book.sql文件：

/\*

SQLyog Ultimate v10.00 Beta1

MySQL - 8.0.26 : Database - book

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

/\*!40101 SET NAMES utf8 \*/;

/\*!40101 SET SQL\_MODE=''\*/;

/\*!40014 SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0 \*/;

/\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/;

/\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/;

/\*!40111 SET @OLD\_SQL\_NOTES=@@SQL\_NOTES, SQL\_NOTES=0 \*/;

CREATE DATABASE /\*!32312 IF NOT EXISTS\*/`book` /\*!40100 DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci \*/ /\*!80016 DEFAULT ENCRYPTION='N' \*/;

USE `book`;

/\*Table structure for table `t\_book` \*/

DROP TABLE IF EXISTS `t\_book`;

CREATE TABLE `t\_book` (

`id` int NOT NULL AUTO\_INCREMENT,

`name` varchar(100) CHARACTER SET utf8mb4 COLLATE utf8mb4\_general\_ci DEFAULT NULL,

`price` decimal(11,2) DEFAULT NULL,

`author` varchar(100) CHARACTER SET utf8mb4 COLLATE utf8mb4\_general\_ci DEFAULT NULL,

`sales` int DEFAULT NULL,

`stock` int DEFAULT NULL,

`img\_path` varchar(200) CHARACTER SET utf8mb4 COLLATE utf8mb4\_general\_ci DEFAULT 'static/img/default.jpg',

PRIMARY KEY (`id`) USING BTREE

) ENGINE=InnoDB AUTO\_INCREMENT=47 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci ROW\_FORMAT=DYNAMIC;

/\*Data for the table `t\_book` \*/

insert into `t\_book`(`id`,`name`,`price`,`author`,`sales`,`stock`,`img\_path`) values (23,'Java从入门到精通','25.90','张玉宏',11,989,'static/img/books/java-rumengdaojintong-zhangyuhong.jpg'),(24,'数据结构与算法','78.50','严蔚敏',34,966,'static/img/books/da-yanweimin.jpg'),(25,'MySQL从入门到精通','74.80','明日科技',12,988,'static/img/books/MySQL从入门到精通.png'),(26,'Python编程-从入门到实践','66.80','Eric Matthes',556,1000,'static/img/books/Python编程-从入门到实践.png'),(27,'C++编程思想','95.70','Bruce Eckel、Chuck Allison\n',681,1000,'static/img/books/C++编程思想.png'),(28,'Hive编程指南','34.50','Edward Capriolo',15,985,'static/img/books/Hive编程指南.png'),(29,'Scala和Spark大数据分析','66.50','雷扎尔·卡里姆',87,913,'static/img/books/Scala和Spark大数据分析.png'),(30,'Java编程思想','99.50','埃克尔',75,925,'static/img/books/Java编程思想.png'),(31,'疯狂H5+C3+JS讲义','41.00','李刚',216,1000,'static/img/books/疯狂HTML5+CSS3+JavaScript讲义.png'),(32,'Cocos2d-x游戏开发','55.80','刘克男',95,1000,'static/img/books/Cocos2d-x游戏开发.png'),(33,'C语言程序设计','28.00','谭浩强',156,1000,'static/img/books/C语言程序设计.png'),(34,'Lua程序设计','44.00','Roberto,Ierusalimschy',841,1000,'static/img/books/Lua程序设计.png'),(36,'机器学习 西瓜书','66.00','周志华',24,1000,'static/img/books/机器学习-西瓜书.png'),(37,'操作系统导论','49.50','雷姆兹·H.阿帕希杜塞尔',564,1000,'static/img/books/操作系统导论.jpg'),(38,'数据结构与算法分析','173.15','封大神',59,999,'static/img/books/数据结构与算法分析-Java语言描述.png'),(39,'UNIX环境高级编程','118.30','W.理查德·史蒂文斯',11,989,'static/img/books/UNIX环境高级编程.png'),(40,'JavaScript语言精粹','33.10','道格拉斯·克罗克福德',522,1000,'static/img/books/JavaScript语言精粹.png'),(41,'大话设计模式','24.00','程杰',653,1000,'static/img/books/大话设计模式.png'),(42,'Hadoop大数据技术开发实战','49.50','张伟洋',896,1000,'static/img/books/Hadoop大数据技术开发实战.png'),(46,'计算机网络 自顶向下方法','44.50','詹姆斯·F.库罗斯',189,999,'static/img/books/计算机网络-自顶向下方法.png');

/\*Table structure for table `t\_order` \*/

DROP TABLE IF EXISTS `t\_order`;

CREATE TABLE `t\_order` (

`order\_id` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4\_general\_ci NOT NULL COMMENT '订单id',

`create\_time` datetime DEFAULT '2020-01-01 00:00:00' COMMENT '创建时间',

`price` decimal(11,2) DEFAULT NULL COMMENT '订单价格',

`status` tinyint DEFAULT NULL COMMENT '订单状态,0未发货,1已发货,2已收货',

`user\_id` int DEFAULT NULL COMMENT '所属的用户id',

PRIMARY KEY (`order\_id`) USING BTREE,

KEY `user\_id` (`user\_id`) USING BTREE,

CONSTRAINT `t\_order\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `t\_user` (`id`) ON DELETE RESTRICT ON UPDATE RESTRICT

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci ROW\_FORMAT=DYNAMIC COMMENT='订单表';

/\*Data for the table `t\_order` \*/

insert into `t\_order`(`order\_id`,`create\_time`,`price`,`status`,`user\_id`) values ('16360997619564','2021-11-05 16:09:22','6395.50',2,4),('16361024784194','2021-11-05 16:54:38','4596.80',2,4),('16362871119234','2021-11-07 20:11:52','1549.25',0,4),('16365994816754','2021-11-11 10:58:02','1080.90',2,4),('16365995899624','2021-11-11 10:59:50','78.50',2,4),('163661377901311','2021-11-11 14:56:19','4574.00',2,11),('16384106695694','2021-12-02 10:04:30','74.80',1,4);

/\*Table structure for table `t\_order\_item` \*/

DROP TABLE IF EXISTS `t\_order\_item`;

CREATE TABLE `t\_order\_item` (

`id` int NOT NULL AUTO\_INCREMENT COMMENT '订单项id(主键)',

`name` varchar(200) CHARACTER SET utf8mb4 COLLATE utf8mb4\_general\_ci DEFAULT NULL COMMENT '商品名称',

`count` int DEFAULT NULL COMMENT '商品数量',

`price` decimal(11,2) DEFAULT NULL COMMENT '商品单价',

`total\_price` decimal(11,2) DEFAULT NULL COMMENT '商品总价',

`order\_id` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4\_general\_ci DEFAULT NULL COMMENT '所属的订单id',

PRIMARY KEY (`id`) USING BTREE,

KEY `order\_id` (`order\_id`) USING BTREE,

CONSTRAINT `t\_order\_item\_ibfk\_1` FOREIGN KEY (`order\_id`) REFERENCES `t\_order` (`order\_id`) ON DELETE RESTRICT ON UPDATE RESTRICT

) ENGINE=InnoDB AUTO\_INCREMENT=122 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci ROW\_FORMAT=DYNAMIC COMMENT='订单项表';

/\*Data for the table `t\_order\_item` \*/

insert into `t\_order\_item`(`id`,`name`,`count`,`price`,`total\_price`,`order\_id`) values (105,'数据结构与算法',19,'78.50','1491.50','16360997619564'),(106,'Hive编程指南',13,'34.50','448.50','16360997619564'),(107,'Scala和Spark大数据分析',67,'66.50','4455.50','16360997619564'),(108,'MySQL从入门到精通',1,'74.80','74.80','16361024784194'),(109,'Java编程思想',45,'99.50','4477.50','16361024784194'),(110,'计算机网络 自顶向下方法',1,'44.50','44.50','16361024784194'),(111,'数据结构与算法分析',1,'173.15','173.15','16362871119234'),(112,'UNIX环境高级编程',11,'118.30','1301.30','16362871119234'),(113,'MySQL从入门到精通',1,'74.80','74.80','16362871119234'),(114,'Java从入门到精通',1,'25.90','25.90','16365994816754'),(115,'数据结构与算法',13,'78.50','1020.50','16365994816754'),(116,'Hive编程指南',1,'34.50','34.50','16365994816754'),(117,'数据结构与算法',1,'78.50','78.50','16365995899624'),(118,'Java从入门到精通',10,'25.90','259.00','163661377901311'),(119,'Scala和Spark大数据分析',20,'66.50','1330.00','163661377901311'),(120,'Java编程思想',30,'99.50','2985.00','163661377901311'),(121,'MySQL从入门到精通',1,'74.80','74.80','16384106695694');

/\*Table structure for table `t\_user` \*/

DROP TABLE IF EXISTS `t\_user`;

CREATE TABLE `t\_user` (

`id` int NOT NULL AUTO\_INCREMENT,

`username` varchar(20) CHARACTER SET utf8mb4 COLLATE utf8mb4\_general\_ci NOT NULL,

`password` varchar(30) CHARACTER SET utf8mb4 COLLATE utf8mb4\_general\_ci NOT NULL,

`email` varchar(200) CHARACTER SET utf8mb4 COLLATE utf8mb4\_general\_ci DEFAULT NULL,

PRIMARY KEY (`id`) USING BTREE,

UNIQUE KEY `username` (`username`) USING BTREE

) ENGINE=InnoDB AUTO\_INCREMENT=12 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci ROW\_FORMAT=DYNAMIC;

/\*Data for the table `t\_user` \*/

insert into `t\_user`(`id`,`username`,`password`,`email`) values (4,'admin','admin','admin@qq.com'),(6,'adam1224','123123','123@123.com'),(7,'as1245','123123','123@112.com'),(8,'12312','123123aa','123123@qq.com'),(9,'sq\_123','sq\_123','123@qq.com'),(10,'云云老婆','123','1234@qq.com'),(11,'xxy1224','xxy1224','850340745@qq.com');

/\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/;

/\*!40014 SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS \*/;

/\*!40014 SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS \*/;

/\*!40111 SET SQL\_NOTES=@OLD\_SQL\_NOTES \*/;

---------------------------------------------------

pom.xml文件：

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>Bookstore</artifactId>

<version>2.0</version>

<name>Bookstore</name>

<packaging>war</packaging>

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<junit.version>5.7.1</junit.version>

<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>

</properties>

<dependencies>

<dependency>

<groupId>com.google.code.gson</groupId>

<artifactId>gson</artifactId>

<version>2.8.6</version>

</dependency>

<dependency>

<groupId>com.github.penggle</groupId>

<artifactId>kaptcha</artifactId>

<version>2.3.2</version>

</dependency>

<dependency>

<groupId>jstl</groupId>

<artifactId>jstl</artifactId>

<version>1.1.2</version>

</dependency>

<dependency>

<groupId>taglibs</groupId>

<artifactId>standard</artifactId>

<version>1.1.2</version>

</dependency>

<dependency>

<groupId>com.google.guava</groupId>

<artifactId>guava</artifactId>

<version>30.0-jre</version>

</dependency>

<dependency>

<groupId>commons-beanutils</groupId>

<artifactId>commons-beanutils</artifactId>

<version>1.9.4</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>4.0.1</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-api</artifactId>

<version>${junit.version}</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-params</artifactId>

<version>${junit.version}</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-engine</artifactId>

<version>${junit.version}</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.25</version>

</dependency>

<dependency>

<groupId>com.alibaba</groupId>

<artifactId>druid</artifactId>

<version>1.2.1</version>

</dependency>

<dependency>

<groupId>commons-dbutils</groupId>

<artifactId>commons-dbutils</artifactId>

<version>1.6</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-war-plugin</artifactId>

<version>3.3.1</version>

</plugin>

</plugins>

</build>

</project>

---------------------------------------------------

BookDao.java文件：

package com.dao;

import com.pojo.Book;

import java.util.List;

public interface BookDao {

int addBook(Book book) ;

int deleteBookById(Integer id) ;

int updateBook(Book book) ;

Book queryBookById(Integer id) ;

List<Book> queryBooks();

int queryForPageTotalCountByPrice() ;

List<Book> queryForItemsByPrice(int begin, int pageSize) ;

int queryForPageTotalCountByPrice(int min, int max) ;

List<Book> queryForItemsByPrice(int min, int max, int begin, int pageSize) ;

int queryForPageTotalCountByName(String name) ;

List<Book> queryForItemsByName(String name, int begin, int pageSize) ;

}

---------------------------------------------------

BaseDao.java文件：

package com.dao.impl;

import com.pojo.Book;

import com.utils.JdbcUtils;

import org.apache.commons.dbutils.\*;

import org.apache.commons.dbutils.handlers.BeanHandler;

import org.apache.commons.dbutils.handlers.BeanListHandler;

import org.apache.commons.dbutils.handlers.ScalarHandler;

import java.sql.Connection;

import java.sql.SQLException;

import java.util.Arrays;

import java.util.List;

public abstract class BaseDao {

//Dbutils的执行器, 类似于Statement

private QueryRunner queryRunner = new QueryRunner();

//行处理器: 将结果集的每一行转换成Bean对象

//(GenerousBeanProcessor的匹配规则: 忽略下划线、不区分大小写, 在此基础上来匹配列名和Bean属性名)

private RowProcessor rowProcessor = new BasicRowProcessor(new GenerousBeanProcessor());

public BaseDao() {

}

public BaseDao(RowProcessor rowProcessor) {

this.rowProcessor = rowProcessor;

}

/\*\*

\* 执行增删改语句, Insert/Delete/Update

\* @return 受影响的行数

\*/

public int update(String sql, Object... params) {

Connection conn = JdbcUtils.getConnection();

try {

return queryRunner.update(conn, sql, params);

} catch (SQLException throwables) {

throw new RuntimeException("BaseDao#update()执行错误, sql: " + sql + "params: " + Arrays.toString(params),

throwables);

}

}

/\*\*

\* 执行查询语句, 返回结果集的第一行

\* @param c 结果集中每行对应的Java类型

\* @return 查询结果的第一行, 查不到返回null

\*/

public <T> T queryForOne(String sql, Class<T> c, Object... params) {

Connection conn = JdbcUtils.getConnection();

try {

return queryRunner.query(conn, sql, new BeanHandler<>(c, rowProcessor), params);

} catch (SQLException throwables) {

throw new RuntimeException("BaseDao#queryForOne()执行错误, sql: " + sql + "params: " + Arrays.toString(params),

throwables);

}

}

/\*\*

\* 执行查询语句, 将查询结果包装成List并返回

\* @param c 结果集中每行对应的Java类型

\* @return 包含查询结果的List

\*/

public <T> List<T> queryForList(String sql, Class<T> c, Object... params) {

Connection conn = JdbcUtils.getConnection();

try {

return queryRunner.query(conn, sql, new BeanListHandler<>(c, rowProcessor), params);

} catch (SQLException throwables) {

throw new RuntimeException("BaseDao#queryForList()执行错误, sql: " + sql + "params: " + Arrays.toString(params),

throwables);

}

}

/\*\*

\* 执行查询语句, 只返回结果集的第一行第一列的值

\* @return 第一行第一列的值, 没有值时返回null

\*/

public Object queryForSingleValue(String sql, Object... params) {

Connection conn = JdbcUtils.getConnection();

try {

return queryRunner.query(conn, sql, new ScalarHandler<>(), params);

} catch (SQLException throwables) {

throw new RuntimeException("BaseDao#queryForSingleValue()执行错误, sql: " + sql + "params: " + Arrays.toString(params),

throwables);

}

}

}

---------------------------------------------------

BookDaoImpl.java文件：

package com.dao.impl;

import com.dao.BookDao;

import com.pojo.Book;

import java.util.List;

public class BookDaoImpl extends BaseDao implements BookDao {

@Override

public int addBook(Book book) {

String sql = "insert into t\_book(`name`,`author`,`price`,`sales`,`stock`,`img\_path`) values(?,?,?,?,?,?)";

return update(sql, book.getName(), book.getAuthor(), book.getPrice(), book.getSales(), book.getStock(), book.getImgPath());

}

@Override

public int deleteBookById(Integer id) {

String sql = "delete from t\_book where `id` = ?";

return update(sql, id);

}

@Override

public int updateBook(Book book) {

String sql = "update t\_book set `name`=?,`author`=?,`price`=?,`sales`=?,`stock`=?,`img\_path`=? where id=?";

return update(sql, book.getName(), book.getAuthor(), book.getPrice(), book.getSales(), book.getStock(), book.getImgPath(), book.getId());

}

@Override

public Book queryBookById(Integer id) {

String sql = "select `id`,`name`,`author`,`price`,`sales`,`stock`,`img\_path` from t\_book where `id`=?";

return queryForOne(sql, Book.class, id);

}

@Override

public List<Book> queryBooks() {

String sql = "select `id`,`name`,`author`,`price`,`sales`,`stock`,`img\_path` from t\_book";

return queryForList(sql, Book.class);

}

/\*\* 查询总记录数(图书总数量) \*/

@Override

public int queryForPageTotalCountByPrice() {

String sql = "select count(\*) from t\_book";

//因为queryForSingleValue()方法返回的是Long类型

//需要先转为String, 通过String才能转为Integer

Object obj = queryForSingleValue(sql);

return Integer.parseInt(String.valueOf(obj));

}

/\*\*

\* 查询指定起始位置, 指定数量下的图书记录

\*

\* @param begin 起始索引位置

\* @param pageSize 要查询的记录数

\* @return 图书集合

\*/

@Override

public List<Book> queryForItemsByPrice(int begin, int pageSize) {

String sql = "select `id`,`name`,`author`,`price`,`sales`,`stock`,`img\_path` from t\_book limit ?,?";

List<Book> books = queryForList(sql, Book.class, begin, pageSize);

return books;

