  
    for(Orderitem item : items) {  
        addOrderitem(orderId, item);  
    }  
}
```

在保存每个 orderitem 时，会为其指定一个 recordId，然后生成一个新的 Record 对象，并转为 JSON 字符串保存到对应 book 在 mongo DB 中的 bookDetail 中去，从而实现了销售记录。

1.5 用户个人信息的管理

用户可进入个人信息页面管理个人信息，如图

Personal Details


[Change Avatar](#)

User ID 15

Username ace

Phone 15325685588

Address SJTU

Profile Books are the ladders of human's progress.

[Modify](#) [Change password](#)

用户头像是以 BASE64 字符串形式保存在 mongo DB 中的，用户可通过指定网络图片 URL 的形式来修改图片，Util 类中的 ImageUtil 会从指定 URL 下载图片并转为 BASE64 字符串保存。修改密码需要用户提供原始密码和新密码。

2 管理员功能

2.1 书籍和用户的管理




2.1.1 书籍管理

书籍管理页面如图所示

Books


add book +

Show 10 entries Search:

ID	Title	Author	Price	Publisher	Date	isbn	category	
15	The Ministry of Utmost Happiness	Arundhati Roy	1683	Knopf; 1st Edition edition (June 6, 2017)	17-7-15 23:43:55.000	978-1524733155	Literature & Fiction	  

实现了基本的增删改查功能，可点击右下角的按钮进入详情页，如图

Book detail



Written by an expert, Rocket Science for Babies is a colorfully simple introduction to aerospace engineering. Babies (and grownups!) will learn about the basics of how lift and thrust make things fly. With a tongue-in-cheek approach that adults will love, this installment of the Baby University board book series is the perfect way to introduce basic concepts to even the youngest scientists. After all, it's never too early to become a rocket scientist! Baby University: It only takes a small spark to ignite a child's mind.

[Change cover](#) [Change description](#) [View sale record](#) [Close](#)

在这里进行修改封面和修改描述的操作和用户修改个人信息的操作类似，还可以查看书籍的销售记录,如图

Sale record ✕

Recordid	Userid	Amount	Date
1	14	2	2017-07-16 04:02:21.06

TotalAmount : 2

Close







2.1.2 用户管理

用户信息管理界面如图

Users


add user +

Show 10 entries Search:

ID	Username	Password	Role	Phone	Address	
14	admin	123	admin	15684778865	xxxx	  
15	ace	123	user	15325685688	SJTU	  

也可通过右侧按钮进入详情页面，如图

User detail ✕



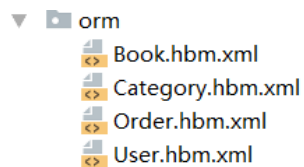
this is me

Change avator Change profile Close

技术要求完成情况

1. OR 映射

保存在 MySQL 中的数据与对应的实体类采用了 Hibernate 进行 OR 映射，总共对四个类进行了映射，如图



映射过程都只是将表中的属性与对象的成员变量一一对应，唯一有一点不同的是，在 Order 对象中有一个 Orderitem 变量是被用作临时变量使用的，故没有与表中的属性映射。如图

2. MongoDB

MongoDB 在该应用中用于保存 userDetails, BookDetail 和 OrderDetail 三个类的信息，对应结构如图

```

_id: ObjectId('596a383bae54870eb2698')
  _class: 'model.mongo.BookDetail'
bookId: 15
records: Array
description: "The Ministry of Utmost Happiness takes us on an intimate journey of many years across the Indian subcontinent—from the cramped neighborhoods of Old Delhi and the roads of the new city to the mountains and valleys of Kashmir and beyond, where war is peace and peace is war.

It is an aching love story and a decisive reexamination, a story told in a whisper, in a shout, through unsentimental tears and sometimes with a bitter laugh. Each of its characters is indelibly, tenderly rendered; its heroes are people who have been broken by the world they live in and then rescued, patched together by acts of love—and by hope.

