实验 15、16 Spring 综合编程

一、实验目的

1、理解 Spring 容器的概念。

2、掌握 Struts 2、Hibernate 和 Spring 整合方法。

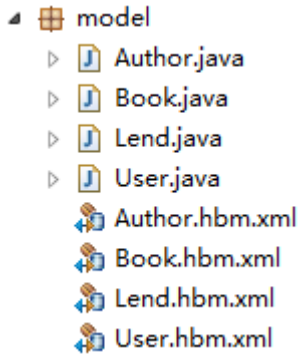
二、实验内容

综合运用 Struts 2、Hibernate 和 Spring 三大框架技术，设计并实现一个简单的图书应用系统，实现管理员对图书的管理功能和用户对图书的查询借还功能。

三、实验过程及要求

1、JavaBean 设计及 ORM 映射文件

如下图所示，设计各个实体类，并编写其与数据库表的 ORM 映射文件。



（1）用户类 User 及 ORM 映射文件。

```
public class User {
   private String name;
   private String password;
   private String role;
   public User(String name, String password, String role) {
        this.name = name;
        this.password = password;
        this.role = role;
   }
   public User() {
   }
  //setter 和 getter 方法
}
<hibernate-mapping>
   <class name="model.User" table="usr" schema="app">
        <id name="name" type="string">
             <column name="name" />
             <generator class="assigned" />
        </id>
```

```
            <property name="password" type="java.lang.String">
                <column name="password" />
            </property>
            <property name="role" type="java.lang.String">
                <column name="role" />
            </property>
        </class>
    </hibernate-mapping>
```

（2）作者类 Author 及 ORM 映射文件。

```
    public class Author {
        private String name;
        private String tel;
        private String email;
       //两个构造方法
      //setter 和 getter 方法
    }
    <hibernate-mapping>
        <class    name="model.Author"    table="author"    schema="app"
    lazy="false">
            <id name="name" type="string" column="name">
                <generator class="assigned" />
            </id>
            <property name="tel" type="string" column="tel" />
            <property name="email" type="string" column="email" />
        </class>
    </hibernate-mapping>
```

（3）书籍类 Book 及 ORM 映射文件。

```
    public class Book {
        private String isbn;
        private String title;
        private int price;
        private String intro;
        private Author author;
       //两个构造方法
      //setter 和 getter 方法
    }
    <hibernate-mapping>
        <class name="model.Book" schema="app">
            <id name="isbn" type="java.lang.String" column="isbn">
                <generator class="assigned"></generator>
            </id>
            <property name="title" type="string" column="title" />
            <property name="price" type="int" column="price" />
            <property name="intro" type="string" column="intro" />
```

```
        <many-to-one name="author" cascade="all"
class="model.Author" column="name" />
    </class>
</hibernate-mapping>
```

（4）借阅信息类 Lend 及 ORM 映射文件。

```
public class Lend {
    private int bookId;
    private String name;
    private String isbn;
    private Date ltime;
   //两个构造方法
  //setter 和 getter 方法
}
<hibernate-mapping>
    <class name="model.Lend" table="lend" schema="app">
        <id name="bookId" type="int">
            <column name="bookid" />
            <generator class="identity" />
        </id>
        <property name="name" type="java.lang.String">
            <column name="name" />
        </property>
        <property name="isbn" type="java.lang.String">
            <column name="isbn" />
        </property>
        <property name="ltime" type="java.util.Date">
            <column name="ltime" />
        </property>
    </class>
</hibernate-mapping>
```

2、数据库表设计

（1）用户表 usr。其中，角色 role 只能取“读者”和“管理员”。

```
create table app.usr(
    name varchar(20),
    password varchar(6) not null,
    role varchar(10) not null default '读者',
    primary key(name)
);
insert into app.usr values('李明', '123456', '管理员');
```

NAME	PASSWORD	ROLE
李明	123456	管理员
Rose	rose	读者
Tom	123456	读者
admin	admin	管理员

（2）作者表 author。

```
create table app.author(
    name varchar(10),
    tel varchar(20),
    email varchar(50),
    primary key(name)
);
```

NAME	TEL	EMAIL
谢希仁	7788	xie@qq.com
王珊	5566	wang@qq.com
赵刚	8877	zg@qq.com
李平	6655	zhao@qq.com
周伟	5588	zw@qq.com