}

/\*\* 查询指定价格区间下的图书总数量 \*/

@Override

public int queryForPageTotalCountByPrice(int min, int max) {

String sql = "select count(\*) from t\_book tb\_book where price between ? and ?";

int pageTotalCount = Integer.parseInt(String.valueOf(queryForSingleValue(sql, min, max)));

return pageTotalCount;

}

/\*\* 查询指定价格区间, 起始索引, 记录数下的图书数据 \*/

@Override

public List<Book> queryForItemsByPrice(int min, int max, int begin, int pageSize) {

String sql = "select `id`,`name`,`author`,`price`,`sales`,`stock`,`img\_path` from t\_book " +

"where price between ? and ? order by price limit ?,?";

List<Book> books = queryForList(sql, Book.class, min, max, begin, pageSize);

return books;

}

/\*\* 查询指定书名下的图书总数量 \*/

@Override

public int queryForPageTotalCountByName(String name) {

String sql = "select count(\*) from t\_book tb\_book where `name` LIKE ?";

int pageTotalCount = Integer.parseInt(String.valueOf(queryForSingleValue(sql, "%" + name + "%")));

return pageTotalCount;

}

@Override

public List<Book> queryForItemsByName(String name, int begin, int pageSize) {

String sql = "select `id`,`name`,`author`,`price`,`sales`,`stock`,`img\_path` from t\_book " +

"where `name` LIKE ? order by price limit ?,?";

List<Book> books = queryForList(sql, Book.class, "%" + name + "%", begin, pageSize);

System.out.println(books);

return books;

}

}

---------------------------------------------------

OrderDaoImpl.java文件：

package com.dao.impl;

import com.dao.OrderDao;

import com.pojo.Order;

import com.pojo.OrderItem;

import java.util.List;

public class OrderDaoImpl extends BaseDao implements OrderDao {

@Override

public int saveOrder(Order order) {

String sql = "insert into t\_order (order\_id,create\_time,price,`status`,user\_id) values(?,?,?,?,?)";

return update(sql, order.getOrderId(), order.getCreateTime(), order.getPrice(), order.getStatus(),

order.getUserId());

}

@Override

public List<Order> queryMyOrders(int userId) {

String sql = "select order\_id,create\_time,price,`status`,user\_id from t\_order where user\_id=?";

return queryForList(sql, Order.class, userId);

}

@Override

public Order queryOrderByOrderId(String orderId) {

String sql = "select order\_id, create\_time, price, status, user\_id from t\_order where order\_id=?";

return queryForOne(sql, Order.class, orderId);

}

@Override

public int changeOrderStatus(String orderId, int status) {

String sql = "update t\_order set `status`=? where order\_id=?";

return update(sql, status, orderId);

}

@Override

public List<OrderItem> queryOrderDetailByOrderId(String orderId) {

String sql = "select id,`name`,count,price,total\_price,order\_id from t\_order\_item where order\_id=?";

return queryForList(sql, OrderItem.class, orderId);

}

@Override

public int queryOrderStatus(String orderId) {

String sql = "select status from t\_order where order\_id = ?";

return (int) queryForSingleValue(sql, orderId);

}

@Override

public int queryTotalCount() {

String sql = "select count(\*) from t\_order;";

System.out.println(queryForSingleValue(sql).getClass().getName());

return Integer.parseInt(String.valueOf(queryForSingleValue(sql)));

}

@Override

public List<Order> queryOrdersByLimit(int begin, int size) {

String sql = "select order\_id, create\_time, price, status, user\_id from t\_order order by create\_time desc " +

"limit ?,?";

return queryForList(sql, Order.class, begin, size);

}

}

---------------------------------------------------

OrderItemDaoImpl.java文件：

package com.dao.impl;

import com.dao.OrderItemDao;

import com.pojo.OrderItem;

import java.util.List;

public class OrderItemDaoImpl extends BaseDao implements OrderItemDao {

@Override

public int saveOrderItem(OrderItem orderItem) {

String sql = "insert into t\_order\_item(id,`name`,count,price,total\_price,order\_id) values(?,?,?,?,?,?)";

return update(sql, orderItem.getId(), orderItem.getName(), orderItem.getCount(), orderItem.getPrice(),

orderItem.getTotalPrice(), orderItem.getOrderId());

}

@Override

public List<OrderItem> queryOrderDetailByOrderId(String orderId) {

String sql = "select id, `name`, count, price, total\_price, order\_id from t\_order\_item where order\_id=?";

return queryForList(sql, OrderItem.class, orderId);

}

}

---------------------------------------------------

UserDaoImpl.java文件：

package com.dao.impl;

import com.dao.UserDao;

import com.pojo.User;

public class UserDaoImpl extends BaseDao implements UserDao {

@Override

public User queryUserByUsername(String username) {

String sql = "select id,username,password,email from t\_user where username = ?";

return queryForOne(sql, User.class, username);

}

@Override

public User queryUserByUsername(String username, String password) {

String sql = "select id,username,password,email from t\_user where username = ? and password = ?";

return queryForOne(sql, User.class, username, password);

}

@Override

public int saveUser(User user) {

String sql = "insert into t\_user(username,password,email) values(?,?,?)";

return update(sql, user.getUsername(), user.getPassword(), user.getEmail());

}

}

---------------------------------------------------

OrderDao.java文件：

package com.dao;

import com.pojo.Order;

import com.pojo.OrderItem;

import java.util.List;

/\*\*

\* @Description 订单DAO类

\*/

public interface OrderDao {

/\*\* 保存订单 \*/

int saveOrder(Order order);

/\*\* 根据用户id查询订单 \*/

List<Order> queryMyOrders(int userId);

/\*\* 根据订单id查询订单 \*/

Order queryOrderByOrderId(String orderId);

/\*\*

\* 修改订单状态

\* @param orderId 要修改的订单的id

\* @param status 要修改的订单状态, 0未发货, 1已发货, 2已收货

\* @return 受影响的行数

\*/

int changeOrderStatus(String orderId, int status);

/\*\*

\* 查询订单详情

\* @param orderId 订单id

\* @return 该订单的所有订单项

\*/

List<OrderItem> queryOrderDetailByOrderId(String orderId);

/\*\* 查询订单状态 \*/

int queryOrderStatus(String orderId);

/\*\* 查询总订单数 \*/

int queryTotalCount();

/\*\* 根据起始索引和个数查找订单 \*/

List<Order> queryOrdersByLimit(int begin, int size);

}

---------------------------------------------------

OrderItemDao.java文件：

package com.dao;

import com.pojo.OrderItem;

import java.util.List;

/\*\*

\* @Description 订单项的DAO类

\*/

public interface OrderItemDao {

/\*\* 保存订单项 \*/

int saveOrderItem(OrderItem orderItem) ;

/\*\* 根据订单编号查询所有订单项 \*/

List<OrderItem> queryOrderDetailByOrderId(String orderId) ;

}

---------------------------------------------------

UserDao.java文件：

package com.dao;

import com.pojo.User;

public interface UserDao {

/\*\*

\* 根据用户名查用户信息

\*

\* @param username 用户名

\* @return 若返回null, 说明该用户不存在

\*/

User queryUserByUsername(String username) ;

/\*\*

\* 根据用户名和密码查用户信息

\*

\* @param username 用户名

\* @param password 密码

\* @return 若返回null, 说明用户名或密码错误

\*/

User queryUserByUsername(String username, String password) ;

/\*\*

\* 保存用户信息

\*

\* @param user 用户信息

\* @return 返回受影响的行数, 若返回-1说明保存失败

\*/

int saveUser(User user) ;

}

---------------------------------------------------

ManagerFilter.java文件：

package com.filter;

import javax.servlet.\*;

import javax.servlet.annotation.WebFilter;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.IOException;

import java.util.Optional;

/\*\*

\* @Description 拦截所有访问后台资源的请求，对其进行权限检查。

\*/

// /pages/manager/\*拦截后台静态资源, /manager/\*拦截后台动态资源

@WebFilter({"/pages/manager/\*", "/manager/\*"})

public class ManagerFilter implements Filter {

@Override

public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws IOException,

ServletException {

HttpServletRequest req = (HttpServletRequest) request;

HttpServletResponse resp = (HttpServletResponse) response;

System.out.println("拦截到请求: " + req.getRequestURI());

Optional<Object> user = Optional.ofNullable(req.getSession(false)).map(session -> session.getAttribute("user"));

if (user.isPresent()) {

System.out.println("放行...");

chain.doFilter(request, response);

} else {

System.out.println("重定向回登录页面...");

resp.sendRedirect(req.getContextPath() + "/pages/user/login.jsp");

}

}

}

---------------------------------------------------

TransactionFilter.java文件：

package com.filter;

import com.utils.JdbcUtils;

import javax.servlet.Filter;

import javax.servlet.FilterChain;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import javax.servlet.annotation.WebFilter;

@WebFilter("/\*")

public class TransactionFilter implements Filter {

@Override

public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) {

try {

chain.doFilter(request, response);

JdbcUtils.commitAndClose(); //提交事务

} catch (Exception e) {

JdbcUtils.rollbackAndClose(); //回滚事务

throw new RuntimeException(e.getMessage(), e); //500错误

}

}

}

---------------------------------------------------

Book.java文件：

package com.pojo;

import java.io.Serializable;

public class Book implements Serializable {

private static final long serialVersionUID = 3792488105248258110L;

private Integer id;

private String name;

private String author;

private Double price;

private Integer sales;

private Integer stock;

private String imgPath = "static/img/default.jpg";

public Book() {

}

public Book(Integer id, String name, String author, Double price, Integer sales, Integer stock, String imgPath) {

this.id = id;

this.name = name;

this.author = author;

this.price = price;

this.sales = sales;

this.stock = stock;

this.imgPath = imgPath;

//imgPath必须有值

if (imgPath != null && !imgPath.isEmpty()) {

this.imgPath = imgPath;

}

}

@Override

public String toString() {

return "Book{" +

"id=" + id +

", name='" + name + '\'' +

", author='" + author + '\'' +

", price=" + price +

", sales=" + sales +

", stock=" + stock +

", imgPath='" + imgPath + '\'' +

'}';

}

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getAuthor() {

return author;

}

public void setAuthor(String author) {

this.author = author;

}

public Double getPrice() {

return price;

}

public void setPrice(Double price) {

this.price = price;

}

public Integer getSales() {

return sales;

}

public void setSales(Integer sales) {

this.sales = sales;

}

public Integer getStock() {

return stock;

}

public void setStock(Integer stock) {

this.stock = stock;

}

public String getImgPath() {

return imgPath;

}

public void setImgPath(String imgPath) {

//imgPath必须有值

if (imgPath != null && !imgPath.isEmpty()) {

this.imgPath = imgPath;

}

}

}

---------------------------------------------------

Cart.java文件：

package com.pojo;

import java.io.Serializable;

import java.math.BigDecimal;

import java.util.HashMap;

import java.util.Map;

/\*\*

\* @Description 购物车模型

\*/

public class Cart implements Serializable {

private static final long serialVersionUID = -2243771653523724286L;

private Integer totalCount = 0; //总商品个数

private BigDecimal totalPrice = new BigDecimal(0); //总商品价格

private Map<Integer, CartItem> items = new HashMap<>(); //购物车中的商品, key是商品编号

/\*\* 添加商品项 \*/

public void addItem(CartItem cartItem) {

Integer id = cartItem.getId();

CartItem item = items.get(id);

if (item != null) { //商品已存在

item.setCount(item.getCount() + cartItem.getCount());

} else { //商品不存在

items.put(cartItem.getId(), cartItem);

}

totalCount += cartItem.getCount();

totalPrice = totalPrice.add(cartItem.getTotalPrice());

}

/\*\* 修改指定商品数量 \*/

public void updateCount(Integer id, Integer count) {

CartItem item = items.get(id);

int increment = count - item.getCount();

this.totalCount = this.totalCount + increment;

this.totalPrice = this.totalPrice.add(item.getPrice().multiply(new BigDecimal(increment)));

item.setCount(count);

}

/\*\* 删除商品项 \*/

public void delete(Integer id) {

CartItem item = items.get(id);

totalCount = totalCount - item.getCount();

totalPrice = totalPrice.subtract(item.getTotalPrice());

items.remove(id);

}

/\*\* 清空购物车 \*/

public void clear() {

totalCount = 0;

totalPrice = new BigDecimal(0);

items.clear();

}

public Integer getTotalCount() {

return totalCount;

}

public BigDecimal getTotalPrice() {

return totalPrice;

}

public Map<Integer, CartItem> getItems() {

return items;

}

@Override

public String toString() {

return "Cart{" +

"totalCount=" + totalCount +

", totalPrice=" + totalPrice +

", items=" + items +

'}';

}

}

---------------------------------------------------

CartItem.java文件：

package com.pojo;

import java.io.Serializable;

import java.math.BigDecimal;

/\*\*

\* @Description 购物车的商品项

\*/

public class CartItem implements Serializable {

private static final long serialVersionUID = 8383012871260829697L;

private Integer id; //商品编号

private String name; //商品名称

private Integer count; //商品数量

private BigDecimal price; //商品单价

private BigDecimal totalPrice; //商品总价

public CartItem() {

}

public CartItem(Integer id, String name, Integer count, BigDecimal price) {

this.id = id;

this.name = name;

this.count = count;

this.price = price;

updateTotalPrice();

}

private void updateTotalPrice() {

if (this.price != null && this.count != null) { //避免空指针异常

this.totalPrice = this.price.multiply(BigDecimal.valueOf(this.count));

}

}

public Integer getCount() {

return count;

}

public void setCount(Integer count) {

this.count = count;

updateTotalPrice();

}

public BigDecimal getPrice() {

return price;

}

public void setPrice(BigDecimal price) {

this.price = price;

updateTotalPrice();

}

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public BigDecimal getTotalPrice() {

return totalPrice;

}

@Override

public String toString() {

return "CartItem{" +

"id=" + id +

", name='" + name + '\'' +

", count=" + count +

", price=" + price +

", totalPrice=" + totalPrice +

'}';

}

}

---------------------------------------------------

Order.java文件：

package com.pojo;

import java.io.Serializable;

import java.math.BigDecimal;

import java.sql.Timestamp;

/\*\*

\* @Description 订单模型, 只存储订单的元数据, 订单项目

\*/

public class Order implements Serializable {

private static final long serialVersionUID = 6264111272179828920L;

private String orderId; //订单编号

private Timestamp createTime; //创建时间

private BigDecimal price; //订单价格

private Integer status = 0; //订单状态, 0未发货, 1已发货, 2已收货

private Integer userId; //所属的用户id

public Order() {

}

public Order(String orderId, Timestamp createTime, BigDecimal price, Integer status, Integer userId) {

this.orderId = orderId;

this.createTime = createTime;

this.price = price;

this.status = status;

this.userId = userId;

}

@Override

public String toString() {

return "Order{" +

"orderId='" + orderId + '\'' +

", createTime=" + createTime +

", price=" + price +

", status=" + status +

", userId=" + userId +

'}';

}

public String getOrderId() {

return orderId;

}

public void setOrderId(String orderId) {

this.orderId = orderId;

}

public Timestamp getCreateTime() {

return createTime;

}

public void setCreateTime(Timestamp createTime) {

this.createTime = createTime;

}

public BigDecimal getPrice() {

return price;

}

public void setPrice(BigDecimal price) {

this.price = price;

}

public Integer getStatus() {

return status;

}

public void setStatus(Integer status) {

this.status = status;

}

public Integer getUserId() {

return userId;

}

public void setUserId(Integer userId) {

this.userId = userId;

}

}

---------------------------------------------------

OrderItem.java文件：

package com.pojo;

import java.io.Serializable;

import java.math.BigDecimal;

/\*\*

\* @Description 订单项模型

\*/

public class OrderItem implements Serializable {

private static final long serialVersionUID = 9170294652039349360L;

private Integer id; //订单项id(主键)

private String name; //商品名称

private Integer count; //商品个数

private BigDecimal price; //商品单价

private BigDecimal totalPrice; //商品总价

private String orderId; //所属的订单编号

public OrderItem() {

updateTotalPrice();

}

public OrderItem(Integer id, String name, Integer count, BigDecimal price, String orderId) {

this.id = id;

this.name = name;

this.count = count;

this.price = price;

this.orderId = orderId;

updateTotalPrice();

}

@Override

public String toString() {

return "OrderItem{" +

"id=" + id +

", name='" + name + '\'' +

", count=" + count +

", price=" + price +

", totalPrice=" + totalPrice +

", orderId='" + orderId + '\'' +

'}';

}

public void updateTotalPrice() {

if (price != null && count != null) { //避免空指针异常

totalPrice = price.multiply(BigDecimal.valueOf(count));

}

}

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Integer getCount() {

return count;

}

public void setCount(Integer count) {

this.count = count;

updateTotalPrice();

}

public BigDecimal getPrice() {

return price;

}

public void setPrice(BigDecimal price) {

this.price = price;

updateTotalPrice();

}

public BigDecimal getTotalPrice() {

return totalPrice;

}

public void setTotalPrice(BigDecimal totalPrice) {

this.totalPrice = totalPrice;

}

public String getOrderId() {

return orderId;

}

public void setOrderId(String orderId) {

this.orderId = orderId;

}

}

---------------------------------------------------

Page.java文件：

package com.pojo;

import java.io.Serializable;

import java.util.List;

/\*\*

\* @Description Page是分页的对象模型, 存储一页数据

\*/

public class Page<T> implements Serializable {

public static final Integer PAGE\_SIZE = 8; //默认页大小

public static final Integer MAX\_PAGE\_SIZE = 30; //最大页大小

private static final long serialVersionUID = 2916853673773967720L;

private Integer pageTotal; //总页数

private Integer pageTotalCount; //总记录数

private Integer pageNo; //当前页码

private Integer pageSize = PAGE\_SIZE; //当前页的记录数

private List<T> items; //当前页的数据

private String url; //分页栏的请求地址 (相对于工程路径)

