已设置一页50行

**目 录**

[LoginFilter.java 2](#_Toc104489462)

[MailUtils.java 2](#_Toc104489463)

[UserDao.java 4](#_Toc104489464)

[User.java 6](#_Toc104489465)

[UserService.java 8](#_Toc104489466)

[UserServlet.java 11](#_Toc104489467)

[PageBean.java 13](#_Toc104489468)

[Expression.java 14](#_Toc104489469)

[OrderDao.java 15](#_Toc104489470)

[Order.java 15](#_Toc104489471)

[OrderItem.java 16](#_Toc104489472)

[OrderService.java 20](#_Toc104489473)

[OrderServlet.java 22](#_Toc104489474)

[PaymentUtil.java 23](#_Toc104489475)

[CategoryDao.java 28](#_Toc104489476)

[Category.java 31](#_Toc104489477)

[CategoryService.java 32](#_Toc104489478)

[CategoryServlet.java 34](#_Toc104489479)

[CartItemDao.java 35](#_Toc104489480)

[CartItem.java 37](#_Toc104489481)

[CartItemService.java 39](#_Toc104489482)

[CartItemServlet.java 40](#_Toc104489483)

[BookDao.java 41](#_Toc104489484)

[Book.java 45](#_Toc104489485)

[BookServlet.java 48](#_Toc104489486)

[BookService.java 49](#_Toc104489487)

# LoginFilter.java

package cn.itcast.goods.web.filter;

import javax.servlet.\*;

import javax.servlet.http.HttpServletRequest;

import java.io.IOException;

public class LoginFilter implements Filter {

    public void destroy() {

    }

    public void doFilter(ServletRequest request, ServletResponse response,

            FilterChain chain) throws IOException, ServletException {

        HttpServletRequest req = (HttpServletRequest) request;

        Object user = req.getSession().getAttribute("sessionUser");

        if(user == null) {

            req.setAttribute("code", "error");//为了显示X图片

            req.setAttribute("msg", "您还没有登录，不能访问本资源");

            req.getRequestDispatcher("/jsps/msg.jsp").forward(req, response);

        } else {

            chain.doFilter(request, response);//放行

        }

    }

    public void init(FilterConfig fConfig) throws ServletException {

    }

}

# MailUtils.java

package cn.itcast.goods.util;

import javax.mail.\*;

import javax.mail.internet.InternetAddress;

import javax.mail.internet.MimeMessage;

import java.util.Properties;

public final class MailUtils {

    private static final String USER = "770485230@qq.com"; // 发件人称号，同邮箱地址

    private static final String PASSWORD = "iapkcybydhfcbffd"; // 如果是qq邮箱可以使户端授权码，或者登录密码

    /\* 发送验证信息的邮件 \*/

    public static boolean sendMail(String to, String text, String title){

        try {

            final Properties props = new Properties();

            props.put("mail.smtp.auth", "true");

            props.put("mail.smtp.host", "smtp.qq.com");

            // 发件人的账号

            props.put("mail.user", USER);

            //发件人的密码

            props.put("mail.password", PASSWORD);

            // 构建授权信息，用于进行SMTP进行身份验证

            Authenticator authenticator = new Authenticator() {

                @Override

                protected PasswordAuthentication getPasswordAuthentication() {

                    // 用户名、密码

                    String userName = props.getProperty("mail.user");

                    String password = props.getProperty("mail.password");

                    return new PasswordAuthentication(userName, password);

                }

            };

            // 使用环境属性和授权信息，创建邮件会话

            Session mailSession = Session.getInstance(props, authenticator);

            // 创建邮件消息

            MimeMessage message = new MimeMessage(mailSession);

            // 设置发件人

            String username = props.getProperty("mail.user");

            InternetAddress form = new InternetAddress(username);

            message.setFrom(form);

            // 设置收件人

            InternetAddress toAddress = new InternetAddress(to);

            message.setRecipient(Message.RecipientType.TO, toAddress);

            // 设置邮件标题

            message.setSubject(title);

            // 设置邮件的内容体

            message.setContent(text, "text/html;charset=UTF-8");

            // 发送邮件

            Transport.send(message);

            return true;

        }catch (Exception e){

            e.printStackTrace();

        }

        return false;

    }

    public static void main(String[] args) throws Exception { // 做测试用

        MailUtils.sendMail("1326477275@qq.com","你好，这是一封测试邮件，无需回复。","测试邮件");

        System.out.println("发送成功");

    }

}

# UserDao.java

package cn.itcast.goods.user.dao;

import cn.itcast.goods.user.domain.User;

import cn.itcast.jdbc.TxQueryRunner;

import org.apache.commons.dbutils.QueryRunner;

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

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

import java.sql.SQLException;

/\*\*

 \* 用户模块持久层

 \*/

public class UserDao {

    private QueryRunner qr = new TxQueryRunner();

    /\*\*

     \* 按uid和password查询

     \*/

    public boolean findByUidAndPassword(String uid, String password) throws SQLException {

        String sql = "select count(\*) from t\_user where uid=? and loginpass=?";