The tale begins with Anjum who used to be Aftab—unrolling a threadbare Persian carpet in a city graveyard she calls home. We encounter the odd, unforgettable Tilo and the men who loved her—including Musa, sweetheart and ex-sweetheart, lover and ex-lover; their fates are as entwined as their arms used to be and always will be. We meet Tilo's landlord, a former suitor, now an intelligence officer posted to Kabul. And then we meet the two Miss Jeben: the first a child born in Srinagar and buried in its overcrowded Martyrs' Graveyard; the second found at midnight, abandoned on a concrete sidewalk in the heart of New Delhi.

As this ravishing, deeply humane novel brads these lives together, it reinvents what a novel can do and can be.

The Ministry of Utmost Happiness demonstrates on every page the miracle of Arundhati Roy's storytelling gifts."
image: "/9j/4AAQSkZJRgABAQAAQABAD/2wBDAAUDBAQEAWUEBAQFBQUGBwI2BwCHBwSLCkHEQSEHEP

_id: ObjectId('596a74cdae5486f5c7fd7b7')
  _class: "model.mongo.OrderDetail"
orderId: 15
▼ orderItems: Array
  0: {"recordId":1,"bookId":22,"title":"Rocket Science for Babies","amount":2,"unitPrice":777}
  1: {"recordId":1,"bookId":23,"title":"Quantum Physics for Babies","amount":3,"unitPrice":655}

_id: ObjectId('59611c399ffbf9e48747d3ae')
  _class: "model.mongo.UserDetail"
userId: 14
profile: "this is me"
avatar: "/9j/4AAQSkZJRgABAQAAQABIAAD/4QBXRhphZAAU0AKGAAAAAGAAESAAAMAAAABAAEAAIdpAAQAA
AAABAAQAAQAAAGAAABAAAMAAAABAAEAAKACAAQAAAAABAAAAAYKADAAQAAAAABAAAAAYAAAAAD/7QA4
"

```

3. MVC 框架

该项目采用 Struts2 作为 MVC 框架，Model 层即之前提到的 model 包中的类，View 层即 JSP 页面，Controller 层即 Struts 配置文件以及 action 包中的类。

Struts 配置文件确定了某个 action 该由哪个 action 类处理，这些 action 的处理方式有两种，第一种是处理之后返回一个页面，这类 action 放到了

`<package name="process" extends="struts-default">` 这个 package 下；另一种是处理之后返回 JSON 数据，这类 action 放到了 `<package name="textjson" extends="json-default"...>` 这个 package 下。

在调用指定 Action 类中的方法时，表单提交的内容经由 Action 中的 set 方法传递到了 Action 类中的成员变量中，在该 Action 返回的页面中，可用 get 方法获得 Action 类中成员变量的值，从而实现前后端数据交互。

4. IoC 框架

该项目采用 Spring 进行依赖注入，在 applicationContext.xml 中采用如下方式进行注入

```
<bean id="bookDao" class="dao.mysqlDao.impl.BookDaoImpl">
  <property name="sessionFactory" ref="sessionFactory" />
</bean>
```

图中将 sessionFactory 注入到了 bookDao 中，之后只要在 bookDao 的成员变量中加一个 sessionFactory 并提供对应 setter 即可完成注入。

5. AJAX 和 JSON 的使用

该项目在多处使用到了 AJAX 和 JSON 技术，比如在管理员管理书籍信息的界面，查看书籍详情这一动作就是通过 AJAX 发起请求，后台返回 JSON 数据然后在前台解析实现的，如图

```
$('#info').click(function (e) {
    var dataset = g.currentTarget.dataset;
    var id = dataset.id;
    $('#modal-detail').attr('bid', id);
    $.ajax({
        url: 'getBookDetail',
        processData: true,
        context: this,
        dataType: 'text',
        data: {
            bookId: id
        },
        success: function (data) {
            var bookDetail = JSON.parse(data);
            bookRecords = bookDetail.records;
            $('#bookImage').attr('src', 'data:image/jpg;base64,' + bookDetail.image);
            $('#descriptionContent').text(bookDetail.description);
            $('#modal-detail').modal('show');
        }
    });
});
```