@Override

public String toString() {

return "Page{" +

"pageTotal=" + pageTotal +

", pageTotalCount=" + pageTotalCount +

", pageNo=" + pageNo +

", pageSize=" + pageSize +

", items=" + items +

", url='" + url + '\'' +

'}';

}

public Integer getPageTotal() {

return pageTotal;

}

public void setPageTotal(Integer pageTotal) {

this.pageTotal = pageTotal;

}

public Integer getPageTotalCount() {

return pageTotalCount;

}

public void setPageTotalCount(Integer pageTotalCount) {

this.pageTotalCount = pageTotalCount;

}

public Integer getPageNo() {

return pageNo;

}

public void setPageNo(Integer pageNo) {

if (pageTotal == null) {

throw new RuntimeException("请先设置pageTotal");

}

//必须先设置pageTotal, 再设置pageNo, 避免空指针异常

if (pageNo > 0 && pageNo <= pageTotal) {

this.pageNo = pageNo;

} else {

this.pageNo = 1;

}

}

public Integer getPageSize() {

return pageSize;

}

public void setPageSize(Integer pageSize) {

if (pageSize > 0 && pageSize <= MAX\_PAGE\_SIZE) {

this.pageSize = pageSize;

} else {

this.pageSize = PAGE\_SIZE;

}

}

public List<T> getItems() {

return items;

}

public void setItems(List<T> items) {

this.items = items;

}

public String getUrl() {

return url;

}

public void setUrl(String url) {

this.url = url;

}

}

---------------------------------------------------

User.java文件：

package com.pojo;

import java.io.Serializable;

public class User implements Serializable {

private static final long serialVersionUID = -8448661564950880315L;

private Integer id;

private String username;

private String password;

private String email;

public User() {

}

public User(Integer id, String username, String password, String email) {

this.id = id;

this.username = username;

this.password = password;

this.email = email;

}

@Override

public String toString() {

return "User{" +

"id=" + id +

", username='" + username + '\'' +

", password='" + password + '\'' +

", email='" + email + '\'' +

'}';

}

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

}

---------------------------------------------------

BookService.java文件：

package com.service;

import com.pojo.Book;

import com.pojo.Page;

import java.util.List;

public interface BookService {

boolean addBook(Book book);

boolean deleteBookById(Integer id);

boolean updateBook(Book book);

Book queryBookById(Integer id);

List<Book> queryBooks();

Page<Book> page(int pageNo, int pageSize);

Page<Book> pageByPrice(int min, int max, int pageNo, int pageSize);

Page<Book> pageByName(String bookName, int pageNo, int pageSize);

}

---------------------------------------------------

BookServiceImpl.java文件：

package com.service.impl;

import com.dao.BookDao;

import com.dao.impl.BookDaoImpl;

import com.pojo.Book;

import com.pojo.Page;

import com.service.BookService;

import java.util.List;

public class BookServiceImpl implements BookService {

private final BookDao bookDao = new BookDaoImpl();

@Override

public boolean addBook(Book book) {

return bookDao.addBook(book) != 0;

}

@Override

public boolean deleteBookById(Integer id) {

return bookDao.deleteBookById(id) != 0;

}

@Override

public boolean updateBook(Book book) {

return bookDao.updateBook(book) != 0;

}

@Override

public Book queryBookById(Integer id) {

return bookDao.queryBookById(id);

}

@Override

public List<Book> queryBooks() {

return bookDao.queryBooks();

}

/\*\* 根据pageNo和pageSize创建Page对象 \*/

@Override

public Page<Book> page(int pageNo, int pageSize) {

Page<Book> page = new Page<>();

//1.设置总记录数

int pageTotalCount = bookDao.queryForPageTotalCountByPrice();

page.setPageTotalCount(pageTotalCount);

//2.设置页大小

page.setPageSize(pageSize);

//3.设置总页数

int pageTotal = (int) Math.ceil(pageTotalCount \* 1.0 / page.getPageSize());

page.setPageTotal(pageTotal);

//3.设置页码

page.setPageNo(pageNo);

//4.求当前页数据

int begin = (page.getPageNo() - 1) \* page.getPageSize(); //当前页数据的起始索引

List<Book> books = bookDao.queryForItemsByPrice(begin, page.getPageSize());

page.setItems(books);

return page;

}

/\*\*

\* 搜索指定价格区间, 页码, 页大小下的图书

\*

\* @param min 最低价格

\* @param max 最高价格

\* @param pageNo 页码

\* @param pageSize 页大小

\* @return 返回包含图书数据和分页信息的Page对象

\*/

@Override

public Page<Book> pageByPrice(int min, int max, int pageNo, int pageSize) {

Page<Book> page = new Page<>();

//设置总记录数

int pageTotalCount = bookDao.queryForPageTotalCountByPrice(min, max);

page.setPageTotalCount(pageTotalCount);

//设置页大小

page.setPageSize(pageSize);

//设置总页数

int pageTotal = (int) Math.ceil(pageTotalCount \* 1.0 / page.getPageSize());

page.setPageTotal(pageTotal);

//设置页码

page.setPageNo(pageNo);

//设置图书数据

int begin = (page.getPageNo() - 1) \* page.getPageSize();

List<Book> items = bookDao.queryForItemsByPrice(min, max, begin, page.getPageSize());

page.setItems(items);

return page;

}

/\*\*

\* 搜索指定书名, 页码, 页大小下的图书

\*

\* @param bookName 书名

\* @param pageNo 页码

\* @param pageSize 页大小

\* @return

\*/

@Override

public Page<Book> pageByName(String bookName, int pageNo, int pageSize) {

Page<Book> page = new Page<>();

//设置总记录数

int pageTotalCount = bookDao.queryForPageTotalCountByName(bookName);

// queryForPageTotalCountByPrice(min, max);

page.setPageTotalCount(pageTotalCount);

//设置页大小

page.setPageSize(pageSize);

//设置总页数

int pageTotal = (int) Math.ceil(pageTotalCount \* 1.0 / page.getPageSize());

page.setPageTotal(pageTotal);

//设置页码

page.setPageNo(pageNo);

//设置图书数据

int begin = (page.getPageNo() - 1) \* page.getPageSize();

List<Book> items = bookDao.queryForItemsByName(bookName, begin, page.getPageSize());

// queryForItemsByPrice(min, max, begin, page.getPageSize());

page.setItems(items);

return page;

}

}

---------------------------------------------------

OrderServiceImpl.java文件：

package com.service.impl;

import com.dao.BookDao;

import com.dao.OrderDao;

import com.dao.OrderItemDao;

import com.dao.impl.BookDaoImpl;

import com.dao.impl.OrderDaoImpl;

import com.dao.impl.OrderItemDaoImpl;

import com.pojo.\*;

import com.service.OrderService;

import java.sql.Timestamp;

import java.util.List;

public class OrderServiceImpl implements OrderService {

private OrderDao orderDao = new OrderDaoImpl();

private OrderItemDao orderItemDao = new OrderItemDaoImpl();

private BookDao bookDao = new BookDaoImpl();

@Override

public String createOrder(Cart cart, int userId) {

//创建订单号

String orderId = "" + System.currentTimeMillis() + userId;

//保存订单

Order order = new Order(orderId, new Timestamp(System.currentTimeMillis()), cart.getTotalPrice(), 0, userId);

orderDao.saveOrder(order);

cart.getItems().values().forEach(cartItem -> { //获取购物车中每一个商品项

//转换成订单项

OrderItem orderItem = new OrderItem(null, cartItem.getName(), cartItem.getCount(), cartItem.getPrice(),

orderId);

//保存订单项

orderItemDao.saveOrderItem(orderItem);

//更新图书销量和库存

Book book = bookDao.queryBookById(cartItem.getId());

book.setSales(book.getSales() + cartItem.getCount());

book.setStock(book.getStock() - cartItem.getCount());

int rows = bookDao.updateBook(book);

if (rows == 0) {

throw new RuntimeException("更新图书销量和库存失败...");

}

});

//清空购物车

cart.clear();

return orderId;

}

@Override

public List<OrderItem> orderDetail(String orderId) {

return orderDao.queryOrderDetailByOrderId(orderId);

}

@Override

public Order queryOrderByOrderId(String orderId) {

return orderDao.queryOrderByOrderId(orderId);

}

@Override

public List<Order> queryMyOrders(int userId) {

return orderDao.queryMyOrders(userId);

}

@Override

public void sendOrder(String orderId) {

orderDao.changeOrderStatus(orderId, 1);

}

@Override

public void receiveOrder(String orderId, Integer userId) {

//检查订单状态 (已发货状态才能收货)

int status = orderDao.queryOrderStatus(orderId);

if (status == 1) {

//确认收货

orderDao.changeOrderStatus(orderId, 2);

}

}

@Override

public Page<Order> page(int pageNo, int pageSize) {

Page<Order> page = new Page<>();

//1.设置总记录数

int pageTotalCount = orderDao.queryTotalCount();

page.setPageTotalCount(pageTotalCount);

//2.设置页大小

page.setPageSize(pageSize);

//3.设置总页数

int pageTotal = (int) Math.ceil(pageTotalCount \* 1.0 / pageSize);

page.setPageTotal(pageTotal);

//3.设置页码

page.setPageNo(pageNo);

//4.求当前页数据

int begin = (page.getPageNo() - 1) \* page.getPageSize(); //起始索引

List<Order> orders = orderDao.queryOrdersByLimit(begin, page.getPageSize());

page.setItems(orders);

return page;

}

}

---------------------------------------------------

UserServiceImpl.java文件：

package com.service.impl;

import com.dao.UserDao;

import com.dao.impl.UserDaoImpl;

import com.pojo.User;

import com.service.UserService;

public class UserServiceImpl implements UserService {

private final UserDao userDao = new UserDaoImpl();

@Override

public User login(User user) {

return userDao.queryUserByUsername(user.getUsername(), user.getPassword());

}

@Override

public boolean register(User user) {

if (existsUsername(user.getUsername())) {

return false;

}

int rows = userDao.saveUser(user);

return rows != -1;

}

@Override

public boolean existsUsername(String username) {

return userDao.queryUserByUsername(username) != null;

}

}

---------------------------------------------------

OrderService.java文件：

package com.service;

import com.pojo.Cart;

import com.pojo.Order;

import com.pojo.OrderItem;

import com.pojo.Page;

import java.util.List;

public interface OrderService {

/\*\*

\* 根据购物车商品为用户创建订单

\* @param cart 用于创建订单的购物车

\* @param userId 创建这个订单的用户id

\* @return 创建的订单号, 创建失败返回空串

\*/

String createOrder(Cart cart, int userId);

/\*\*

\* 查看订单详情

\* @param orderId 订单id

\* @return 包含订单项的List

\*/

List<OrderItem> orderDetail(String orderId);

/\*\*

\* 根据订单id查询订单

\* @param orderId 订单id

\* @return 订单对象

\*/

Order queryOrderByOrderId(String orderId);

/\*\*

\* 查询指定用户的所有订单

\* @param userId 用户id

\* @return 订单

\*/

List<Order> queryMyOrders(int userId);

/\*\*

\* 订单发货

\* @param orderId 要发货的订单编号

\*/

void sendOrder(String orderId);

/\*\*

\* 订单收货

\* @param orderId 要收货的订单编号

\* @param user

\*/

void receiveOrder(String orderId, Integer user);

/\*\* 分页查看指定页码的订单 \*/

Page<Order> page(int pageNo, int pageSize);

}

---------------------------------------------------

UserService.java文件：

package com.service;

import com.pojo.User;

import java.util.Optional;

public interface UserService {

/\*\*

\* 用户登录

\*

\* @param user 用户

\* @return 返回登录后的用户信息

\*/

User login(User user);

/\*\*

\* 注册用户

\*

\* @param user 用户

\* @return 返回注册是否成功

\*/

boolean register(User user);

/\*\*

\* 检查用户名是否已存在

\*

\* @param username 用户名

\* @return 返回true表示已存在, 返回false表示不存在

\*/

boolean existsUsername(String username);

}

---------------------------------------------------

AdminLoginServlet.java文件：

package com.servlet;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.IOException;

/\*\*

\* @author DamianAdam

\* @create 2021-11-11-14:02.

\*/

public class AdminLoginServlet extends HttpServlet {

@Override

protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

// 重定向到管理员登录页面。

resp.sendRedirect("pages/manager/manager\_login.jsp");

}

}

---------------------------------------------------

BaseServlet.java文件：

package com.servlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.UnsupportedEncodingException;

import java.lang.reflect.InvocationTargetException;

import java.lang.reflect.Method;

public abstract class BaseServlet extends HttpServlet {

public static final String ENCODING = "utf-8"; //HTTP请求和响应的编码

public static final String METHOD\_PARAM = "action"; //一个请求参数的名字, 这个请求参数指明了要调用的目标方法

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response) {

String methodName = null;

try {

request.setCharacterEncoding(ENCODING);

response.setCharacterEncoding(ENCODING);

methodName = request.getParameter(METHOD\_PARAM);

Method method = this.getClass().getDeclaredMethod(methodName, HttpServletRequest.class,

HttpServletResponse.class);

method.invoke(this, request, response);

} catch (NoSuchMethodException e) {

throw new RuntimeException("请求参数错误" + METHOD\_PARAM + "=" + methodName, e);

} catch (IllegalAccessException e) {

throw new RuntimeException("没有方法访问权限" + METHOD\_PARAM + "=" + methodName, e);

} catch (InvocationTargetException e) {

throw new RuntimeException("反射调用方法错误" + METHOD\_PARAM + "=" + methodName, e);

} catch (UnsupportedEncodingException e) {

throw new RuntimeException("不支持的编码格式" + ENCODING, e);

}

}

@Override

protected void doGet(HttpServletRequest req, HttpServletResponse resp) {

this.doPost(req, resp);

}

}

---------------------------------------------------

CartServlet.java文件：

package com.servlet;

import com.google.gson.Gson;

import com.google.gson.JsonObject;

import com.google.gson.JsonPrimitive;

import com.pojo.Book;

import com.pojo.Cart;

import com.pojo.CartItem;

import com.service.BookService;

import com.service.impl.BookServiceImpl;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import java.io.IOException;

import java.math.BigDecimal;

import java.util.HashMap;

import java.util.Map;

/\*\*

\* @Description 处理购物车相关的请求

\*/

@WebServlet("/CartServlet")

public class CartServlet extends BaseServlet {

private BookService bookService = new BookServiceImpl();

/\*\* 处理增加商品项的请求, \*/

protected void ajaxAddItem(HttpServletRequest req, HttpServletResponse resp) throws IOException {

HttpSession session = req.getSession(false);

int bookId = Integer.parseInt(req.getParameter("bookId")); //获取图书编号

Cart cart = (Cart) session.getAttribute("cart");

//第一次增加商品项时需要在Session中创建购物车

if (cart == null) {

cart = new Cart();

session.setAttribute("cart", cart);

}

//获取图书信息

Book book = bookService.queryBookById(bookId);

//将图书信息转换成商品项

CartItem cartItem = new CartItem(bookId, book.getName(), 1, BigDecimal.valueOf(book.getPrice()));

//图书加入购物车

cart.addItem(cartItem);

session.setAttribute("lastItemName", book.getName());

//将购物车商品数量和图书名称以Json形式返回

resp.setContentType("application/json");

JsonObject jsonObject = new JsonObject();

jsonObject.add("totalCount", new JsonPrimitive(cart.getTotalCount()));

jsonObject.add("lastItemName", new JsonPrimitive(book.getName()));

new Gson().toJson(jsonObject, resp.getWriter());

}

/\*\* 处理删除商品项的请求 \*/

protected void deleteItem(HttpServletRequest req, HttpServletResponse resp) throws IOException {

HttpSession session = req.getSession(false);

String userKey = "user";

if (session != null && session.getAttribute(userKey) != null) {

int itemId = Integer.parseInt(req.getParameter("itemId"));

Cart cart = (Cart) session.getAttribute("cart");

Map<Integer, CartItem> items = cart.getItems();

items.remove(itemId);

}

resp.sendRedirect(req.getHeader("Referer"));

}

/\*\* 处理清空购物车的请求 \*/

protected void clear(HttpServletRequest req, HttpServletResponse resp) throws IOException {

HttpSession session = req.getSession(false);

String userKey = "user";

if (session != null && session.getAttribute(userKey) != null) {

Cart cart = (Cart) session.getAttribute("cart");

cart.clear();

}

resp.sendRedirect(req.getHeader("Referer"));

}

/\*\* 处理修改商品数量的请求 \*/

protected void updateCount(HttpServletRequest req, HttpServletResponse resp) throws IOException {

HttpSession session = req.getSession(false);

String userKey = "user";

if (session != null && session.getAttribute(userKey) != null) {

int itemId = Integer.parseInt(req.getParameter("itemId"));

int count = Integer.parseInt(req.getParameter("count"));

Cart cart = (Cart) session.getAttribute("cart");

cart.updateCount(itemId, count); //修改商品数量

}

resp.sendRedirect(req.getHeader("Referer"));

}

}

---------------------------------------------------

ClientBookServlet.java文件：

package com.servlet;

import com.pojo.Book;

import com.pojo.Page;

import com.service.BookService;

import com.service.impl.BookServiceImpl;

import com.utils.WebUtils;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.IOException;

/\*\*

\* @Description 处理主页上的图书相关请求

\*/

@WebServlet("/client/BookServlet")

public class ClientBookServlet extends BaseServlet {

private final BookService bookService = new BookServiceImpl();

/\*\* 处理前台的分页请求, 获得Page对象, 然后转发到 /pages/client/index.jsp \*/

public void page(HttpServletRequest req, HttpServletResponse resp) throws IOException, ServletException {