        Number number = (Number)qr.query(sql, new ScalarHandler(), uid, password);

        return number.intValue() > 0;

    }

    /\*\*

     \* 修改密码

     \*/

    public void updatePassword(String uid, String password) throws SQLException {

        String sql = "update t\_user set loginpass=? where uid=?";

        qr.update(sql, password, uid);

    }

    /\*\*

     \* 按用户名和密码查询

     \*/

    public User findByLoginnameAndLoginpass(String loginname, String loginpass) throws SQLException {

        String sql = "select \* from t\_user where loginname=? and loginpass=?";

        return qr.query(sql, new BeanHandler<User>(User.class), loginname, loginpass);

    }

    /\*\*

     \* 通过激活码查询用户

     \*/

    public User findByCode(String code) throws SQLException {

        String sql = "select \* from t\_user where activationCode=?";

        return qr.query(sql, new BeanHandler<User>(User.class), code);

    }

    /\*\*

     \* 修改用户状态

     \*/

    public void updateStatus(String uid, boolean status) throws SQLException {

        String sql = "update t\_user set status=? where uid=?";

        qr.update(sql, status, uid);

    }

    /\*\*

     \* 校验用户名是否注册

     \*/

    public boolean ajaxValidateLoginname(String loginname) throws SQLException {

        String sql = "select count(1) from t\_user where loginname=?";

        Number number = (Number)qr.query(sql, new ScalarHandler(), loginname);

        return number.intValue() == 0;

    }

    /\*\*

     \* 校验Email是否注册

     \*/

    public boolean ajaxValidateEmail(String email) throws SQLException {

        String sql = "select count(1) from t\_user where email=?";

        Number number = (Number)qr.query(sql, new ScalarHandler(), email);

        return number.intValue() == 0;

    }

    /\*\*

     \* 添加用户

     \*/

    public void add(User user) throws SQLException {

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

        Object[] params = {user.getUid(), user.getLoginname(), user.getLoginpass(),

                user.getEmail(), user.isStatus(), user.getActivationCode()};

        for (Object param : params) {

            System.out.println(param);

        }

        qr.update(sql, params);

    }

}

# User.java

package cn.itcast.goods.user.domain;

/\*\*

 \* 用户模块实体类

 \*/

public class User {

    // 对应数据库表

    private String uid;//主键

    private String loginname;//登录名

    private String loginpass;//登录密码

    private String email;//邮箱

    private boolean status;//状态，true表示已激活，或者未激活

    private String activationCode;//激活码，它是唯一值！即每个用户的激活码是不同的！

    // 注册表单

    private String reloginpass;//确认密码

    private String verifyCode;//验证码

    // 修改密码表单

    private String newpass;//新密码

    public String getReloginpass() {

        return reloginpass;

    }

    public void setReloginpass(String reloginpass) {

        this.reloginpass = reloginpass;

    }

    public String getVerifyCode() {

        return verifyCode;

    }

    public void setVerifyCode(String verifyCode) {

        this.verifyCode = verifyCode;

    }

    public String getUid() {

        return uid;

    }

    public void setUid(String uid) {

        this.uid = uid;

    }

    public String getLoginname() {

        return loginname;

    }

    public void setLoginname(String loginname) {

        this.loginname = loginname;

    }

    public String getLoginpass() {

        return loginpass;

    }

    public void setLoginpass(String loginpass) {

        this.loginpass = loginpass;

    }

    public String getEmail() {

        return email;

    }

    public void setEmail(String email) {

        this.email = email;

    }

    public boolean isStatus() {

        return status;

    }

    public void setStatus(boolean status) {

        this.status = status;

    }

    public String getActivationCode() {

        return activationCode;

    }

    public void setActivationCode(String activationCode) {

        this.activationCode = activationCode;

    }

    public String getNewpass() {

        return newpass;

    }

    public void setNewpass(String newpass) {

        this.newpass = newpass;

    }

    @Override

    public String toString() {

        return "User [uid=" + uid + ", loginname=" + loginname + ", loginpass="

                + loginpass + ", email=" + email + ", status=" + status

                + ", activationCode=" + activationCode + ", reloginpass="

                + reloginpass + ", verifyCode=" + verifyCode + ", newpass="

                + newpass + "]";

    }

}

# UserService.java

package cn.itcast.goods.user.service;

import cn.itcast.commons.CommonUtils;

import cn.itcast.goods.user.dao.UserDao;

import cn.itcast.goods.user.domain.User;

import cn.itcast.goods.user.service.exception.UserException;

import cn.itcast.mail.Mail;

import cn.itcast.mail.MailUtils;

import javax.mail.MessagingException;

import javax.mail.Session;

import java.io.IOException;

import java.sql.SQLException;

import java.text.MessageFormat;

import java.util.Properties;

/\*\*

 \* 用户模块业务层

 \* @author qdmmy6

 \*

 \*/

public class UserService {

    private UserDao userDao = new UserDao();