（3）书籍表 book。

```
create table app.book(
    isbn varchar(10),
    title varchar(50) not null,
    price smallint not null,
    intro varchar(20) not null,
    name varchar(10),
    primary key(isbn)
);
```

ISBN	TITLE	PRICE	INTRO	NAME
9001	计算机网络	58	hello.txt	谢希仁
9002	数据库原理	37	hello.txt	王珊
9003	大数据	40	hello.txt	李平
9004	离散数学	36	hello.txt	赵刚
9008	计算机组成原理	52	hello.txt	周伟

（4）借阅表 lend。其中，bookid 自增长，ltime 为借阅时间。

```
create table app.lend(
    bookid integer generated always as identity (start with 1,
increment by 1) primary key,
    name varchar(20),
    isbn varchar(10),
    ltime date
);
```

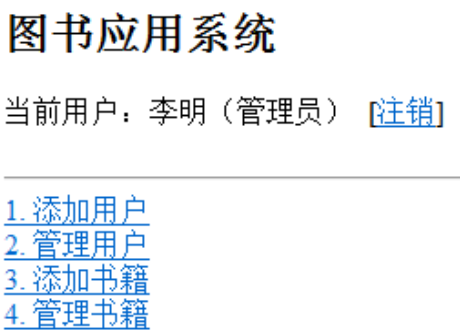
BOOKID	NAME	ISBN	LTIME
25	Rose	9004	2018-11-10
27	Rose	9001	2018-11-10

3、JSP 页面设计

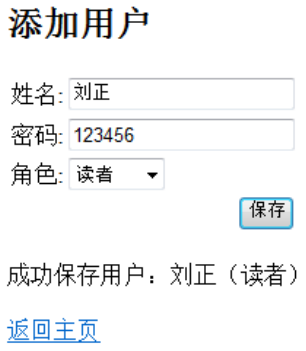
（1）用户登录页面 login.jsp

（2）图书应用系统主页 main.jsp

如下图所示，根据不同的角色，显示不同的功能界面。



（3）添加用户页面 addUser.jsp（管理员）



（4）管理用户页面 listUser.jsp（管理员）

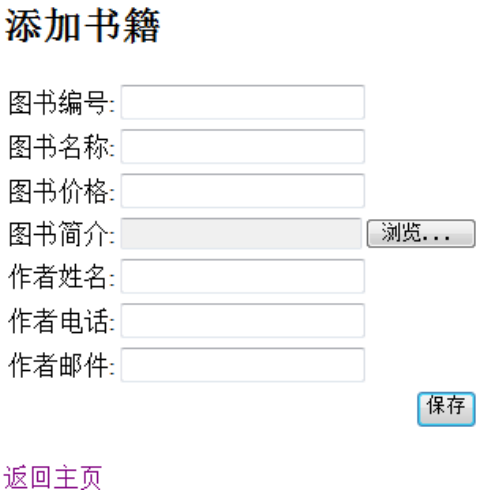
此处，仅能修改用户密码。

（5）修改用户页面 updateUser.jsp（管理员）

用户姓名和角色为只读（readonly）。

删除用户 Tom，界面如下。

（6）添加书籍页面 addBook.jsp（管理员）



（7）管理书籍页面 listBook.jsp（管理员）

此处，仅能修改用户密码。

管理书籍

搜索条件： 查询

图书编号	图书名称	图书价格	作者姓名	作者电话	作者邮件	操作		
9001	计算机网络	58	谢希仁	7788	xie@qq.com	简介	修改	删除
9002	数据库原理	37	王珊	5566	wang@qq.com	简介	修改	删除
9003	大数据	40	李平	6655	zhao@qq.com	简介	修改	删除
9004	离散数学	36	赵刚	8877	zg@qq.com	简介	修改	删除
9008	计算机组成原理	52	周伟	5588	zw@qq.com	简介	修改	删除

查询全部 返回主页

（8）修改书籍页面 updateBook.jsp（管理员）

图书编号为只读（readonly）。

修改书籍

图书编号: 9008
图书名称: 计算机组成原理
图书价格: 52
图书简介: 浏览...
作者姓名: 周伟
作者电话: 5588
作者邮件: zw@qq.com
修改