//1.获取请求参数pageNo和pageSize

int pageNo = WebUtils.parseInt(req.getParameter("pageNo"), 1);

int pageSize = WebUtils.parseInt(req.getParameter("pageSize"), Page.PAGE\_SIZE);

//2.调用bookService.page(pageNo,pageSize), 获得Page对象

Page<Book> page = bookService.page(pageNo, pageSize);

page.setUrl("client/BookServlet?action=page"); //设置分页栏的请求地址

//3.将Page对象保存到request域

req.setAttribute("page", page);

//4.将请求转发到页面 /pages/client/index.jsp

req.getRequestDispatcher("/pages/client/index.jsp").forward(req, resp);

}

/\*\* 处理【价格区间搜索】的分页请求，获得Page对象, 然后转发到 /pages/client/index.jsp \*/

public void pageByPrice(HttpServletRequest req, HttpServletResponse resp) throws IOException, ServletException {

//1.取出请求参数

int pageNo = WebUtils.parseInt(req.getParameter("pageNo"), 1);

int pageSize = WebUtils.parseInt(req.getParameter("pageSize"), Page.PAGE\_SIZE);

int min = WebUtils.parseInt(req.getParameter("min"), 0);

int max = WebUtils.parseInt(req.getParameter("max"), Integer.MAX\_VALUE);

//2.搜索价格区间内的图书, 获得Page对象

Page<Book> page = bookService.pageByPrice(min, max, pageNo, pageSize);

//设置分页栏所需的url, 别忘了附加min和max参数, 否则切页时就变成全局搜索了

page.setUrl("client/BookServlet?action=pageByPrice&min=" + min + "&max=" + max);

//3.将Page对象保存到request域中

req.setAttribute("page", page);

//4.转发到/pages/client/index.jsp

req.getRequestDispatcher("/pages/client/index.jsp").forward(req, resp);

}

}

---------------------------------------------------

ManagerBookServlet.java文件：

package com.servlet;

import com.pojo.Book;

import com.pojo.Page;

import com.service.BookService;

import com.service.impl.BookServiceImpl;

import com.utils.WebUtils;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.IOException;

import java.util.List;

/\*\*

\* @Description 处理后台的图书相关请求

\*/

@WebServlet("/manager/BookServlet")

public class ManagerBookServlet extends BaseServlet {

private final BookService bookService = new BookServiceImpl();

/\*\* 处理分页请求, 获得Page对象, 然后转发到 /pages/manager/book\_manager.jsp \*/

public void page(HttpServletRequest req, HttpServletResponse resp) throws IOException, ServletException {

//1.获取请求参数pageNo和pageSize

int pageNo = WebUtils.parseInt(req.getParameter("pageNo"), 1);

int pageSize = WebUtils.parseInt(req.getParameter("pageSize"), Page.PAGE\_SIZE);

//2.调用bookService.page(pageNo,pageSize), 获得Page对象

Page<Book> page = bookService.page(pageNo, pageSize);

page.setUrl("manager/BookServlet?action=page"); //设置分页栏的请求地址

//3.将Page对象保存到request域

req.setAttribute("page", page);

//4.将请求转发到页面 /pages/manager/book\_manager.jsp

req.getRequestDispatcher("/pages/manager/book\_manager.jsp").forward(req, resp);

}

/\*\*

\* 查询所有图书, 然后转发到图书管理页面 /pages/manager/book\_manager.jsp

\*/

public void list(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

//1.查询全部图书

List<Book> books = bookService.queryBooks();

System.out.println(books);

//2.保存到Request域中

req.setAttribute("books", books);

//3.将请求转发到/pages/manager/book\_manager.jsp

req.getRequestDispatcher("/pages/manager/book\_manager.jsp").forward(req, resp);

}

/\*\* 添加图书, 然后重定向到/book/manager/BookServlet?action=page \*/

public void add(HttpServletRequest req, HttpServletResponse resp) throws IOException {

req.setCharacterEncoding("utf-8");

//1.获取请求参数, 封装成Book对象

Book book = WebUtils.copyParamToBean(req.getParameterMap(), new Book());

//2.添加图书

bookService.addBook(book);

//3.重定向到/book/manager/BookServlet?action=page?pageNo=尾页, 展示尾页图书

int pageNo = Integer.parseInt(req.getParameter("pageNo")) + 1;

resp.sendRedirect(req.getContextPath() + "/manager/BookServlet?action=page&" + "pageNo=" + pageNo);

//不能使用下面的请求转发

//req.getRequestDispatcher("/manager/BookServlet?action=page").forward(req, resp);

}

/\*\* 根据id删除图书, 然后重定向到原书所在页 \*/

public void delete(HttpServletRequest req, HttpServletResponse resp) throws IOException {

//1.获取要删除的图书id

Integer id = Integer.valueOf(req.getParameter("id"));

//2.删除图书

bookService.deleteBookById(id);

//3.重定向到 /book/pages/manager/BookServlet?action=page&pageNo=, 展示原页码下的图书

resp.sendRedirect(req.getContextPath() + "/manager/BookServlet?action=page&pageNo=" + req.getParameter("pageNo"));

}

/\*\* 修改图书的第一步, 根据id获取图书信息, 然后转发到 /pages/manager/book\_edit.jsp \*/

public void getBook(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

//1.获取要修改的图书id

Integer id = Integer.valueOf(req.getParameter("id"));

//2.查询图书, 并保存到request域

Book book = bookService.queryBookById(id);

req.setAttribute("book", book);

//3.转发到 /pages/manager/book\_edit.jsp

req.getRequestDispatcher("/pages/manager/book\_edit.jsp").forward(req, resp);

}

/\*\* 修改图书的第二步, 获取图书信息, 修改图书, 然后重定向到 /book/manager/BookServlet?action=page&pageNo=原书页码 \*/

public void update(HttpServletRequest req, HttpServletResponse resp) throws IOException {

//1.获取请求参数, 封装成Book对象

Book book = WebUtils.copyParamToBean(req.getParameterMap(), new Book());

//2.修改图书

bookService.updateBook(book);

//3.重定向到 /book/manager/BookServlet?action=page&pageNo=原书页码

resp.sendRedirect(req.getContextPath() + "/manager/BookServlet?action=page&pageNo=" + req.getParameter("pageNo"));

}

}

---------------------------------------------------

ManagerOrderServlet.java文件：

package com.servlet;

import com.pojo.Order;

import com.pojo.Page;

import com.service.OrderService;

import com.service.impl.OrderServiceImpl;

import com.utils.WebUtils;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.IOException;

/\*\*

\* @Description 处理订单管理相关请求的Servlet

\*/

@WebServlet("/manager/OrderServlet")

public class ManagerOrderServlet extends BaseServlet {

private OrderService orderService = new OrderServiceImpl();

/\*\* 订单发货 \*/

protected void sendOrder(HttpServletRequest req, HttpServletResponse resp) throws IOException {

//获取订单编号

String orderId = req.getParameter("orderId");

//进行发货

orderService.sendOrder(orderId);

//获取页码和页大小

int pageNo = WebUtils.parseInt(req.getParameter("pageNo"), 1);

int pageSize = WebUtils.parseInt(req.getParameter("pageSize"), Page.PAGE\_SIZE);

//重定向到订单管理页面

resp.sendRedirect(req.getContextPath() + "/manager/OrderServlet?action=page&pageNo=" + pageNo + "&pageSize=" + pageSize);

}

/\*\* 分页查看所有订单 \*/

protected void page(HttpServletRequest req, HttpServletResponse resp) throws IOException, ServletException {

//获取请求参数pageNo和pageSize

int pageNo = WebUtils.parseInt(req.getParameter("pageNo"), 1);

int pageSize = WebUtils.parseInt(req.getParameter("pageSize"), Page.PAGE\_SIZE);

//查询指定页码下的订单

Page<Order> page = orderService.page(pageNo, pageSize);

page.setUrl("manager/OrderServlet?action=page");

//保存到request域

req.setAttribute("page", page);

//转发到订单管理页面

req.getRequestDispatcher("/pages/manager/order\_manager.jsp").forward(req, resp);

}

}

---------------------------------------------------

OrderServlet.java文件：

package com.servlet;

import com.pojo.Cart;

import com.pojo.Order;

import com.pojo.OrderItem;

import com.pojo.User;

import com.service.OrderService;

import com.service.impl.OrderServiceImpl;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import java.io.IOException;

import java.util.List;

import java.util.Objects;

/\*\*

\* @Description 处理订单相关请求

\*/

@WebServlet("/OrderServlet")

public class OrderServlet extends BaseServlet {

private OrderService orderService = new OrderServiceImpl();

/\*\* 处理结账请求 \*/

protected void checkout(HttpServletRequest req, HttpServletResponse resp) throws IOException {

HttpSession session = req.getSession(false);

if (session != null) {

//取出购物车和用户信息

Cart cart = Objects.requireNonNull((Cart) session.getAttribute("cart"), "购物车为空!");

User user = Objects.requireNonNull((User) session.getAttribute("user"), "用户未登录!");

//创建订单

String orderId = orderService.createOrder(cart, user.getId());

//订单号添加进Session中

session.setAttribute("orderId", orderId);

//重定向到结账成功页面

resp.sendRedirect(req.getContextPath() + "/pages/cart/checkout.jsp");

} else {

resp.sendRedirect(req.getContextPath() + "/user/login.jsp");

}

}

/\*\* 查看我的订单列表 \*/

protected void myOrders(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

HttpSession session = req.getSession(false);

//取出用户信息

User user = (User) session.getAttribute("user");

//根据用户id查询所有订单

List<Order> orders = orderService.queryMyOrders(user.getId());

//将订单信息存入Request域

req.setAttribute("orders", orders);

//转发到页面列表页面: /pages/order/order.jsp

req.getRequestDispatcher("/pages/order/order.jsp").forward(req, resp);

}

/\*\* 查看订单详情 \*/

protected void orderDetail(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

//获取订单号

String orderId = req.getParameter("orderId");

//查询订单

Order order = orderService.queryOrderByOrderId(orderId);

//查询订单的所有订单项

List<OrderItem> orderItems = orderService.orderDetail(orderId);

//将订单和订单项信息存入Request域

req.setAttribute("order", order);

req.setAttribute("orderItems", orderItems);

//转发到订单详情页面: /pages/order/order\_detail.jsp

req.getRequestDispatcher("/pages/order/order\_detail.jsp").forward(req, resp);

}

/\*\* 确认收货 \*/

protected void receiveOrder(HttpServletRequest req, HttpServletResponse resp) throws IOException {

//获取订单号

String orderId = req.getParameter("orderId");

//获取用户信息

User user = (User) req.getSession(false).getAttribute("user");

//确认收货

orderService.receiveOrder(orderId, user.getId());

//重定向到订单详情页面

resp.sendRedirect(req.getContextPath() + "/OrderServlet?action=orderDetail&" + "orderId=" + orderId);

}

}

---------------------------------------------------

SearchBookByName.java文件：

package com.servlet;

import com.pojo.Book;

import com.pojo.Page;

import com.service.BookService;

import com.service.impl.BookServiceImpl;

import com.utils.WebUtils;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.IOException;

/\*\*

\* @author DamianAdam

\* @create 2021-11-11-10:07.

\*/

public class SearchBookByName extends HttpServlet {

private final BookService bookService = new BookServiceImpl();

/\*\* 处理【书名搜索】的分页请求，获得Page对象, 然后转发到 /pages/client/index.jsp \*/

@Override

protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

//1.取出请求参数

int pageNo = WebUtils.parseInt(req.getParameter("pageNo"), 1);

int pageSize = WebUtils.parseInt(req.getParameter("pageSize"), Page.PAGE\_SIZE);

String bookName = req.getParameter("bookname");

//2.搜索符合书名的图书, 获得Page对象

Page<Book> page = bookService.pageByName(bookName, pageNo, pageSize);

System.out.println(bookName);

System.out.println(page);

//设置分页栏所需的url, 别忘了附加min和max参数, 否则切页时就变成全局搜索了

page.setUrl("searchBookByName?bookname=" + bookName);

//3.将Page对象保存到request域中

req.setAttribute("page", page);

//4.转发到/pages/client/index.jsp

req.getRequestDispatcher("/pages/client/index.jsp").forward(req, resp);

}

}

---------------------------------------------------

UserServlet.java文件：

package com.servlet;

import com.google.gson.Gson;

import com.google.gson.JsonObject;

import com.google.gson.JsonPrimitive;

import com.pojo.User;

import com.service.UserService;

import com.service.impl.UserServiceImpl;

import com.utils.WebUtils;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import java.io.IOException;

import java.io.PrintWriter;

@WebServlet(name = "UserServlet", urlPatterns = "/UserServlet")

public class UserServlet extends BaseServlet {

private final UserService userService = new UserServiceImpl();

protected void login(HttpServletRequest request, HttpServletResponse response) throws ServletException,

IOException {

//1.获取表单参数

String username = request.getParameter("username");

User user = WebUtils.copyParamToBean(request.getParameterMap(), new User()); //自动注入

user = userService.login(user);

if (user != null) { //登录成功, 跳转到成功页面

HttpSession session = request.getSession(); //创建Session域对象

session.setAttribute("user", user);

request.getRequestDispatcher("pages/user/login\_success.jsp").forward(request, response);

} else { //登录失败, 将错误信息和回显信息放到request域中, 然后跳转回登录页面

request.setAttribute("msg", "账号或密码错误!");

request.setAttribute("username", username);

request.getRequestDispatcher("pages/user/login.jsp").forward(request, response);

}

}

/\*\* 验证用户名是否存在, 然后以json格式回传给客户端 \*/

protected void ajaxExistingUsername(HttpServletRequest request, HttpServletResponse response) throws

IOException {

String username = request.getParameter("username");

boolean existsUsername = userService.existsUsername(username);

JsonObject jsonObject = new JsonObject();

jsonObject.add("existingUsername", new JsonPrimitive(existsUsername));

response.setContentType("application/json");

PrintWriter writer = response.getWriter();

new Gson().toJson(jsonObject, writer);

}

/\*\* 处理注销请求, 即让Session失效, 然后重定向到主页 \*/

protected void logout(HttpServletRequest request, HttpServletResponse response) throws ServletException,

IOException {

HttpSession session = request.getSession(false);

if (session != null) {

session.invalidate();

}

response.sendRedirect(request.getContextPath() + "/index.jsp");

}

protected void regist(HttpServletRequest request, HttpServletResponse response) throws ServletException,

IOException {

request.setCharacterEncoding("UTF-8");

response.setCharacterEncoding("UTF-8");

//1.获取请求的参数

String username = request.getParameter("username");

String email = request.getParameter("email");

//2.验证验证码

String codeParamKey = "code";

if (!WebUtils.validateKaptcha(request, codeParamKey)) {

request.setAttribute("msg", "验证码错误!");

request.setAttribute("username", username);

request.setAttribute("email", email);

request.getRequestDispatcher("pages/user/regist.jsp").forward(request, response);

return;

}

//3.检查用户名是否可用

if (userService.existsUsername(username)) {

request.setAttribute("msg", "用户名已存在!");

request.setAttribute("username", username);

request.setAttribute("email", email);

request.getRequestDispatcher("pages/user/regist.jsp").forward(request, response);

return;

}

//4.注册

User user = WebUtils.copyParamToBean(request.getParameterMap(), new User()); //自动注入

boolean ans = userService.register(user);

if (ans) {

request.getRequestDispatcher("pages/user/regist\_success.jsp").forward(request, response);

} else {

//注册失败时, 将错误信息和回显信息保存到request域中, 然后转发回注册页面

request.setAttribute("msg", "注册失败, 请重试!");

request.setAttribute("username", username);

request.setAttribute("email", email);

request.getRequestDispatcher("pages/user/regist.jsp").forward(request, response);

}

}

}

---------------------------------------------------

JdbcUtils.java文件：

package com.utils;

import com.alibaba.druid.pool.DruidDataSource;

import com.alibaba.druid.pool.DruidDataSourceFactory;

import com.alibaba.druid.pool.DruidPooledConnection;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.io.InputStream;

import java.sql.Connection;

import java.sql.SQLException;

import java.util.Properties;

public class JdbcUtils {

private static DruidDataSource dataSource;

private static final ThreadLocal<Connection> CONNECTION\_THREAD\_LOCAL = new ThreadLocal<Connection>() {

@Override

protected Connection initialValue() {

try {

DruidPooledConnection connection = dataSource.getConnection();

connection.setAutoCommit(false);

return connection;

} catch (SQLException throwables) {

throwables.printStackTrace();

System.err.println("获取连接失败...");

return null;

}

}

};

static {

Properties properties = new Properties();

try {

InputStream in = JdbcUtils.class.getClassLoader().getResourceAsStream("jdbc.properties");

properties.load(in);

dataSource = (DruidDataSource) DruidDataSourceFactory.createDataSource(properties);

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

} catch (Exception e) {

e.printStackTrace();

}

}

/\*\*

\* 从数据库连接池获取连接 (默认关闭了自动提交)

\* @return 若返回null, 说明获取连接失败

\*/

public static Connection getConnection() {

return CONNECTION\_THREAD\_LOCAL.get();

}

/\*\* 提交事务, 关闭连接 \*/

public static void commitAndClose() {

try (Connection connection = CONNECTION\_THREAD\_LOCAL.get()) {

connection.commit();

} catch (SQLException throwables) {

throwables.printStackTrace();

}

//必须移除当前线程的连接, 原因是:

//Tomcat底层使用了线程池技术, 导致线程被复用, 获取到已经关闭的连接

//准确地说是获取到已经释放回连接池的连接, 如果使用了这样的连接, 会导致异常抛出

CONNECTION\_THREAD\_LOCAL.remove();