    /\*\*

     \* 修改密码

     \* @param uid

     \* @param newPass

     \* @param oldPass

     \* @throws UserException

     \*/

    public void updatePassword(String uid, String newPass, String oldPass) throws UserException {

        try {

            boolean bool = userDao.findByUidAndPassword(uid, oldPass);

            if(!bool) {//如果老密码错误

                throw new UserException("老密码错误！");

            }

            userDao.updatePassword(uid, newPass);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 登录功能

     \*/

    public User login(User user) {

        try {

            return userDao.findByLoginnameAndLoginpass(user.getLoginname(), user.getLoginpass());

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 激活功能

     \*/

    public void activatioin(String code) throws UserException {

        try {

            User user = userDao.findByCode(code);

            if(user == null) throw new UserException("无效的激活码！");

            if(user.isStatus()) throw new UserException("您已经激活过了，不要二次激活！");

            userDao.updateStatus(user.getUid(), true);//修改状态

        } catch(SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 用户名注册校验

     \*/

    public boolean ajaxValidateLoginname(String loginname) {

        try {

            return userDao.ajaxValidateLoginname(loginname);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* Email校验

     \*/

    public boolean ajaxValidateEmail(String email) {

        try {

            return userDao.ajaxValidateEmail(email);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 注册功能

     \*/

    public void regist(User user) {

        user.setUid(CommonUtils.uuid());

        user.setStatus(false);

        user.setActivationCode(CommonUtils.uuid() + CommonUtils.uuid());

        try {

            userDao.add(user);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

        Properties prop = new Properties();

        try {

            prop.load(this.getClass().getClassLoader().getResourceAsStream("email\_template.properties"));

        } catch (IOException e1) {

            throw new RuntimeException(e1);

        }

        String host = prop.getProperty("host");//服务器主机名

        String name = prop.getProperty("username");//登录名

        String pass = prop.getProperty("password");//登录密码

        Session session = MailUtils.createSession(host, name, pass);

        String from = prop.getProperty("from");

        String to = user.getEmail();

        String subject = prop.getProperty("subject");

        String str=null;

        try{

             str=new String(prop.getProperty("content").getBytes("iso-8859-1"),"utf-8");

        }catch (Exception e){

            e.printStackTrace();

        }

        String content=MessageFormat.format(str, user.getActivationCode());

        Mail mail = new Mail(from, to, subject, content);

        try {

            MailUtils.send(session, mail);

        } catch (MessagingException e) {

            throw new RuntimeException(e);

        } catch (IOException e) {

            throw new RuntimeException(e);

        }

    }

}

# UserServlet.java

package cn.itcast.goods.user.web.servlet;

import cn.itcast.commons.CommonUtils;

import cn.itcast.goods.user.domain.User;

import cn.itcast.goods.user.service.UserService;

import cn.itcast.goods.user.service.exception.UserException;

import cn.itcast.servlet.BaseServlet;

import javax.servlet.ServletException;

import javax.servlet.http.Cookie;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import java.io.IOException;

import java.net.URLEncoder;

import java.util.HashMap;

import java.util.Map;

/\*\*

 \* 用户模块WEB层

 \* @author qdmmy6

 \*

 \*/

public class UserServlet extends BaseServlet {

    private UserService userService = new UserService();

    /\*

     \* 注册校验

     \*/

    private Map<String,String> validateRegist(User formUser, HttpSession session) {

        Map<String,String> errors = new HashMap<String,String>();

        String loginname = formUser.getLoginname();

        if(loginname == null || loginname.trim().isEmpty()) {

            errors.put("loginname", "用户名不能为空！");

        } else if(loginname.length() < 3 || loginname.length() > 20) {

            errors.put("loginname", "用户名长度必须在3~20之间！");

        } else if(!userService.ajaxValidateLoginname(loginname)) {

            errors.put("loginname", "用户名已被注册！");

        }

        String loginpass = formUser.getLoginpass();

        if(loginpass == null || loginpass.trim().isEmpty()) {

            errors.put("loginpass", "密码不能为空！");

        } else if(loginpass.length() < 3 || loginpass.length() > 20) {

            errors.put("loginpass", "密码长度必须在3~20之间！");

        }

        String reloginpass = formUser.getReloginpass();

        if(reloginpass == null || reloginpass.trim().isEmpty()) {

            errors.put("reloginpass", "确认密码不能为空！");

        } else if(!reloginpass.equals(loginpass)) {

            errors.put("reloginpass", "两次输入不一致！");

        }

        String email = formUser.getEmail();

        if(email == null || email.trim().isEmpty()) {

            errors.put("email", "Email不能为空！");

        } else if(!email.matches("^([a-zA-Z0-9\_-])+@([a-zA-Z0-9\_-])+((\\.[a-zA-Z0-9\_-]{2,3}){1,2})$")) {

            errors.put("email", "Email格式错误！");

        } else if(!userService.ajaxValidateEmail(email)) {

            errors.put("email", "Email已被注册！");

        }

        String verifyCode = formUser.getVerifyCode();

        String vcode = (String) session.getAttribute("vCode");

        if(verifyCode == null || verifyCode.trim().isEmpty()) {

            errors.put("verifyCode", "验证码不能为空！");