图书管理 返回主页

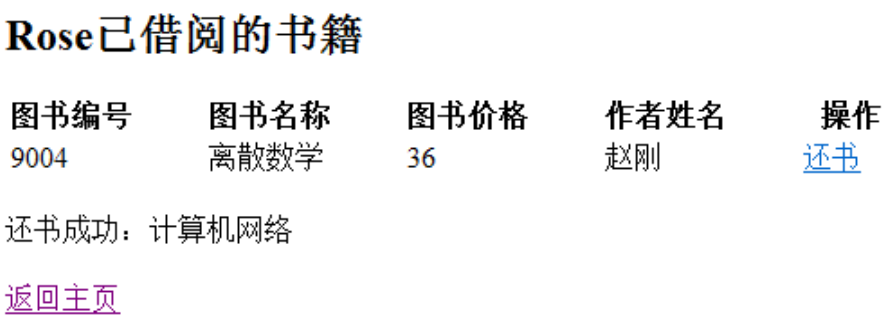
（9）已借书籍页面 lendedBook.jsp（读者）

此处，用户 Rose 登录，显示其所借的图书。点击“还书”，实现还书功能。

Rose已借阅的书籍

图书编号	图书名称	图书价格	作者姓名	操作
9001	计算机网络	58	谢希仁	还书
9004	离散数学	36	赵刚	还书

返回主页



（10）借阅书籍页面 lendBook.jsp（读者）

此处，用户 Rose 登录，显示其未借的所有图书。点击“借书”，实现借书功能。

4、Action 类设计

（1）UserAction 类。

```
public class UserAction extends ActionSupport {
    private String message;
    private User user;
    private ArrayList<User> users;
    private String condition = "";
    private IUserDAO userDAO;
```

（2）BookAction 类。

```
public class BookAction extends ActionSupport {
    private String message;
    private Book book;
    private ArrayList<Book> books;
    private String condition = "";
    private File bookIntro;
    private String bookIntroFileName;
    private String bookIntroContentType;
    private String bookIntroContent = "";
    private BookDAO bookDAO;
```

（3）LendAction 类。

```
public class LendAction extends ActionSupport {
    private String message;
    private Book book;
    private User user;
    private ArrayList<Book> books;
    private String condition = "";
    private String bookIntroContent = "";
    private LendDAO lendDAO;
```

5、Spring 的 bean 定义

在目录 src 下，创建 XML 文件 applicationContext.xml。

```
<bean id="bookDAO" class="dao.BookDAO" />
<bean id="bookAction" class="action.BookAction">
    <property name="bookDAO" ref="bookDAO" />
</bean>

<bean id="userDAO" class="dao.UserDAO" />
<bean id="userAction" class="action.UserAction">
    <property name="userDAO" ref="userDAO" />
</bean>

<bean id="lendDAO" class="dao.LendDAO" />
<bean id="lendAction" class="action.LendAction">
    <property name="lendDAO" ref="lendDAO" />
</bean>
```

6、拦截器设计

同实验 14。

7、数据库连接

编写 Hibernate 配置文件 hibernate.cfg.xml。

```
<hibernate-configuration>
    <session-factory>
        <property name="dialect">org.hibernate.dialect.DerbyD
        <property name="connection.driver_class">org.apache.d
        <property name="connection.url">jdbc:derby://localhos
        <property name="connection.username">admin</property>
        <property name="connection.password">admin</property>
        <property name="show_sql">true</property>
        <mapping resource="model/Book.hbm.xml" />
        <mapping resource="model/User.hbm.xml" />
        <mapping resource="model/Author.hbm.xml" />
        <mapping resource="model/Lend.hbm.xml" />
    </session-factory>
</hibernate-configuration>
```

8、web.xml 文件

在 web.xml 文件中增加以下代码，实现 Spring 容器。

```
<listener>
    <listener-class>
        org.springframework.web.context.ContextLoaderListener
    </listener-class>
</listener>
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>
        /WEB-INF/classes/applicationContext.xml
    </param-value>
</context-param>
```

9、struts.xml 文件

示例：还书功能。

```
<action name="lendedBook" class="lendAction" method="listLendedBook">
    <result name="success">/lendedBook.jsp</result>
</action>
<action name="returnBook" class="lendAction" method="returnBook">
    <result name="success" type="redirectAction">
        <param name="actionName">lendedBook</param>
    </result>
    <result name="error" type="redirectAction">
        <param name="actionName">lendedBook</param>
    </result>
</action>
```

10、在 Tomcat 服务器上部署、在浏览器中测试 Web 应用。

11、完成实验内容后，写出实验过程、结论和心得。