}

/\*\* 回滚事务, 关闭连接 \*/

public static void rollbackAndClose() {

try (Connection connection = CONNECTION\_THREAD\_LOCAL.get()) {

connection.rollback();

} catch (SQLException throwables) {

throwables.printStackTrace();

}

//移除当前线程的连接

CONNECTION\_THREAD\_LOCAL.remove();

}

}

---------------------------------------------------

WebUtils.java文件：

package com.utils;

import com.google.code.kaptcha.Constants;

import com.google.common.base.CaseFormat;

import org.apache.commons.beanutils.BeanUtils;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.lang.reflect.InvocationTargetException;

import java.lang.reflect.Method;

import java.util.\*;

public class WebUtils {

/\*\*

\* 使用反射实现自动注入, bean对象的属性名和set方法名必须严格对应

\* @param paramMap 保存了<属性名, 属性值>的Map

\* @param bean 要注入的bean对象

\*/

public static <T> T copyParamToBean(Map<String, String[]> paramMap, T bean) {

try {

BeanUtils.populate(bean, paramMap);

} catch (IllegalAccessException e) {

e.printStackTrace();

} catch (InvocationTargetException e) {

e.printStackTrace();

}

return bean;

}

/\*\*

\* 使用反射实现自动注入, bean对象的属性名和set方法名必须严格对应

\* @param paramMap 保存了<属性名, 属性值>的Map

\* @param bean 要注入的bean对象

\*/

public static void copyParamToBean2(Map<String, String[]> paramMap, Object bean) {

//获得存储<属性名,属性值>的Map

Set<Map.Entry<String, String[]>> entrySet = paramMap.entrySet();

//遍历每个属性, 并调用相应的set方法

for (Map.Entry<String, String[]> stringEntry : entrySet) {

//拼接出set方法名

String methodName = "set" + CaseFormat.LOWER\_CAMEL.to(CaseFormat.UPPER\_CAMEL, stringEntry.getKey());

//获得所有参数的Class对象

String[] params = stringEntry.getValue();

List<Class<? extends String>> paramClassList = new ArrayList<>();

for (String param : params) {

paramClassList.add(param.getClass());

}

try {

//获得Method对象并调用方法

Method method = bean.getClass().getDeclaredMethod(methodName, paramClassList.toArray(new Class<?>[0]));

method.invoke(bean, params);

} catch (NoSuchMethodException e) {

System.out.println(methodName + "方法不存在");

e.printStackTrace();

} catch (IllegalAccessException e) {

e.printStackTrace();

} catch (InvocationTargetException e) {

e.printStackTrace();

}

}

System.out.println(bean);

}

/\*\*

\* String转int, 失败则返回defaultValue

\*/

public static int parseInt(String value, int defaultValue) {

try {

return Integer.parseInt(value);

} catch (NumberFormatException e) {

return defaultValue;

}

}

/\*\*

\* 将请求中的验证码和Session中的验证码进行比对, 若相等且间隔时间不超过2分钟才返回true, 否则返回false

\* @param request 包含验证码参数的请求

\* @param kaptchaName 请求中的验证码参数的名字

\* @return 验证码的验证结果

\*/

public static boolean validateKaptcha(HttpServletRequest request, String kaptchaName) {

HttpSession session = request.getSession(false);

if (session == null) { //客户端禁用Cookie了, 直接返回false

return false;

}

String token = (String) session.getAttribute(Constants.KAPTCHA\_SESSION\_KEY);

Date date = (Date) session.getAttribute(Constants.KAPTCHA\_SESSION\_DATE);

session.removeAttribute(Constants.KAPTCHA\_SESSION\_KEY);

session.removeAttribute(Constants.KAPTCHA\_SESSION\_DATE);

String code = request.getParameter(kaptchaName);

//验证码相等且间隔时间不超过2分钟才返回true

return token != null && token.equals(code) && System.currentTimeMillis() - date.getTime() <= 120000;

}

}

---------------------------------------------------

jdbc.properties文件：

url:jdbc:mysql://192.168.10.180:3306/book?serverTimezone=GMT%2B8

username:root

password:123456

#监控统计用filter:stat, 日志用filter:log4j, 防御sql注入用filter:wall

filters:stat

#最大连接数

maxActive:20

#初始连接数

initialSize:1

#获得连接时的最大等待时间

maxWait:60000

#最小连接数

minIdle:1

#它有两个作用:

#1.有个【Destroy线程】会每隔timeBetweenEvictionRunsMillis毫秒检测一下连接的空闲时间，如果连接空闲时间大于等于minEvictableIdleTimeMillis则关闭连接, 直到连接池中的连接数为minIdle为止

#2.它也是testWhileIdle的判断依据，详细看testWhileIdle属性的说明

timeBetweenEvictionRunsMillis:60000

#连接保持空闲而不被关闭的最小时间

minEvictableIdleTimeMillis:300000

#用来检测连接是否有效的sql，要求是一个查询语句，常用select 'x'。

#如果validationQuery为null，testOnBorrow、testOnReturn、testWhileIdle都不会起作用。

validationQuery:select 'x

#申请连接时,若连接的空闲时间>timeBetweenEvictionRunsMillis，执行validationQuery检测连接是否有效。

testWhileIdle:true

#申请连接时是否执行validationQuery

testOnBorrow:false

#归还连接时是否执行validationQuery

testOnReturn:false

#是否缓存PreparedStatement对象

poolPreparedStatements:false

---------------------------------------------------

index.jsp文件：

<%@ page contentType="text/html;charset=UTF-8" %>

<%--主页只负责请求转发--%>

<jsp:forward page="/client/BookServlet?action=page"></jsp:forward>

---------------------------------------------------

cart.jsp文件：

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html>

<head>

<title>购物车</title>

<%@include file="../common/head.jsp" %>

<script type="text/javascript">

$(function () {

$("a.deleteItem").click(function () {

var name = $(this).parent().parent().find(":first").text()

return confirm("你确定要删除【" + name + "】吗?")

}

)

$("#clearCart").click(function () {

return confirm("你确定要清空购物车吗?")

})

$(".updateCount").change(function () {

var count = this.value

var itemId = $(this).attr("itemId")

var name = $(this).parent().parent().find(":first").text()

if (confirm("你确定要将【" + name + "】修改成【" + count + "】吗？")) {

location.href = "${pageContext.request.contextPath}/CartServlet?action=updateCount&itemId=" +

itemId + "&count=" + count

} else {

//defaultValue表示标签对象的初始value值

this.value = this.defaultValue

}

})

})

</script>

</head>

<body>

<div id="header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

<span class="wel\_word">购物车</span>

<%@include file="../common/login\_success\_menu.jsp" %>

</div>

<div id="main">

<%--购物车为空--%>

<c:if test="${empty sessionScope.cart.items}">

<table>

<tr>

<td>商品名称</td>

<td>数量</td>

<td>单价</td>

<td>总价</td>

<td>操作</td>

</tr>

<tr>

<th colspan="5"><a href="index.jsp">亲, 当前购物车为空, 快去添加点商品吧!</a></th>

</tr>

</table>

</c:if>

<%--购物车非空--%>

<c:if test="${not empty sessionScope.cart.items}">

<table>

<tr>

<td>商品名称</td>

<td>数量</td>

<td>单价</td>

<td>总价</td>

<td>操作</td>

</tr>

<c:forEach var="entry" items="${sessionScope.cart.items}">

<tr>

<td>${entry.value.name}</td>

<td><input class="updateCount" itemId="${entry.value.id}" type="text" style="width:50px"

value="${entry.value.count}"></td>

<td>${entry.value.price}</td>

<td>${entry.value.totalPrice}</td>

<td><a class="deleteItem"

href="${pageContext.request.contextPath}/CartServlet?action=deleteItem&itemId=${entry.value.id}">删除

</a></td>

</tr>

</c:forEach>

</table>

<div class="cart\_info">

<span class="cart\_span">购物车中共有<span class="b\_count">${sessionScope.cart.totalCount}</span>件商品</span>

<span class="cart\_span">总金额<span class="b\_price">${sessionScope.cart.totalPrice}</span>元</span>

<span

class="cart\_span"><a id="clearCart"

href="${pageContext.request.contextPath}/CartServlet?action=clear">清空购物车</a></span>

<span class="cart\_span"><a href="${pageContext.request.contextPath}/OrderServlet?action=checkout">去结账</a></span>

</div>

</c:if>

</div>

<%@include file="../common/footer.jsp" %>

</body>

</html>

---------------------------------------------------

checkout.jsp文件：

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html>

<head>

<title>结算页面</title>

<%@include file="../common/head.jsp" %>

<style type="text/css">

h1 {

text-align: center;

margin-top: 200px;

}

</style>

</head>

<body>

<div id="header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

<span class="wel\_word">结算</span>

<%@include file="../common/login\_success\_menu.jsp" %>

</div>

<div id="main">

<h1>你的订单已结算，订单号为${sessionScope.orderId}</h1>

</div>

<%@include file="../common/footer.jsp" %>

</body>

</html>

---------------------------------------------------

index.jsp文件：

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html>

<head>

<title>书城首页</title>

<%@include file="../common/head.jsp" %>

<script type="text/javascript">

$(function () {

/\* 将图书加入购物车 \*/

$(".add\_cart").click(function () {

var bookId = $(this).attr(`bookId`)

$.getJSON("CartServlet", "action=ajaxAddItem&bookId=" + bookId, function (data) {

$("#cartTotalCount").text(data.totalCount)

$("#cartLastName").text(data.lastItemName)

})

alert("添加成功！")

})

})

</script>

</head>

<body>

<div id="header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

<span class="wel\_word">网上书城</span>

<div>

<%--如果用户未登录, 显示登录和注册菜单--%>

<c:if test="${empty sessionScope.user}">

<a href="pages/user/login.jsp">登录</a> |

<a href="pages/user/regist.jsp">注册</a> &nbsp;&nbsp;

</c:if>

<%--如果用户已登录, 显示购物车, 后台管理等菜单--%>

<c:if test="${not empty sessionScope.user}">

<span>欢迎<span class="um\_span">${sessionScope.user.username}</span>光临金职书城</span>

<a href="UserServlet?action=logout">注销</a>&nbsp;&nbsp;

<a href="pages/cart/cart.jsp">购物车</a>

<a href="OrderServlet?action=myOrders">我的订单</a>

</c:if>

</div>

</div>

<div id="main" style="height: 800px">

<div id="book">

<div class="book\_cond">

<%-- 根据价格区间搜索图书 --%>

<form action="client/BookServlet" method="get">

<input type="hidden" name="action" value="pageByPrice"/>

<%--pageNo必须等于1 (不能写成${requestScope.page.pageNo}, 否则一旦重新点击搜索的话, 跳转到的并不是第一页) --%>

<input type="hidden" name="pageNo" value="1"/>

<input type="hidden" name="pageSize" value="${requestScope.page.pageSize}"/>

<%--min和max要等于用户上一次搜索时的价格区间, 否则切页时min,max会消失--%>

价格：<input id="min" type="text" name="min"

value="${param.min!=0&&param.max!=2147483647? param.min: ""}"> 元 -

<input id="max" type="text" name="max" value="${param.min!=0&&param.max!=2147483647? param.max: ""}"> 元

<input type="submit" value="查询"/>

</form>

<%-- 根据书名搜素图书 --%>

<form action="searchBookByName" method="get">

<input type="hidden" name="action" value="pageByPrice"/>

<%--pageNo必须等于1 (不能写成${requestScope.page.pageNo}, 否则一旦重新点击搜索的话, 跳转到的并不是第一页) --%>

<input type="hidden" name="pageNo" value="1"/>

<input type="hidden" name="pageSize" value="${requestScope.page.pageSize}"/>

书名：<input id="bookName" type="text" name="bookname" style="width: 145px">

&nbsp;&nbsp;&nbsp;&nbsp;<input type="submit" value="查询"/>

</form>

</div>

<div style="text-align: center">

<%--购物车有商品才显示--%>

<c:if test="${sessionScope.cart.totalCount > 0}">

您的购物车中有<span id="cartTotalCount">${sessionScope.cart.totalCount}</span>件商品

<div>

您刚刚将<span id="cartLastName" style="color: red">${sessionScope.lastItemName}</span>加入到了购物车中

</div>

</c:if>

</div>

<%--遍历输出图书列表--%>

<c:forEach var="book" items="${requestScope.page.items}">

<div class="b\_list">

<div class="img\_div">

<img class="book\_img" alt="" src="${pageContext.request.contextPath}/${book.imgPath}"/>

</div>

<div class="book\_info">

<div class="book\_name">

<span class="sp1">书名:</span>

<span class="sp2">${book.name}</span>

</div>

<div class="book\_author">

<span class="sp1">作者:</span>

<span class="sp2">${book.author}</span>

</div>

<div class="book\_price">

<span class="sp1">价格:</span>

<span class="sp2">${book.price}</span>

</div>

<div class="book\_sales">

<span class="sp1">销量:</span>

<span class="sp2">${book.sales}</span>

</div>

<div class="book\_amount">

<span class="sp1">库存:</span>

<span class="sp2">${book.stock}</span>

</div>

<div class="book\_add">

<button bookId="${book.id}" class="add\_cart">加入购物车</button>

</div>

</div>

</div>

</c:forEach>

</div>

<div>

<%--引入分页栏--%>

<%@include file="/pages/common/page\_nav.jsp" %>

</div>

</div>

<%@include file="../common/footer.jsp" %>

</body>

</html>

---------------------------------------------------

footer.jsp文件：

<%--

User: OneIce

Date: 2020/11/30

Time: 20:02

--%>

<%@ page contentType="text/html;charset=UTF-8" %>

<div id="bottom">

<span>

金职书城.Copyright &copy;2021

</span>

</div>

---------------------------------------------------

head.jsp文件：

<%--

User: OneIce

Date: 2020/11/30

Time: 19:40

--%>

<%@ page contentType="text/html;charset=UTF-8" %>

<meta charset="UTF-8">

<%

String basePath =

request.getScheme() + "://" + request.getServerName() + ":" + request.getServerPort() + request.getContextPath() + "/";

%>

<base href="<%=basePath%>">

<link type="text/css" rel="stylesheet" href="static/css/style.css">

<script type="text/javascript" src="static/script/jquery-3.5.1.js"></script>

---------------------------------------------------

login\_success\_menu.jsp文件：

<%--

User: OneIce

Date: 2020/11/30

Time: 19:05

--%>

<%@ page contentType="text/html;charset=UTF-8" %>

<div>

<span>欢迎<span class="um\_span">${sessionScope.user.username}</span>光临金职书城</span>

<a href="OrderServlet?action=myOrders">我的订单</a>

<a href="UserServlet?action=logout">注销</a>&nbsp;&nbsp;

<a href="javaScript:history.back()">返回</a>

</div>

---------------------------------------------------

manager\_menu.jsp文件：

<%--

User: OneIce

Date: 2020/11/30

Time: 20:18

--%>

<%@ page contentType="text/html;charset=UTF-8" %>

<div>

<a href="http://192.168.10.180:8787/r/5">销量数据可视化</a>

<a href="pages/manager/manager.jsp">返回管理系统首页</a>

</div>

---------------------------------------------------

page\_nav.jsp文件：

<%--

User: OneIce

Date: 2020/12/14

Time: 20:40

--%>

<%@ page contentType="text/html;charset=UTF-8" %>

<script type="text/javascript">

//实现"到第几页"

$(function () {

$('#searchPageButton').click(function () {

let $pageNo = $('#pn\_input').val() //获得指定的页码

if ($pageNo <= 0 || $pageNo > ${requestScope.page.pageTotal}) {

$pageNo = 1

}

//设置window.location对象的href属性

location.href = "${pageContext.request.contextPath}/${requestScope.page.url}&pageNo=" + $pageNo + "&pageSize=${requestScope.page.pageSize}"

})

})

</script>

<%--分页栏--%>

<div id="page\_nav">

<a href="${pageContext.request.contextPath}/${requestScope.page.url}&pageNo=1&pageSize=${requestScope.page.pageSize}">首页</a>

<%--页码大于1才显示"上一页"--%>

<c:if test="${requestScope.page.pageNo > 1}">

<a href="${pageContext.request.contextPath}/${requestScope.page.url}&pageNo=${requestScope.page.pageNo - 1}&pageSize=${requestScope.page.pageSize}">上一页</a>

</c:if>

<%--开始输出页码--%>

<%-- 首先获取页码范围, 保存在pageScope中 --%>

<c:choose>

<%-- 情况1: 总页数≤5, 页码范围: [1, 总页数] --%>

<c:when test="${requestScope.page.pageTotal <= 5}">

<c:set scope="page" var="begin" value="1"></c:set>

<c:set scope="page" var="end" value="${requestScope.page.pageTotal}"></c:set>

</c:when>

<%-- 情况2：总页码>5 --%>

<c:otherwise>

<c:choose>

<%-- 当前页码是前2页, 页码范围：[1, 5] --%>

<c:when test="${requestScope.page.pageNo <=2}">

<c:set scope="page" var="begin" value="1"></c:set>

<c:set scope="page" var="end" value="5"></c:set>

</c:when>

<%-- 当前页码是后2页，页码范围：[总页数-5+1, 总页数] --%>

<c:when test="${requestScope.page.pageNo >= requestScope.page.pageTotal - 1}">

<c:set scope="page" var="begin" value="${requestScope.page.pageTotal - 4}"></c:set>