        } else if(!verifyCode.equalsIgnoreCase(vcode)) {

            errors.put("verifyCode", "验证码错误！");

        }

        return errors;

    }

    /\*

     \* 登录校验方法

     \*/

    private Map<String,String> validateLogin(User formUser, HttpSession session) {

        Map<String,String> errors = new HashMap<String,String>();

        String verifyCode = formUser.getVerifyCode();

        String vcode = (String) session.getAttribute("vCode");

        if(verifyCode == null || verifyCode.trim().isEmpty()) {

            errors.put("verifyCode", "验证码不能为空！");

        } else if(!verifyCode.equalsIgnoreCase(vcode)) {

            errors.put("verifyCode", "验证码错误！");

        }

        return errors;

    }

}

# PageBean.java

package cn.itcast.goods.pager;

import java.util.List;

/\*\*

 \* 分页Bean，它会在各层之间传递！

 \*/

public class PageBean<T> {

    private int pc;//当前页码

    private int tr;//总记录数

    private int ps;//每页记录数

    private String url;//请求路径和参数，例如：/BookServlet?method=findXXX&cid=1&bname=2

    private List<T> beanList;

    // 计算总页数

    public int getTp() {

        int tp = tr / ps;

        return tr % ps == 0 ? tp : tp + 1;

    }

    public int getPc() {

        return pc;

    }

    public void setPc(int pc) {

        this.pc = pc;

    }

    public int getTr() {

        return tr;

    }

    public void setTr(int tr) {

        this.tr = tr;

    }

    public int getPs() {

        return ps;

    }

    public void setPs(int ps) {

        this.ps = ps;

    }

    public String getUrl() {

        return url;

    }

    public void setUrl(String url) {

        this.url = url;

    }

    public List<T> getBeanList() {

        return beanList;

    }

    public void setBeanList(List<T> beanList) {

        this.beanList = beanList;

    }

}

# Expression.java

package cn.itcast.goods.pager;

public class Expression {

    private String name;

    private String operator;

    private String value;

    public String getName() {

        return name;

    }

    public void setName(String name) {

        this.name = name;

    }

    public String getOperator() {

        return operator;

    }

    public void setOperator(String operator) {

        this.operator = operator;

    }

    public String getValue() {

        return value;

    }

    public void setValue(String value) {

        this.value = value;

    }

    @Override

    public String toString() {

        return "Expression [name=" + name + ", operator=" + operator

                + ", value=" + value + "]";

    }

    public Expression() {

        super();

        // TODO Auto-generated constructor stub

    }

    public Expression(String name, String operator, String value) {

        super();

        this.name = name;

        this.operator = operator;

        this.value = value;

    }

}

# OrderDao.java

# Order.java

package cn.itcast.goods.order.domain;

import cn.itcast.goods.user.domain.User;

import java.util.List;

public class Order {

    private String oid;//主键

    private String ordertime;//下单时间

    private double total;//总计

    private int status;//订单状态：1未付款, 2已付款但未发货, 3已发货未确认收货, 4确认收货了交易成功, 5已取消(只有未付款才能取消)

    private String address;//收货地址

    private User owner;//订单的所有者

    private List<OrderItem> orderItemList;

    public List<OrderItem> getOrderItemList() {

        return orderItemList;

    }

    public void setOrderItemList(List<OrderItem> orderItemList) {

        this.orderItemList = orderItemList;

    }

    public String getOid() {

        return oid;

    }

    public void setOid(String oid) {

        this.oid = oid;

    }

    public String getOrdertime() {

        return ordertime;

    }

    public void setOrdertime(String ordertime) {

        this.ordertime = ordertime;

    }

    public double getTotal() {

        return total;

    }

    public void setTotal(double total) {

        this.total = total;

    }

    public int getStatus() {

        return status;

    }

    public void setStatus(int status) {

        this.status = status;

    }

    public String getAddress() {

        return address;

    }

    public void setAddress(String address) {

        this.address = address;

    }

    public User getOwner() {

        return owner;

    }

    public void setOwner(User owner) {

        this.owner = owner;

    }

}

# OrderItem.java

package cn.itcast.goods.order.dao;

import cn.itcast.commons.CommonUtils;

import cn.itcast.goods.book.domain.Book;

import cn.itcast.goods.order.domain.Order;

import cn.itcast.goods.order.domain.OrderItem;

import cn.itcast.goods.pager.Expression;

import cn.itcast.goods.pager.PageBean;

import cn.itcast.goods.pager.PageConstants;

import cn.itcast.jdbc.TxQueryRunner;

import org.apache.commons.dbutils.QueryRunner;

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

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

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

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

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

public class OrderDao {

    private QueryRunner qr = new TxQueryRunner();

    /\*\*

     \* 查询订单状态

     \*/

    public int findStatus(String oid) throws SQLException {

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

        Number number = (Number)qr.query(sql, new ScalarHandler(), oid);

        return number.intValue();

    }

    /\*\*

     \* 修改订单状态

     \*/

    public void updateStatus(String oid, int status) throws SQLException {

        String sql = "update t\_order set status=? where oid=?";

        qr.update(sql, status, oid);

    }

    /\*\*

     \* 加载订单

     \*/

    public Order load(String oid) throws SQLException {

        String sql = "select \* from t\_order where oid=?";

        Order order = qr.query(sql, new BeanHandler<Order>(Order.class), oid);

        loadOrderItem(order);//为当前订单加载它的所有订单条目

        return order;

    }

    /\*\*

     \* 生成订单

     \*/

    public void add(Order order) throws SQLException {

        String sql = "insert into t\_order values(?,?,?,?,?,?)";

        Object[] params = {order.getOid(), order.getOrdertime(),

                order.getTotal(),order.getStatus(),order.getAddress(),

                order.getOwner().getUid()};

        qr.update(sql, params);

        sql = "insert into t\_orderitem values(?,?,?,?,?,?,?,?)";

        int len = order.getOrderItemList().size();

        Object[][] objs = new Object[len][];

        for(int i = 0; i < len; i++){

            OrderItem item = order.getOrderItemList().get(i);

            objs[i] = new Object[]{item.getOrderItemId(),item.getQuantity(),

                    item.getSubtotal(),item.getBook().getBid(),

                    item.getBook().getBname(),item.getBook().getCurrPrice(),

                    item.getBook().getImage\_b(),order.getOid()};

        }

        qr.batch(sql, objs);

    }

    /\*\*

     \* 按用户查询订单

     \*/

    public PageBean<Order> findByUser(String uid, int pc) throws SQLException {

        List<Expression> exprList = new ArrayList<Expression>();

        exprList.add(new Expression("uid", "=", uid));

        return findByCriteria(exprList, pc);

    }

    /\*\*

     \* 查询所有

     \*/

    public PageBean<Order> findAll(int pc) throws SQLException {

        List<Expression> exprList = new ArrayList<Expression>();

        return findByCriteria(exprList, pc);

    }

    /\*\*

     \* 按状态查询

     \*/

    public PageBean<Order> findByStatus(int status, int pc) throws SQLException {

        List<Expression> exprList = new ArrayList<Expression>();

        exprList.add(new Expression("status", "=", status + ""));

        return findByCriteria(exprList, pc);

    }

    private PageBean<Order> findByCriteria(List<Expression> exprList, int pc) throws SQLException {

        int ps = PageConstants.ORDER\_PAGE\_SIZE;//每页记录数

        StringBuilder whereSql = new StringBuilder(" where 1=1");

        List<Object> params = new ArrayList<Object>();//SQL中有问号，它是对应问号的值

        for(Expression expr : exprList) {

            whereSql.append(" and ").append(expr.getName())

                .append(" ").append(expr.getOperator()).append(" ");

            // where 1=1 and bid = ?

            if(!expr.getOperator().equals("is null")) {

                whereSql.append("?");

                params.add(expr.getValue());

            }

        }

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

        Number number = (Number)qr.query(sql, new ScalarHandler(), params.toArray());

        int tr = number.intValue();//得到了总记录数

        sql = "select \* from t\_order" + whereSql + " order by ordertime desc limit ?,?";

        params.add((pc-1) \* ps);//当前页首行记录的下标

        params.add(ps);//一共查询几行，就是每页记录数

        List<Order> beanList = qr.query(sql, new BeanListHandler<Order>(Order.class),

                params.toArray());

        for(Order order : beanList) {

            loadOrderItem(order);

        }

        PageBean<Order> pb = new PageBean<Order>();

        pb.setBeanList(beanList);

        pb.setPc(pc);

        pb.setPs(ps);

        pb.setTr(tr);

        return pb;

    }

    private void loadOrderItem(Order order) throws SQLException {

        String sql = "select \* from t\_orderitem where oid=?";

        List<Map<String,Object>> mapList = qr.query(sql, new MapListHandler(), order.getOid());

        List<OrderItem> orderItemList = toOrderItemList(mapList);

        order.setOrderItemList(orderItemList);

    }

    private List<OrderItem> toOrderItemList(List<Map<String, Object>> mapList) {

        List<OrderItem> orderItemList = new ArrayList<OrderItem>();

        for(Map<String,Object> map : mapList) {

            OrderItem orderItem = toOrderItem(map);

            orderItemList.add(orderItem);

        }

        return orderItemList;

    }

    private OrderItem toOrderItem(Map<String, Object> map) {

        OrderItem orderItem = CommonUtils.toBean(map, OrderItem.class);

        Book book = CommonUtils.toBean(map, Book.class);

        orderItem.setBook(book);

        return orderItem;

    }

}

# OrderService.java

package cn.itcast.goods.order.service;

import cn.itcast.goods.order.dao.OrderDao;

import cn.itcast.goods.order.domain.Order;

import cn.itcast.goods.pager.PageBean;

import cn.itcast.jdbc.JdbcUtils;

import java.sql.SQLException;

public class OrderService {

    private OrderDao orderDao = new OrderDao();