<c:set scope="page" var="end" value="${requestScope.page.pageTotal}"></c:set>

</c:when>

<%-- 当前页码是中间页，页码范围：[当前页码-2, 当前页码+2] --%>

<c:otherwise>

<c:set scope="page" var="begin" value="${requestScope.page.pageNo - 2}"></c:set>

<c:set scope="page" var="end" value="${requestScope.page.pageNo + 2}"></c:set>

</c:otherwise>

</c:choose>

</c:otherwise>

</c:choose>

<%-- 从pageContext中取出页码范围, 输出页码 --%>

<c:forEach var="i" begin="${pageScope.begin}" end="${pageScope.end}">

<%-- 当前页码不可跳转 --%>

<c:if test="${requestScope.page.pageNo == i}">

【${i}】

</c:if>

<%-- 其他页码可以跳转 --%>

<c:if test="${requestScope.page.pageNo != i}">

<a href="${pageContext.request.contextPath}/${requestScope.page.url}&pageNo=${i}&pageSize=${requestScope.page.pageSize}">${i}</a>

</c:if>

<%=' '%>

</c:forEach>

<%--结束输出页码--%>

<%--页码小于总页数才显示"下一页"--%>

<c:if test="${requestScope.page.pageNo < requestScope.page.pageTotal}">

<a href="${pageContext.request.contextPath}/${requestScope.page.url}&pageNo=${requestScope.page.pageNo + 1}&pageSize=${requestScope.page.pageSize}">下一页</a>

</c:if>

<a href="${pageContext.request.contextPath}/${requestScope.page.url}&pageNo=${requestScope.page.pageTotal}&pageSize=${requestScope.page.pageSize}">末页</a>

共${requestScope.page.pageTotal}页，${requestScope.page.pageTotalCount}条记录

<%--"到第几页", 通过JQuery绑定单击事件来实现--%>

到第<input name="pn" value="${requestScope.page.pageNo}" id="pn\_input"/>页

<input type="button" id="searchPageButton" value="确定">

</div>

---------------------------------------------------

error404.jsp文件：

<%@ page contentType="text/html;charset=UTF-8" %>

<html>

<head>

<title>书城</title>

<style type="text/css">

\*{

margin:0;

padding:0;

}

body{

font-family: 'Open Sans', sans-serif;

font-size: 14px;

color: #3c3c3c;

background-color: #ECECEC;

}

#duobei\_wrap{

float: left;

position: relative;

width: 600px;

/\*height: 600px;\*/

margin-top: 150px;

/\*margin-left: 100px;\*/

-webkit-traisition: all 1s ease-out;

-moz-traisition: all 1s ease-out;

-o-traisition: all 1s ease-out;

traisition: all 1s ease-out;

}

#duobei{

margin: 60px auto;

position: absolute;

-webkit-transform: rotate(-30deg);

-moz-transform: rotate(-30deg);

-o-transform: rotate(-30deg);

transform: rotate(-30deg);

-webkit-transition: all 1s ease-in-out;

-webkit-transform-origin: 50%;

transition: all 1s ease-in-out;

transform-origin: 50%;

}

.fl {float:left;display:inline;}

.fr {float:right;display:inline;}

.cf {display:block;zoom:1;}

\* html .cf {height:1%;}

.cf:after {content:" ";display:block;height:0;clear:both;visibility:hidden;}

.logo\_wrap{

height: 84px;

}

#title{

float: left;

line-height: 80px;

margin-left: 20px;

display: inline-block;

}

i{

display: inline-block;

font-size: 24px;

font-family: "Microsoft YaHei";

font-style: normal;

-webkit-animation-duration: 1.4s;/\*动画持续时间\*/

-webkit-animation-iteration-count: 1;/\*定义循环资料，infinite为无限次\*/

-webkit-animation-timing-function: ease;/\*动画频率\*/

animation-duration: 1.4s;

animation-iteration-count: 1;

animation-timing-function: ease;

}

i:nth-child(-n+2){

color: #00b7fa;

font-size: 24px;

}

i:nth-child(n+3){

color: #666;

font-size: 24px;

}

.title\_move i:nth-child(1){

-webkit-animation-name:'zoom-d';

animation-name:'zoom-d';

}

.title\_move i:nth-child(2){

-webkit-animation-name:'zoom-b';

animation-name:'zoom-b';

}

.title\_move i:nth-child(3){

-webkit-animation-name:'zoom-g';

animation-name:'zoom-g';

}

.title\_move i:nth-child(4){

-webkit-animation-name:'zoom-k';

animation-name:'zoom-k';

}

.title\_move i:nth-child(5){

-webkit-animation-name:'zoom-c';

animation-name:'zoom-c';

}

@-webkit-keyframes 'zoom-d' {

0% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

10% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

30% {

-webkit-transform: scale(1.1)

transform: scale(1.1)

}

50% {

-webkit-transform: scale(1);

transform: scale(1);

opacity: 1.0

}

100% {

-webkit-transform: scale(1);

transform: scale(1);

opacity: 1.0

}

}

@-webkit-keyframes 'zoom-b' {

0% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

20% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

40% {

-webkit-transform: scale(1.1)

transform: scale(1.1)

}

60% {

-webkit-transform: scale(1);

transform: scale(1);

opacity: 1.0

}

}

@-webkit-keyframes 'zoom-g' {

0% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

30% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

50% {

-webkit-transform: scale(1.1)

transform: scale(1.1)

}

70% {

-webkit-transform: scale(1);

transform: scale(1);

opacity: 1.0

}

}

@-webkit-keyframes 'zoom-k' {

0% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

40% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

60% {

-webkit-transform: scale(1.1)

transform: scale(1.1)

}

80% {

-webkit-transform: scale(1);

transform: scale(1);

opacity: 1.0

}

}

@-webkit-keyframes 'zoom-c' {

0% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

50% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

70% {

-webkit-transform: scale(1.1)

transform: scale(1.1)

}

90% {

-webkit-transform: scale(1);

transform: scale(1);

opacity: 1.0

}

}

.not\_found\_tip{

/\* float: left;\*/

text-align: left;

color: #666;

font-family: "Microsoft YaHei";

font-size: 20px;

text-shadow: 0 1px 0 #fff;

letter-spacing: 1px;

line-height: 2em;

text-indent: 18px;

-webkit-transform: rotate(0deg);

transform: rotate(0deg);

}

.reason{

margin-top: 54px;

display: block;

}

.possible{

font-weight: bold;

text-indent: 18px;

margin-top: 10px;

}

.reason ul{

list-style: square;

padding-left: 82px;

margin-top: 10px;

}

.reason li{

line-height: 22px;

margin: 8px 0;

}

.line{

float: left;

width: 1px;

height: 400px;

margin-left: 50px;

margin-top: 100px;

background: #ddd;

box-shadow: 1px 0 2px #FFF;

}

#content{

width: 1202px;

margin: 0 auto;

}

.not\_found{

float: right;

width: 400px;

height: auto;

margin-top: 150px;

position: relative;

}

.not\_found\_404 {

text-align: center;

font-family: cursive,sans-serif;

font-size: 150px;

font-weight: bold;

line-height: 150px;

letter-spacing: 25px;

color: #fff;

-webkit-transform: rotate(18deg);

transform: rotate(18deg);

-webkit-transform-origin: 12% 60%;

transform-origin: 12% 60%;

}

.not\_found\_404 span {

float: left;

cursor: pointer;

text-shadow: 0px 0px 2px #686868,

0px 1px 1px #ddd,

0px 2px 1px #d6d6d6,

0px 3px 1px #ccc,

0px 4px 1px #c5c5c5,

0px 5px 1px #c1c1c1,

0px 6px 1px #bbb,

0px 7px 1px #777,

0px 8px 3px rgba(100, 100, 100, 0.4),

0px 9px 5px rgba(100, 100, 100, 0.1),

0px 10px 7px rgba(100, 100, 100, 0.15),

0px 11px 9px rgba(100, 100, 100, 0.2),

0px 12px 11px rgba(100, 100, 100, 0.25),

0px 13px 15px rgba(100, 100, 100, 0.3);

-webkit-transition: all .1s linear;

transition: all .1s linear;

}

.not\_found\_404 span:hover {

text-shadow: 0px 0px 2px #686868,

0px 1px 1px #fff,

0px 2px 1px #fff,

0px 3px 1px #fff,

0px 4px 1px #fff,

0px 5px 1px #fff,

0px 6px 1px #fff,

0px 7px 1px #777,

0px 8px 3px #fff,

0px 9px 5px #fff,

0px 10px 7px #fff,

0px 11px 9px #fff,

0px 12px 11px #fff,

0px 13px 15px #fff;

-webkit-transition: all .1s linear;

transition: all .1s linear;

}

.ribbon {

position: absolute;

top: 0;

left: 0;

width: 130px;

height: 40px;

background: rgba(0, 0, 0, 0.1);

background: -webkit-gradient(linear, 555% 20%, 0% 92%, from(rgba(0, 0, 0, 0.1)), to(rgba(0, 0, 0, 0.0)), color-stop(.1,rgba(0, 0, 0, 0.2)));

background: -moz-linear-gradient(555% 0% 180deg, rgba(0,0,0,0.1), rgba(0,0,0,0.2) 10%, rgba(0,0,0,0.0));

border-left: 1px dashed rgba(0, 0, 0, 0.1);

border-right: 1px dashed rgba(0, 0, 0, 0.1);

box-shadow: 0px 0px 12px rgba(0,0,0,0.2);

}

@-webkit-keyframes 'not'{

0%{

}

25%{

-webkit-transform: rotate(12deg);

}

50%{

-webkit-transform: rotate(18deg);

}

75%{

-webkit-transform: rotate(12deg);

}

100%{

-webkit-transform: rotate(18deg);

}

}

.not\_found\_404{

-webkit-animation-duration: 4s;

-webkit-animation-name: not;

-webkit-animation-iteration-count: infinite;

-webkit-animation-timing-function: ease-in-out;

}

.btn{

width: 100%;

margin-top: 100px;

}

.button{

-webkit-box-shadow: inset 0px 1px 0px rgba(255, 255, 255, 0.5), 0px 1px 2px rgba(0, 0, 0, 0.15);

-moz-box-shadow: inset 0px 1px 0px rgba(255, 255, 255, 0.5), 0px 1px 2px rgba(0, 0, 0, 0.15);

box-shadow: inset 0px 1px 0px rgba(255, 255, 255, 0.5), 0px 1px 2px rgba(0, 0, 0, 0.15);

background-color: #ffffff;

background: -webkit-gradient(linear, 50% 0%, 50% 100%, color-stop(0%, whitesmoke), color-stop(100%, #fff));

background: -webkit-linear-gradient(top, whitesmoke, #fff);

background: -moz-linear-gradient(top, whitesmoke, #fff);

background: -o-linear-gradient(top, whitesmoke, #fff);

background: linear-gradient(top, whitesmoke, #fff);

display: -moz-inline-stack;

display: inline-block;

vertical-align: middle;

zoom: 1;

border: 1px solid #d4d4d4;

height: 32px;

line-height: 32px;

padding: 0px 25.6px;

font-weight: 300;

font-size: 14px;

font-family: "HelveticaNeue-Light", "Helvetica Neue Light", "Helvetica Neue", Helvetica, Arial, "Lucida Grande", sans-serif;

color: #666666;

text-shadow: 0 1px 1px white;

margin: 0;

text-decoration: none;

text-align: center;

-webkit-border-radius: 30px;

-moz-border-radius: 30px;

-ms-border-radius: 30px;

-o-border-radius: 30px;

border-radius: 30px;

-moz-animation:funny 3s linear infinite;

-o-animation:funny 3s linear infinite;

animation:funny 3s linear infinite;

-webkit-animation:funny 3s linear infinite;

margin-right: 40px;

clear: left;

}

@-webkit-keyframes funny{

0% { -webkit-box-shadow:0px 0px 0px whitesmoke; }

50% { -webkit-box-shadow:0px 0px 20px rgba(82, 168, 236, 0.8); }

100% { -webkit-box-shadow:0px 0px 0px whitesmoke; }

}

@-moz-keyframes funny{

0% { -moz-box-shadow:0px 0px 0px whitesmoke; }

50% { -moz-box-shadow:0px 0px 20px rgba(82, 168, 236, 0.8); }

100% { -moz-box-shadow:0px 0px 0px whitesmoke; }

}

</style>

<script type="text/javascript" src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.7.2.js"></script>

<script type="text/javascript">

$(function () {

$(function(){

var time=setTimeout(function(){

$("#duobei").css({

"-webkit-transform": "rotate(-30deg) scale(0.5)",

});

},2000);

$("#title").hide();

var title=setTimeout(function(){

$("#title").addClass("title\_move");

$("#title").show();

},400);

$(".cancle").click(function(){

window.history.go(-1);

return false;

});

})

});

</script>

</head>

<body>

<div id="content" class="cf">

<div id="duobei\_wrap">

<div class="logo\_wrap cf">

<div id="title"><i>金</i><i>职</i><i>书</i><i>城</i><i>！</i></div>

</div>

<div class="reason">

<p class="not\_found\_tip">Not Found :( 很抱歉，您访问的页面不存在!</p>

<p class="possible">可能原因：</p>

<ul>

<li>该页面已被删除，但却被搜索引擎收录了，通过搜索引擎访问就会看到此页面</li>

<li>您手动输入了一个从来没有的访问url，这样系统也是无法识别您的url，报错</li>

</ul>

</div>

</div>

<div class="line"></div>

<div class="not\_found">

<i class="ribbon"></i>

<div class="not\_found\_404 cf">

<span>4</span>

<span>0</span>

<span>4</span>

</div>

<div class="btn">

<a href="${pageContext.request.contextPath}" class="button button-rounded">看看首页</a>

</div>

</div>

</div>

</body>

</html>

---------------------------------------------------

error500.jsp文件：

<%--

User: OneIce

Date: 2021/1/25

Time: 20:55

--%>

<%@ page contentType="text/html;charset=UTF-8" %>

<html>

<head>

<title>书城</title>

<style type="text/css">

\*{

margin:0;

padding:0;

}

body{

font-family: 'Open Sans', sans-serif;

font-size: 14px;

color: #3c3c3c;

background-color: #ECECEC;

}

#duobei\_wrap{

float: left;

position: relative;

width: 600px;

/\*height: 600px;\*/

margin-top: 150px;

/\*margin-left: 100px;\*/

-webkit-traisition: all 1s ease-out;

-moz-traisition: all 1s ease-out;

-o-traisition: all 1s ease-out;

traisition: all 1s ease-out;

}

#duobei{

margin: 60px auto;

position: absolute;

-webkit-transform: rotate(-30deg);

-moz-transform: rotate(-30deg);

-o-transform: rotate(-30deg);

transform: rotate(-30deg);

-webkit-transition: all 1s ease-in-out;

-webkit-transform-origin: 50%;

transition: all 1s ease-in-out;

transform-origin: 50%;

}

.fl {float:left;display:inline;}

.fr {float:right;display:inline;}

.cf {display:block;zoom:1;}

\* html .cf {height:1%;}

.cf:after {content:" ";display:block;height:0;clear:both;visibility:hidden;}

.logo\_wrap{

height: 84px;

}

#title{

float: left;

line-height: 80px;

margin-left: 20px;

display: inline-block;

}

i{

display: inline-block;

font-size: 24px;

font-family: "Microsoft YaHei";

font-style: normal;

-webkit-animation-duration: 1.4s;/\*动画持续时间\*/

-webkit-animation-iteration-count: 1;/\*定义循环资料，infinite为无限次\*/

-webkit-animation-timing-function: ease;/\*动画频率\*/

animation-duration: 1.4s;

animation-iteration-count: 1;

animation-timing-function: ease;

}

i:nth-child(-n+2){

color: #00b7fa;

font-size: 24px;

}

i:nth-child(n+3){

color: #666;

font-size: 24px;

}

.title\_move i:nth-child(1){

-webkit-animation-name:'zoom-d';

animation-name:'zoom-d';

}

.title\_move i:nth-child(2){

-webkit-animation-name:'zoom-b';

animation-name:'zoom-b';

}

.title\_move i:nth-child(3){

-webkit-animation-name:'zoom-g';

animation-name:'zoom-g';

}

.title\_move i:nth-child(4){

-webkit-animation-name:'zoom-k';

animation-name:'zoom-k';

}

.title\_move i:nth-child(5){

-webkit-animation-name:'zoom-c';

animation-name:'zoom-c';

}

@-webkit-keyframes 'zoom-d' {

0% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

10% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

30% {

-webkit-transform: scale(1.1)

transform: scale(1.1)

}

50% {

-webkit-transform: scale(1);

transform: scale(1);

opacity: 1.0

}

100% {

-webkit-transform: scale(1);

transform: scale(1);

opacity: 1.0

}

}

@-webkit-keyframes 'zoom-b' {

0% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

20% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

40% {

-webkit-transform: scale(1.1)

transform: scale(1.1)

}

60% {

-webkit-transform: scale(1);

transform: scale(1);

opacity: 1.0

}

}

@-webkit-keyframes 'zoom-g' {

0% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

30% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

50% {

-webkit-transform: scale(1.1)

transform: scale(1.1)

}

70% {

-webkit-transform: scale(1);

transform: scale(1);

opacity: 1.0

}

}

@-webkit-keyframes 'zoom-k' {

0% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

40% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

60% {

-webkit-transform: scale(1.1)

transform: scale(1.1)

}

80% {

-webkit-transform: scale(1);

transform: scale(1);

opacity: 1.0

}

}

@-webkit-keyframes 'zoom-c' {

0% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

50% {

-webkit-transform: scale(0.6);

transform: scale(0.6);

opacity: 0;

}

70% {

-webkit-transform: scale(1.1)

transform: scale(1.1)

}

90% {

-webkit-transform: scale(1);

transform: scale(1);

opacity: 1.0

}

}

.not\_found\_tip{

/\* float: left;\*/

text-align: left;

color: #666;

font-family: "Microsoft YaHei";

font-size: 20px;

text-shadow: 0 1px 0 #fff;

letter-spacing: 1px;

line-height: 2em;

text-indent: 18px;

-webkit-transform: rotate(0deg);

transform: rotate(0deg);

}

.reason{

margin-top: 54px;

display: block;