    /\*\*

     \* 修改订单状态

     \*/

    public void updateStatus(String oid, int status) {

        try {

            orderDao.updateStatus(oid, status);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 查询订单状态

     \*/

    public int findStatus(String oid) {

        try {

            return orderDao.findStatus(oid);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 加载订单

     \*/

    public Order load(String oid) {

        try {

            JdbcUtils.beginTransaction();

            Order order = orderDao.load(oid);

            JdbcUtils.commitTransaction();

            return order;

        } catch (SQLException e) {

            try {

                JdbcUtils.rollbackTransaction();

            } catch (SQLException e1) {}

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 生成订单

     \*/

    public void createOrder(Order order) {

        try {

            JdbcUtils.beginTransaction();

            orderDao.add(order);

            JdbcUtils.commitTransaction();

        } catch (SQLException e) {

            try {

                JdbcUtils.rollbackTransaction();

            } catch (SQLException e1) {}

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 我的订单

     \*/

    public PageBean<Order> myOrders(String uid, int pc) {

        try {

            JdbcUtils.beginTransaction();

            PageBean<Order> pb = orderDao.findByUser(uid, pc);

            JdbcUtils.commitTransaction();

            return pb;

        } catch (SQLException e) {

            try {

                JdbcUtils.rollbackTransaction();

            } catch (SQLException e1) {}

            throw new RuntimeException(e);

        }

    }

}

# OrderServlet.java

package cn.itcast.goods.order.web.servlet;

import cn.itcast.commons.CommonUtils;

import cn.itcast.goods.cart.domain.CartItem;

import cn.itcast.goods.cart.service.CartItemService;

import cn.itcast.goods.order.domain.Order;

import cn.itcast.goods.order.domain.OrderItem;

import cn.itcast.goods.order.service.OrderService;

import cn.itcast.goods.pager.PageBean;

import cn.itcast.goods.user.domain.User;

import cn.itcast.servlet.BaseServlet;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.IOException;

import java.math.BigDecimal;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

import java.util.Properties;

public class OrderServlet extends BaseServlet {

    private OrderService orderService = new OrderService();

    private CartItemService cartItemService = new CartItemService();

    /\*\*

     \* 获取当前页码

     \*/

    private int getPc(HttpServletRequest req) {

        int pc = 1;

        String param = req.getParameter("pc");

        if(param != null && !param.trim().isEmpty()) {

            try {

                pc = Integer.parseInt(param);

            } catch(RuntimeException e) {}

        }

        return pc;

    }

    /\*\*

     \* 截取url，页面中的分页导航中需要使用它做为超链接的目标！

     \*/

    private String getUrl(HttpServletRequest req) {

        String url = req.getRequestURI() + "?" + req.getQueryString();

        int index = url.lastIndexOf("&pc=");

        if(index != -1) {

            url = url.substring(0, index);

        }

        return url;

    }

    /\*\*

     \* 加载订单

     \* @param req

     \* @param resp

     \* @return

     \* @throws ServletException

     \* @throws IOException

     \*/

    public String load(HttpServletRequest req, HttpServletResponse resp)

            throws ServletException, IOException {

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

        Order order = orderService.load(oid);

        req.setAttribute("order", order);

        String btn = req.getParameter("btn");//btn说明了用户点击哪个超链接来访问本方法的

        req.setAttribute("btn", btn);

        return "/jsps/order/desc.jsp";

    }

}

# PaymentUtil.java

package cn.itcast.goods.order.web.servlet;

import java.io.UnsupportedEncodingException;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

import java.util.Arrays;

public class PaymentUtil {

    private static String encodingCharset = "UTF-8";

    /\*\*

     \* 生成hmac方法

     \*

     \* @param p0\_Cmd 业务类型

     \* @param p1\_MerId 商户编号

     \* @param p2\_Order 商户订单号

     \* @param p3\_Amt 支付金额

     \* @param p4\_Cur 交易币种

     \* @param p5\_Pid 商品名称

     \* @param p6\_Pcat 商品种类

     \* @param p7\_Pdesc 商品描述

     \* @param p8\_Url 商户接收支付成功数据的地址

     \* @param p9\_SAF 送货地址

     \* @param pa\_MP 商户扩展信息

     \* @param pd\_FrpId 银行编码

     \* @param pr\_NeedResponse 应答机制

     \* @param keyValue 商户密钥

     \* @return

     \*/

    public static String buildHmac(String p0\_Cmd,String p1\_MerId,

            String p2\_Order, String p3\_Amt, String p4\_Cur,String p5\_Pid, String p6\_Pcat,

            String p7\_Pdesc,String p8\_Url, String p9\_SAF,String pa\_MP,String pd\_FrpId,

            String pr\_NeedResponse,String keyValue) {

        StringBuilder sValue = new StringBuilder();

        // 业务类型

        sValue.append(p0\_Cmd);

        // 商户编号

        sValue.append(p1\_MerId);

        // 商户订单号

        sValue.append(p2\_Order);

        // 支付金额

        sValue.append(p3\_Amt);

        // 交易币种

        sValue.append(p4\_Cur);

        // 商品名称

        sValue.append(p5\_Pid);

        // 商品种类

        sValue.append(p6\_Pcat);

        // 商品描述

        sValue.append(p7\_Pdesc);

        // 商户接收支付成功数据的地址

        sValue.append(p8\_Url);

        // 送货地址

        sValue.append(p9\_SAF);

        // 商户扩展信息

        sValue.append(pa\_MP);

        // 银行编码

        sValue.append(pd\_FrpId);

        // 应答机制

        sValue.append(pr\_NeedResponse);

        return PaymentUtil.hmacSign(sValue.toString(), keyValue);

    }

    /\*\*

     \* 返回校验hmac方法

     \*

     \* @param hmac 支付网关发来的加密验证码

     \* @param p1\_MerId 商户编号

     \* @param r0\_Cmd 业务类型

     \* @param r1\_Code 支付结果

     \* @param r2\_TrxId 易宝支付交易流水号

     \* @param r3\_Amt 支付金额

     \* @param r4\_Cur 交易币种

     \* @param r5\_Pid 商品名称

     \* @param r6\_Order 商户订单号

     \* @param r7\_Uid 易宝支付会员ID

     \* @param r8\_MP 商户扩展信息

     \* @param r9\_BType 交易结果返回类型

     \* @param keyValue 密钥

     \* @return

     \*/

    public static boolean verifyCallback(String hmac, String p1\_MerId,

            String r0\_Cmd, String r1\_Code, String r2\_TrxId, String r3\_Amt,

            String r4\_Cur, String r5\_Pid, String r6\_Order, String r7\_Uid,

            String r8\_MP, String r9\_BType, String keyValue) {

        StringBuilder sValue = new StringBuilder();

        // 商户编号

        sValue.append(p1\_MerId);

        // 业务类型

        sValue.append(r0\_Cmd);

        // 支付结果

        sValue.append(r1\_Code);

        // 易宝支付交易流水号

        sValue.append(r2\_TrxId);

        // 支付金额

        sValue.append(r3\_Amt);

        // 交易币种

        sValue.append(r4\_Cur);

        // 商品名称

        sValue.append(r5\_Pid);

        // 商户订单号

        sValue.append(r6\_Order);

        // 易宝支付会员ID

        sValue.append(r7\_Uid);

        // 商户扩展信息

        sValue.append(r8\_MP);

        // 交易结果返回类型

        sValue.append(r9\_BType);

        String sNewString = PaymentUtil.hmacSign(sValue.toString(), keyValue);

        return sNewString.equals(hmac);

    }

    /\*\*

     \* @param aValue

     \* @param aKey

     \* @return

     \*/

    public static String hmacSign(String aValue, String aKey) {

        byte k\_ipad[] = new byte[64];

        byte k\_opad[] = new byte[64];

        byte keyb[];

        byte value[];

        try {

            keyb = aKey.getBytes(encodingCharset);

            value = aValue.getBytes(encodingCharset);

        } catch (UnsupportedEncodingException e) {

            keyb = aKey.getBytes();

            value = aValue.getBytes();

        }

        Arrays.fill(k\_ipad, keyb.length, 64, (byte) 54);

        Arrays.fill(k\_opad, keyb.length, 64, (byte) 92);

        for (int i = 0; i < keyb.length; i++) {

            k\_ipad[i] = (byte) (keyb[i] ^ 0x36);

            k\_opad[i] = (byte) (keyb[i] ^ 0x5c);

        }

        MessageDigest md = null;

        try {

            md = MessageDigest.getInstance("MD5");

        } catch (NoSuchAlgorithmException e) {

            return null;

        }

        md.update(k\_ipad);

        md.update(value);

        byte dg[] = md.digest();

        md.reset();

        md.update(k\_opad);

        md.update(dg, 0, 16);

        dg = md.digest();

        return toHex(dg);

    }

    public static String toHex(byte input[]) {

        if (input == null)

            return null;

        StringBuffer output = new StringBuffer(input.length \* 2);

        for (int i = 0; i < input.length; i++) {

            int current = input[i] & 0xff;

            if (current < 16)

                output.append("0");

            output.append(Integer.toString(current, 16));

        }

        return output.toString();

    }

    /\*\*

     \*

     \* @param args

     \* @param key

     \* @return

     \*/

    public static String getHmac(String[] args, String key) {

        if (args == null || args.length == 0) {

            return (null);

        }

        StringBuffer str = new StringBuffer();

        for (int i = 0; i < args.length; i++) {

            str.append(args[i]);

        }

        return (hmacSign(str.toString(), key));

    }

    /\*\*

     \* @param aValue

     \* @return

     \*/

    public static String digest(String aValue) {

        aValue = aValue.trim();

        byte value[];

        try {

            value = aValue.getBytes(encodingCharset);

        } catch (UnsupportedEncodingException e) {

            value = aValue.getBytes();

        }

        MessageDigest md = null;

        try {

            md = MessageDigest.getInstance("SHA");