}

.possible{

font-weight: bold;

text-indent: 18px;

margin-top: 10px;

}

.reason ul{

list-style: square;

padding-left: 82px;

margin-top: 10px;

}

.reason li{

line-height: 22px;

margin: 8px 0;

}

.line{

float: left;

width: 1px;

height: 400px;

margin-left: 50px;

margin-top: 100px;

background: #ddd;

box-shadow: 1px 0 2px #FFF;

}

#content{

width: 1202px;

margin: 0 auto;

}

.not\_found{

float: right;

width: 400px;

height: auto;

margin-top: 150px;

position: relative;

}

.not\_found\_404 {

text-align: center;

font-family: cursive,sans-serif;

font-size: 150px;

font-weight: bold;

line-height: 150px;

letter-spacing: 25px;

color: #fff;

-webkit-transform: rotate(18deg);

transform: rotate(18deg);

-webkit-transform-origin: 12% 60%;

transform-origin: 12% 60%;

}

.not\_found\_404 span {

float: left;

cursor: pointer;

text-shadow: 0px 0px 2px #686868,

0px 1px 1px #ddd,

0px 2px 1px #d6d6d6,

0px 3px 1px #ccc,

0px 4px 1px #c5c5c5,

0px 5px 1px #c1c1c1,

0px 6px 1px #bbb,

0px 7px 1px #777,

0px 8px 3px rgba(100, 100, 100, 0.4),

0px 9px 5px rgba(100, 100, 100, 0.1),

0px 10px 7px rgba(100, 100, 100, 0.15),

0px 11px 9px rgba(100, 100, 100, 0.2),

0px 12px 11px rgba(100, 100, 100, 0.25),

0px 13px 15px rgba(100, 100, 100, 0.3);

-webkit-transition: all .1s linear;

transition: all .1s linear;

}

.not\_found\_404 span:hover {

text-shadow: 0px 0px 2px #686868,

0px 1px 1px #fff,

0px 2px 1px #fff,

0px 3px 1px #fff,

0px 4px 1px #fff,

0px 5px 1px #fff,

0px 6px 1px #fff,

0px 7px 1px #777,

0px 8px 3px #fff,

0px 9px 5px #fff,

0px 10px 7px #fff,

0px 11px 9px #fff,

0px 12px 11px #fff,

0px 13px 15px #fff;

-webkit-transition: all .1s linear;

transition: all .1s linear;

}

.ribbon {

position: absolute;

top: 0;

left: 0;

width: 130px;

height: 40px;

background: rgba(0, 0, 0, 0.1);

background: -webkit-gradient(linear, 555% 20%, 0% 92%, from(rgba(0, 0, 0, 0.1)), to(rgba(0, 0, 0, 0.0)), color-stop(.1,rgba(0, 0, 0, 0.2)));

background: -moz-linear-gradient(555% 0% 180deg, rgba(0,0,0,0.1), rgba(0,0,0,0.2) 10%, rgba(0,0,0,0.0));

border-left: 1px dashed rgba(0, 0, 0, 0.1);

border-right: 1px dashed rgba(0, 0, 0, 0.1);

box-shadow: 0px 0px 12px rgba(0,0,0,0.2);

}

@-webkit-keyframes 'not'{

0%{

}

25%{

-webkit-transform: rotate(12deg);

}

50%{

-webkit-transform: rotate(18deg);

}

75%{

-webkit-transform: rotate(12deg);

}

100%{

-webkit-transform: rotate(18deg);

}

}

.not\_found\_404{

-webkit-animation-duration: 4s;

-webkit-animation-name: not;

-webkit-animation-iteration-count: infinite;

-webkit-animation-timing-function: ease-in-out;

}

.btn{

width: 100%;

margin-top: 100px;

}

.button{

-webkit-box-shadow: inset 0px 1px 0px rgba(255, 255, 255, 0.5), 0px 1px 2px rgba(0, 0, 0, 0.15);

-moz-box-shadow: inset 0px 1px 0px rgba(255, 255, 255, 0.5), 0px 1px 2px rgba(0, 0, 0, 0.15);

box-shadow: inset 0px 1px 0px rgba(255, 255, 255, 0.5), 0px 1px 2px rgba(0, 0, 0, 0.15);

background-color: #ffffff;

background: -webkit-gradient(linear, 50% 0%, 50% 100%, color-stop(0%, whitesmoke), color-stop(100%, #fff));

background: -webkit-linear-gradient(top, whitesmoke, #fff);

background: -moz-linear-gradient(top, whitesmoke, #fff);

background: -o-linear-gradient(top, whitesmoke, #fff);

background: linear-gradient(top, whitesmoke, #fff);

display: -moz-inline-stack;

display: inline-block;

vertical-align: middle;

zoom: 1;

border: 1px solid #d4d4d4;

height: 32px;

line-height: 32px;

padding: 0px 25.6px;

font-weight: 300;

font-size: 14px;

font-family: "HelveticaNeue-Light", "Helvetica Neue Light", "Helvetica Neue", Helvetica, Arial, "Lucida Grande", sans-serif;

color: #666666;

text-shadow: 0 1px 1px white;

margin: 0;

text-decoration: none;

text-align: center;

-webkit-border-radius: 30px;

-moz-border-radius: 30px;

-ms-border-radius: 30px;

-o-border-radius: 30px;

border-radius: 30px;

-moz-animation:funny 3s linear infinite;

-o-animation:funny 3s linear infinite;

animation:funny 3s linear infinite;

-webkit-animation:funny 3s linear infinite;

margin-right: 40px;

clear: left;

}

@-webkit-keyframes funny{

0% { -webkit-box-shadow:0px 0px 0px whitesmoke; }

50% { -webkit-box-shadow:0px 0px 20px rgba(82, 168, 236, 0.8); }

100% { -webkit-box-shadow:0px 0px 0px whitesmoke; }

}

@-moz-keyframes funny{

0% { -moz-box-shadow:0px 0px 0px whitesmoke; }

50% { -moz-box-shadow:0px 0px 20px rgba(82, 168, 236, 0.8); }

100% { -moz-box-shadow:0px 0px 0px whitesmoke; }

}

</style>

<script type="text/javascript" src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.7.2.js"></script>

<script type="text/javascript">

$(function () {

$(function(){

var time=setTimeout(function(){

$("#duobei").css({

"-webkit-transform": "rotate(-30deg) scale(0.5)",

});

},2000);

$("#title").hide();

var title=setTimeout(function(){

$("#title").addClass("title\_move");

$("#title").show();

},400);

$(".cancle").click(function(){

window.history.go(-1);

return false;

});

})

});

</script>

</head>

<body>

<div id="content" class="cf">

<div id="duobei\_wrap">

<div class="logo\_wrap cf">

<div id="title"><i>金</i><i>职</i><i>书</i><i>城</i><i>！</i></div>

</div>

<div class="reason">

<p class="not\_found\_tip">Not Found :( 很抱歉，服务器内部出现错误，暂时无法访问!</p>

<p class="possible">可能原因：</p>

<ul>

<li>服务器内部出了错误, 程序员小哥正在努力为您抢修!!</li>

</ul>

</div>

</div>

<div class="line"></div>

<div class="not\_found">

<i class="ribbon"></i>

<div class="not\_found\_404 cf">

<span>5</span>

<span>0</span>

<span>0</span>

</div>

<div class="btn">

<a href="${pageContext.request.contextPath}" class="button button-rounded">看看首页</a>

</div>

</div>

</div>

</body>

</html>

---------------------------------------------------

book\_edit.jsp文件：

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html>

<head>

<%@include file="../common/head.jsp" %>

<title>编辑图书</title>

<style type="text/css">

h1 {

text-align: center;

margin-top: 200px;

}

h1 a {

color: red;

}

input {

text-align: center;

}

</style>

</head>

<body>

<div id="header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

<span class="wel\_word">编辑图书</span>

<%@include file="../common/manager\_menu.jsp" %>

</div>

<div id="main">

<%--提交到BookServlet--%>

<form action="${pageContext.request.contextPath}/manager/BookServlet" method="post">

<%--添加或修改完图书后, 要显示的页码--%>

<input type="hidden" name="pageNo" value="${param.pageNo}"/>

<%--通过判断请求中是否携带id参数, 来决定BookServlet要调用add()还是update()方法--%>

<input type="hidden" name="action" value="${empty param.id?"add":"update"}"/>

<%--用于修改图书时, 需要携带的图书id--%>

<input type="hidden" name="id" value="${requestScope.book.id}"/>

<table>

<tr>

<td>名称</td>

<td>价格</td>

<td>作者</td>

<td>销量</td>

<td>库存</td>

<td colspan="2">操作</td>

</tr>

<tr>

<td><input name="name" type="text" value="${requestScope.book.name}"/></td>

<td><input name="price" type="text" value="${requestScope.book.price}"/></td>

<td><input name="author" type="text" value="${requestScope.book.author}"/></td>

<td><input name="sales" type="text" value="${requestScope.book.sales}"/></td>

<td><input name="stock" type="text" value="${requestScope.book.stock}"/></td>

<td><input type="submit" value="提交"/></td>

</tr>

</table>

</form>

</div>

<%@include file="../common/footer.jsp" %>

</body>

</html>

---------------------------------------------------

book\_manager.jsp文件：

<%--suppress CheckTagEmptyBody --%>

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html>

<head>

<title>图书管理</title>

<%@include file="../common/head.jsp" %>

<script type="text/javascript">

$(function () {

//给用于删除的a标签绑定单击事件, 提示用户是否确认删除

$("a.deleteClass").click(function () {

/\*\*

\* confirm()是一个确认提示框,返回类型boolean

\* return false → 阻止标签的默认行为 (对于a标签就是阻止跳转)

\* \*/

return confirm("你确定要删除《" + $(this).parent().parent().find("td:first").text() + "》吗?")

})

})

</script>

</head>

<body>

<div id="header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

<span class="wel\_word">图书管理系统</span>

<%@include file="../common/manager\_menu.jsp" %>

</div>

<div id="main" style="height: 600px">

<table>

<tr>

<td>名称</td>

<td>价格</td>

<td>作者</td>

<td>销量</td>

<td>库存</td>

<td colspan="2">操作</td>

</tr>

<%--核心代码--%>

<%--取出request域的books, 显示所有图书信息--%>

<c:forEach items="${requestScope.page.items}" var="book">

<tr>

<td>${book.name}</td>

<td>${book.price}</td>

<td>${book.author}</td>

<td>${book.sales}</td>

<td>${book.stock}</td>

<td><a href="${pageContext.request.contextPath}/manager/BookServlet?action=getBook&id=${book.id}&pageNo=${requestScope.page.pageNo}">修改</a>

</td>

<td><a class="deleteClass"

href="${pageContext.request.contextPath}/manager/BookServlet?action=delete&id=${book.id}&pageNo=${requestScope.page.pageNo}">删除</a>

</td>

</tr>

</c:forEach>

<tr>

<td></td>

<td></td>

<td></td>

<td></td>

<td></td>

<td></td>

<td><a href="pages/manager/book\_edit.jsp?pageNo=${requestScope.page.pageTotal}">添加图书</a></td>

</tr>

</table>

<br/>

<%--引入分页栏--%>

<%@include file="/pages/common/page\_nav.jsp"%>

</div>

<%@include file="../common/footer.jsp" %>

</body>

</html>

---------------------------------------------------

manager.jsp文件：

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html>

<head>

<title>后台管理</title>

<%@include file="../common/head.jsp" %>

<style type="text/css">

#manager\_info {

text-align: center;

margin-top: 200px;

}

</style>

</head>

<body>

<div id="header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

<span class="wel\_word">后台管理系统</span>

</div>

<div id="main">

<div id="manager\_info">

<h1>欢迎管理员进入后台管理系统</h1>

<a href="manager/BookServlet?action=page" style="font-size: 20px">图书管理</a>

<a href="manager/OrderServlet?action=page" style="font-size: 20px">订单管理</a>

</div>

</div>

<%@include file="../common/footer.jsp"%>

</body>

</html>

---------------------------------------------------

manager\_login.jsp文件：

<%--

Time: 14:00

To change this template use File | Settings | File Templates.

--%>

<%@ page contentType="text/html;charset=UTF-8" language="java" %>

<html>

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>书城管理员登录页面</title>

<style type="text/css">

\* {

/\* 初始化 清除页面元素的内外边距 \*/

padding: 0;

margin: 0;

/\* 盒子模型 \*/ Created by IntelliJ IDEA.

User: DamianAdam

Date: 2021/11/11

box-sizing: border-box;

}

body {

/\* 弹性布局 让页面元素垂直+水平居中 \*/

display: flex;

justify-content: center;

align-items: center;

/\* 让页面始终占浏览器可视区域总高度 \*/

height: 100vh;

/\* 背景渐变色 \*/

background: linear-gradient(45deg, blue, red);

}

.login {

/\* 弹性布局 让子元素称为弹性项目 \*/

display: flex;

/\* 让弹性项目垂直排列 原理是改变弹性盒子的主轴方向 父元素就是弹性盒子 现在改变后的主轴方向是向下了 \*/

flex-direction: column;

/\* 让弹性项目在交叉轴方向水平居中 现在主轴的方向是向下 交叉轴的方向是与主轴垂直 交叉轴的方向是向右 \*/

align-items: center;

width: 400px;

padding: 40px;

background-color: rgba(0, 0, 0, 0.2);

box-shadow: 0 15px 25px rgba(0, 0, 0, 0.4);

float: right;

}

.login h2 {

color: #fff;

margin-bottom: 30px;

}

.login .login\_box {

/\* 相对定位 \*/

position: relative;

width: 100%;

}

.login .login\_box input {

/\* 清除input框自带的边框和轮廓 \*/

outline: none;

border: none;

width: 100%;

padding: 10px 0;

margin-bottom: 30px;

color: #fff;

font-size: 16px;

border-bottom: 1px solid #fff;

/\* 背景颜色为透明色 \*/

background-color: transparent;

}

.login .login\_box label {

position: absolute;

top: 0;

left: 0;

padding: 10px 0;

color: #fff;

/\* 这个属性的默认值是auto 默认是这个元素可以被点击 但是如果我们写了none 就是这个元素不能被点击 , 就好像它可见但是不能用 可望而不可即 \*/

/\* 这个就是两者的区别 \*/

pointer-events: none;

/\* 加个过渡 \*/

transition: all 0.5s;

}

/\* :focus 选择器是当input获得焦点是触发的样式 + 是相邻兄弟选择器 去找与input相邻的兄弟label \*/

/\* :valid 选择器是判断input框的内容是否合法,如果合法会执行下面的属性代码,不合法就不会执行,我们刚开始写布局的时候给input框写了required 我们删掉看对比 当没有required的话input框的值就会被认为一直合法,所以一直都是下方的样式 ,但是密码不会,密码框内的值为空,那么这句话局不合法,required不能为空 当我们给密码框写点东西的时候才会执行以下代码\*/

.login .login\_box input:focus+label,

.login .login\_box input:valid+label {

top: -20px;

color: #f7ae10;

font-size: 12px;

}

.login a {

overflow: hidden;

position: relative;

padding: 10px 20px;

color: #f7ae10;

/\* 取消a表现原有的下划线 \*/

text-decoration: none;

/\* 同样加个过渡 \*/

transition: all 0.5s;

}

.login a:hover {

color: #fff;

border-radius: 5px;

background-color: #f7ae10;

box-shadow: 0 0 5px #f7ae10, 0 0 25px #f7ae10, 0 0 50px #f7ae10,

0 0 100px #f7ae10;

}

.login a span {

position: absolute;

}

.login a span:first-child {

top: 0;

left: -100%;

width: 100%;

height: 2px;

/\* to right 就是往右边 下面的同理 \*/

background: linear-gradient(to right, transparent, #f7ae10);

/\* 动画 名称 时长 linear是匀速运动 infinite是无限次运动 \*/

animation: move1 4s linear infinite;

}

.login a span:nth-child(2) {

right: 0;

top: -100%;

width: 2px;

height: 100%;

background: linear-gradient(transparent, #f7ae10);

/\* 这里多了个0.25s其实是延迟时间 \*/

animation: move2 4s linear 0.25s infinite;

}

.login a span:nth-child(3) {

right: -100%;

bottom: 0;

width: 100%;

height: 2px;

background: linear-gradient(to left, transparent, #f7ae10);

animation: move3 4s linear 0.5s infinite;

}

.login a span:last-child {

left: 0;

bottom: -100%;

width: 2px;

height: 100%;

background: linear-gradient(#f7ae10, transparent);

animation: move4 4s linear 0.75s infinite;

}

/\* 动画 \*/

@keyframes move1 {

0% {

left: -100%;

}

50%,

100% {

left: 100%;

}

}

@keyframes move2 {

0% {

top: -100%;

}

50%,

100% {

top: 100%;

}

}

@keyframes move3 {

0% {

right: -100%;

}

50%,

100% {

right: 100%;

}

}

@keyframes move4 {

0% {

bottom: -100%;

}

50%,

100% {

bottom: 100%;

}

}

</style>

<script type="text/javascript" src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.7.2.js"></script>

<script type="text/javascript">

$(function () {

$("#loginButton").click(function () {

var username = $("#username").val();

var password = $("#password").val();

if (username != "jhc\_admin" || password != "jhc123") {

alert("用户名或密码错误，登录失败。");

return false;

}

});

});

</script>

</head>

<body>

<div class="divright">

<div class="login">

<h2>书城管理员登录</h2>

<!-- 用户名：jhc\_admin，密码：jhc123 -->

<div class="login\_box">

<input type="text" id="username" required /><label>请输入管理员用户id</label>

</div>

<div class="login\_box">

<input type="password" id="password" required="required" /><label>请输入密码</label>

</div>

<!-- <a id="loginButton" href="javascript:void(0)"> -->

<a id="loginButton" href="manager.jsp">

登录

</a>

</div>

</div>

</body>

</html>

---------------------------------------------------

order\_manager.jsp文件：

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html>

<head>

<title>订单管理</title>

<%@include file="../common/head.jsp" %>

</head>

<body>

<div id="header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

<span class="wel\_word">订单管理系统</span>

<%@include file="../common/manager\_menu.jsp" %>

</div>

<div id="main">

<table>

<tr>

<td>编号</td>

<td>日期</td>

<td>金额</td>

<td>详情</td>

<td>发货</td>

</tr>

<c:forEach var="order" items="${requestScope.page.items}">

<tr>

<td>${order.orderId}</td>

<td>${order.createTime}</td>

<td>${order.price}</td>

<td><a href="OrderServlet?action=orderDetail&orderId=${order.orderId}">查看详情</a></td>

<td>

<c:if test="${order.status==0}">

<a href="manager/OrderServlet?action=sendOrder&orderId=${order.orderId}&pageNo=${requestScope.page.pageNo}&pageSize=${requestScope.page.pageSize}">

点击发货</a>

</c:if>

<c:if test="${order.status==1}">

<span>已发货</span>

</c:if>

<c:if test="${order.status==2}">

<span>已收货</span>

</c:if>

</td>

</tr>

</c:forEach>

</table><br/>

<%@include file="../common/page\_nav.jsp"%>

</div>

<%@include file="../common/footer.jsp" %>

</body>

</html>

---------------------------------------------------

order.jsp文件：

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html>

<head>

<title>订单详情</title>

<%@include file="../common/head.jsp" %>

<style type="text/css">

h1 {

text-align: center;

margin-top: 200px;

}

</style>

</head>

<body>

<div id="header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

<span class="wel\_word">我的订单</span>

<%@include file="../common/login\_success\_menu.jsp" %>

</div>

<div id="main">

<table>

<tr>

<td>日期</td>

<td>金额</td>

<td>状态</td>

<td>详情</td>

</tr>

<c:forEach var="order" items="${requestScope.orders}">

<tr>

<td>${order.createTime}</td>

<td>${order.price}</td>

<td>

<c:if test="${order.status==0}">

<span>未发货</span>

</c:if>

<c:if test="${order.status==1}">

<span>已发货</span>

</c:if>

<c:if test="${order.status==2}">

<span>已收货</span>

</c:if>

</td>

<td><a href="OrderServlet?action=orderDetail&orderId=${order.orderId}">查看详情</a></td>

</tr>

</c:forEach>

</table>

</div>

<%@include file="../common/footer.jsp" %>

</body>

</html>

---------------------------------------------------

order\_detail.jsp文件：

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html>

<head>

<title>购物车</title>

<%@include file="../common/head.jsp" %>

</head>

<body>

<div id="header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

<span class="wel\_word">订单详情</span>

<%@include file="../common/login\_success\_menu.jsp" %>

</div>

<div id="main">

<%--订单不存在--%>

<c:if test="${empty requestScope.orderItems}">

<table>

<tr>

<td>商品名称</td>

<td>数量</td>

<td>单价</td>

<td>总价</td>

<td>操作</td>

</tr>

<tr>

<th colspan="5"><a href="index.jsp">亲, 该订单不存在!</a></th>

</tr>

</table>

</c:if>

<%--订单存在--%>

<c:if test="${not empty requestScope.orderItems}">

<table>

<tr>

<td>商品名称</td>

<td>数量</td>

<td>单价</td>

<td>总价</td>

</tr>

<c:forEach var="item" items="${requestScope.orderItems}">

<tr>

<td>${item.name}</td>

<td>${item.count}</td>

<td>${item.price}</td>

<td>${item.totalPrice}</td>

</tr>

</c:forEach>

</table>

<div class="order\_info" align="center">

<span class="order\_span">总金额<span class="b\_price">${requestScope.order.price}</span>元</span><br/>

<span class="order\_span">订单编号<span class="b\_price">${requestScope.order.orderId}</span></span><br/>

<span class="order\_span">创建时间<span class="b\_price">${requestScope.order.createTime}</span></span><br/>

<span class="order\_span">订单状态<span class="b\_price">

<c:if test="${order.status==0}">

<span>未发货</span>

</c:if>

<c:if test="${order.status==1}">

<span>已发货</span>

<span class="order\_span"><a

href="OrderServlet?action=receiveOrder&orderId=${requestScope.order.orderId}">确认收货

</a></span>

</c:if>

<c:if test="${order.status==2}">

<span>已收货</span>

</c:if>

</span></span><br/>

</div>

</c:if>

</div>

<%@include file="../common/footer.jsp" %>

</body>

</html>

---------------------------------------------------

login.jsp文件：

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html>

<head>

<title>金职书城会员登录页面</title>

<%@include file="../common/head.jsp" %>

<script type="text/javascript">

<%--$(function () {--%>

<%-- $("#code\_img").click(function () {--%>

<%-- this.src = "${pageContext.request.contextPath}/kaptcha.jpg?t=" + Date.now()--%>

<%-- })--%>

<%--})--%>

</script>

</head>

<body>

<div id="login\_header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

</div>

<div class="login\_banner">

<div id="l\_content">

<span class="login\_word">欢迎登录</span>

</div>

<div id="content">

<div class="login\_form">

<div class="login\_box">

<div class="tit">

<h1>金职书城会员</h1>

<a href="pages/user/regist.jsp">立即注册</a>

</div>

<div class="msg\_cont">

<b></b>

<!--设置错误提示-->

<span class="errorMsg">

<%-- <%=request.getAttribute("msg") == null ? "请输入用户名和密码" : request.getAttribute("msg")%>--%>

${empty requestScope.msg ? "请输入用户名和密码" : requestScope.msg}

</span>

</div>

<div class="form">

<!--设置action和method属性-->

<form action="UserServlet" method="post">

<%--添加隐藏域--%>

<input type="hidden" name="action" value="login"/>

<label>用户名称：</label>

<input class="itxt" type="text" placeholder="请输入用户名" autocomplete="off" tabindex="1"

name="username"

<%--实现表单回显--%>

<%-- value="<%=request.getAttribute("username")==null?"":request.getAttribute("username")%>"--%>

value="${requestScope.username}" <%--EL表达式会将null以空串输出--%>

/>

<br/>

<br/>

<label>用户密码：</label>

<input class="itxt" type="password" placeholder="请输入密码" autocomplete="off" tabindex="1"

name="password"/>

<br/>

<br/>

<%-- <label>验证码：</label>--%>

<%-- <input class="itxt" type="text" name="code" style="width: 150px;" id="code"/>--%>

<%-- <img id="code\_img" alt="验证码" src="${pageContext.request.contextPath}/kaptcha.jpg"--%>

<%-- style="float: right;--%>

<%-- margin-right:--%>

<%-- 40px">--%>

<input type="submit" value="登录" id="sub\_btn"/>

</form>

</div>

</div>

</div>

</div>

</div>

<%@include file="../common/footer.jsp" %>

</body>

</html>

---------------------------------------------------

login\_success.jsp文件：

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html>

<head>

<title>金职书城会员注册页面</title>

<%-- 静态包含base标签,meta标签,css样式, jQuery文件--%>

<%@include file="../common/head.jsp"%>

<style type="text/css">

h1 {

text-align: center;

margin-top: 200px;

}

h1 a {

color: red;

}

</style>

</head>

<body>

<div id="header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

<%-- 静态包含div标签--%>

<%@include file="../common/login\_success\_menu.jsp" %>

</div>

<div id="main">

<h1>欢迎回来 <a href="index.jsp">转到主页</a></h1>

</div>

<%--静态包含页脚信息--%>

<%@include file="../common/footer.jsp"%>

</body>

</html>

---------------------------------------------------

regist.jsp文件：

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html lang="zh\_cn">

<head>

<title>金职书城会员注册页面</title>

<%@include file="../common/head.jsp" %>

<script type="text/javascript">

//页面加载完后

$(function () {

//为验证码图片绑定单击事件

$("#code\_img").click(function () {

//修改img的src属性, 让浏览器重新请求图片

//同时加个时间戳参数, 避免浏览器加载缓存

this.src = "${pageContext.request.contextPath}/kaptcha.jpg?t=" + Date.now()

})

//实现用户名的Ajax验证

$("#username").change(function () {

var param = "action=ajaxExistingUsername&" + $(this).serialize()

$.getJSON("http://localhost:8080/book/UserServlet", param, function (data) {

if (data.existingUsername) {

$(".errorMsg").text("用户名已存在!")

}else{

$(".errorMsg").empty()

}

})

})

//为注册按钮绑定单击事件

$("#sub\_btn").click(

function () {

// 验证用户名：必须由字母，数字下划线组成，并且长度为 5 到 12 位

let username = $("#username").val();

let usernamePat = /^\w{5,12}$/;

if (!usernamePat.test(username)) {

$("span.errorMsg").text("用户名不合法");

return false;

}

// 验证密码：必须由字母，数字下划线组成，并且长度为 5 到 12 位验证确认密码：和密码相同

let password = $("#password").val();

let repwd = $("#repwd").val();

let passwordPat = /^\w{5,12}$/;

if (password !== repwd || !passwordPat.test(password)) {

$("span.errorMsg").text("密码不合法");

return false;

}

// 邮箱验证：xxxxx@xxx.com

let email = $("#email").val();

let emailPat = /^([\w-])+@([\w-])+(.[\w-])+/;

if (!emailPat.test(email)) {

$("span.errorMsg").text("邮箱不合法");

return false;

}

// 验证码：现在只需要验证用户已输入。因为还没讲到服务器验证码生成。

let code = $("#code").val().trim();

if (code === "") {

$("span.errorMsg").text("验证码不合法");

return false;

}

//验证通过, 清空errorMsg, 返回true

$("span.errorMsg").text("");

return true;

}

)

})

</script>

<style type="text/css">

.login\_form {

height: 420px;

margin-top: 25px;

}

</style>

</head>

<body>

<div id="login\_header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

</div>

<div class="login\_banner">

<div id="l\_content">

<span class="login\_word">欢迎注册</span>

</div>

<div id="content">

<div class="login\_form">

<div class="login\_box">

<div class="tit">

<h1>注册金职书城会员</h1>

<%--回显错误信息--%>

<span class="errorMsg"><%=request.getAttribute("msg") == null ? "" : request.getAttribute("msg")%></span>

${requestScope.msg}

</div>

<div class="form">

<!--设置action和method属性-->

<form action="UserServlet" method="post">

<%--添加隐藏域--%>

<input type="hidden" name="action" value="regist">

<label>用户名称：</label>

<input class="itxt" type="text" placeholder="请输入用户名" autocomplete="off" tabindex="1"

name="username" id="username"

<%--回显用户名--%>

<%-- value="<%=request.getAttribute("username")==null?"":request.getAttribute("username")%>"--%>

value="${requestScope.username}"

/>

<br/>

<br/>

<label>用户密码：</label>

<input class="itxt" type="password" placeholder="请输入密码" autocomplete="off" tabindex="1"

name="password" id="password"/>

<br/>

<br/>

<label>确认密码：</label>

<input class="itxt" type="password" placeholder="确认密码" autocomplete="off" tabindex="1"

name="repwd" id="repwd"/>

<br/>

<br/>

<label>电子邮件：</label>

<input class="itxt" type="text" placeholder="请输入邮箱地址" autocomplete="off" tabindex="1"

name="email" id="email"

<%--回显邮箱--%>

<%-- value="<%=request.getAttribute("email")==null?"":request.getAttribute("email")%>"--%>

value="${requestScope.email}"

/>

<br/>

<br/>

<label>验证码：</label>

<input class="itxt" type="text" name="code" style="width: 150px;" id="code"/>

<img id="code\_img" alt="验证码" src="${pageContext.request.contextPath}/kaptcha.jpg" style="float:

right;

margin-right:

40px">

<br/>

<br/>

<input type="submit" value="注册" id="sub\_btn"/>

</form>

</div>

</div>

</div>

</div>

</div>

<%@include file="../common/footer.jsp" %>

</body>

</html>

---------------------------------------------------

regist\_success.jsp文件：

<%@ page contentType="text/html;charset=UTF-8" %>

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>金职书城注册页面</title>

<%@include file="../common/head.jsp"%>

<style type="text/css">

h1 {

text-align: center;

margin-top: 200px;

}

h1 a {

color: red;

}

</style>

</head>

<body>

<div id="header">

<img class="logo\_img" alt="" src="static/img/logo.gif">

<span class="wel\_word"></span>

<%@include file="../common/login\_success\_menu.jsp" %>

</div>

<div id="main">

<h1>注册成功! <a href="index.jsp">转到主页</a></h1>

</div>

<%@include file="../common/footer.jsp"%>

</body>

</html>

---------------------------------------------------

style.css文件：

@CHARSET "UTF-8";

body {

overflow: hidden;

}

\* {

margin: 0;

font-family:"Microsoft Yahei";

color: #666;

}

div{

margin: auto;

margin-bottom: 10px;

margin-top: 10px;

}

#header {

height: 82px;

width: 1200px;

}

#main {

height: 460px;

width: 1200px;

border: 1px #0456A8 solid;

overflow: auto;

}

#bottom {

height: 30px;

width: 1200px;

text-align: center;

}

#book{

width: 100%;

height: 90%;

margin: auto;

}

.b\_list{

height:300px;

width:250px;

margin: 20px;

float: left;

margin-top:0px;

margin-bottom:0px;

border: 1px #e3e3e3 solid;

}

#page\_nav{

width: 100%;

height: 10px;

margin: auto;

text-align: center;

}

#pn\_input {

width: 30px;

text-align: center;

}

.img\_div{

height: 150px;

text-align: center;

}

.book\_img {

height:150px;

width:150px;

}

.book\_info {

text-align: center;

}

.book\_info div{

height: 10px;

width: 300px;

text-align: left;

}

.wel\_word{

color: #0456A8;

font-size: 60px;

float: left;

}

.logo\_img{

float: left;

}

#header div a {

text-decoration: none;

font-size: 20px;

}

#header div{

float: right;

margin-top: 55px;

}

.book\_cond{

margin-left: 500px;

}

.book\_cond input{

width: 50px;

text-align: center;

}

/\*登录页面CSS样式 \*/

#login\_header{

height: 82px;

width: 1200px;

}

.login\_banner{

height:475px;

background-color: #0456A8;

}

.login\_form{

height:310px;

width:406px;

float: right;

margin-right:50px;

margin-top: 50px;

background-color: #fff;

}

#content {

height: 475px;

width: 1200px;

}

.login\_box{

margin: 20px;

height: 260px;

width: 366px;

}

h1 {

font-size: 20px;

}

.msg\_cont{

background: none repeat scroll 0 0 #fff6d2;

border: 1px solid #ffe57d;

color: #666;

height: 18px;

line-height: 18px;

padding: 3px 10px 3px 40px;

position: relative;

border: none;

}

.msg\_cont b {

background: url("../img/pwd-icons-new.png") no-repeat scroll -104px -22px rgba(0, 0, 0, 0);

display: block;

height: 17px;

left: 10px;

margin-top: -8px;

overflow: hidden;

position: absolute;

top: 50%;

width: 16px;

}

.form .itxt {

border: 0 none;

float: none;

font-family: "宋体";

font-size: 14px;

height: 18px;

line-height: 18px;

overflow: hidden;

padding: 10px 0 10px 10px;

width: 220px;

border: 1px #e3e3e3 solid;

}

#sub\_btn{

background-color: #39987c;

border: none;

color: #fff;

width: 360px;

height: 40px;

}

#l\_content {

float: left;

margin-top: 150px;

margin-left: 300px;

}

#l\_content span {

font-size: 60px;

color: white;

}

.tit h1 {

float: left;

margin-top: 5px;

}

.tit a {

float: right;

margin-left: 10px;

margin-top: 10px;

color: red;

text-decoration: none;

}

.tit .errorMsg {

float: right;

margin-left: 10px;

margin-top: 10px;

color: red;

}

.tit {

height: 30px;

}

/\*购物车\*/

#main table{

margin: auto;

margin-top: 80px;

border-collapse: collapse;

}

#main table td{

width: 120px;

text-align:center;

border-bottom: 1px #e3e3e3 solid;

padding: 10px;

}

.cart\_info{

width: 700px;

text-align: right;

}

.cart\_span {

margin-left: 20px;

}

.cart\_span span{

color: red;

font-size: 20px;

margin: 10px;

}

.cart\_span a , td a{

font-size: 20px;

color: blue;

}

#header div span {

margin: 10px;

}

#header div .um\_span{

color: red;

font-size: 25px;

margin: 10px;

}

#header div a {

color: blue;

}

---------------------------------------------------

web.xml文件：

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app\_4\_0.xsd"

version="4.0">

<servlet>

<servlet-name>KaptchaServlet</servlet-name>

<servlet-class>com.google.code.kaptcha.servlet.KaptchaServlet</servlet-class>

<init-param>

<param-name>kaptcha.textproducer.char.length</param-name>

<param-value>4</param-value>

</init-param>

</servlet>

<servlet-mapping>

<servlet-name>KaptchaServlet</servlet-name>

<url-pattern>/kaptcha.jpg</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>SearchBookByName</servlet-name>

<servlet-class>com.servlet.SearchBookByName</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>SearchBookByName</servlet-name>

<url-pattern>/searchBookByName</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>AdminLoginServlet</servlet-name>

<servlet-class>com.servlet.AdminLoginServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>AdminLoginServlet</servlet-name>

<url-pattern>/admin</url-pattern>

</servlet-mapping>

<error-page>

<error-code>500</error-code>

<location>/pages/error/error500.jsp</location>

</error-page>

<error-page>

<error-code>404</error-code>

<location>/pages/error/error404.jsp</location>

</error-page>

</web-app>