        } catch (NoSuchAlgorithmException e) {

            e.printStackTrace();

            return null;

        }

        return toHex(md.digest(value));

    }

}

# CategoryDao.java

package cn.itcast.goods.category.dao;

import cn.itcast.commons.CommonUtils;

import cn.itcast.goods.category.domain.Category;

import cn.itcast.jdbc.TxQueryRunner;

import org.apache.commons.dbutils.QueryRunner;

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

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

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

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

/\*\*

 \* 分类持久层

 \* @author qdmmy6

 \*

 \*/

public class CategoryDao {

    private QueryRunner qr = new TxQueryRunner();

    private Category toCategory(Map<String,Object> map) {

        Category category = CommonUtils.toBean(map, Category.class);

        String pid = (String)map.get("pid");// 如果是一级分类，那么pid是null

        if(pid != null) {//如果父分类ID不为空，

            Category parent = new Category();

            parent.setCid(pid);

            category.setParent(parent);

        }

        return category;

    }

    private List<Category> toCategoryList(List<Map<String,Object>> mapList) {

        List<Category> categoryList = new ArrayList<Category>();//创建一个空集合

        for(Map<String,Object> map : mapList) {//循环遍历每个Map

            Category c = toCategory(map);//把一个Map转换成一个Category

            categoryList.add(c);//添加到集合中

        }

        return categoryList;//返回集合

    }

    /\*\*

     \* 返回所有分类

     \*/

    public List<Category> findAll() throws SQLException {

        String sql = "select \* from t\_category where pid is null order by orderBy";

        List<Map<String,Object>> mapList = qr.query(sql, new MapListHandler());

        List<Category> parents = toCategoryList(mapList);

        for(Category parent : parents) {

            List<Category> children = findByParent(parent.getCid());

            parent.setChildren(children);

        }

        return parents;

    }

    /\*\*

     \* 通过父分类查询子分类

     \*/

    public List<Category> findByParent(String pid) throws SQLException {

        String sql = "select \* from t\_category where pid=? order by orderBy";

        List<Map<String,Object>> mapList = qr.query(sql, new MapListHandler(), pid);

        return toCategoryList(mapList);

    }

    /\*\*

     \* 添加分类

     \*/

    public void add(Category category) throws SQLException {

        String sql = "insert into t\_category(cid,cname,pid,`desc`) values(?,?,?,?)";

        String pid = null;//一级分类

        if(category.getParent() != null) {

            pid = category.getParent().getCid();

        }

        Object[] params = {category.getCid(), category.getCname(), pid, category.getDesc()};

        qr.update(sql, params);

    }

    /\*\*

     \* 获取所有父分类，但不带子分类的！

     \*/

    public List<Category> findParents() throws SQLException {

        /\*

         \* 1. 查询出所有一级分类

         \*/

        String sql = "select \* from t\_category where pid is null order by orderBy";

        List<Map<String,Object>> mapList = qr.query(sql, new MapListHandler());

        return toCategoryList(mapList);

    }

    /\*\*

     \* 加载分类

     \*/

    public Category load(String cid) throws SQLException {

        String sql = "select \* from t\_category where cid=?";

        return toCategory(qr.query(sql, new MapHandler(), cid));

    }

    /\*\*

     \* 修改分类

     \*/

    public void edit(Category category) throws SQLException {

        String sql = "update t\_category set cname=?, pid=?, `desc`=? where cid=?";

        String pid = null;

        if(category.getParent() != null) {

            pid = category.getParent().getCid();

        }

        Object[] params = {category.getCname(), pid, category.getDesc(), category.getCid()};

        qr.update(sql, params);

    }

    /\*\*

     \* 查询指定父分类下子分类的个数

     \*/

    public int findChildrenCountByParent(String pid) throws SQLException {

        String sql = "select count(\*) from t\_category where pid=?";

        Number cnt = (Number)qr.query(sql, new ScalarHandler(), pid);

        return cnt == null ? 0 : cnt.intValue();

    }

    /\*\*

     \* 删除分类

     \*/

    public void delete(String cid) throws SQLException {

        String sql = "delete from t\_category where cid=?";

        qr.update(sql, cid);

    }

}

# Category.java

package cn.itcast.goods.category.domain;

import java.util.List;

/\*\*

 \* 分类模块的实体类

 \*

 \* @author qdmmy6

 \*

 \*/

public class Category {

    private String cid;// 主键

    private String cname;// 分类名称

    private Category parent;// 父分类

    private String desc;// 分类描述

    private List<Category> children;// 子分类

    public String getCid() {

        return cid;

    }

    public void setCid(String cid) {

        this.cid = cid;

    }

    public String getCname() {

        return cname;

    }

    public void setCname(String cname) {

        this.cname = cname;

    }

    public Category getParent() {

        return parent;

    }

    public void setParent(Category parent) {

        this.parent = parent;

    }

    public String getDesc() {

        return desc;

    }

    public void setDesc(String desc) {

        this.desc = desc;

    }

    public List<Category> getChildren() {

        return children;

    }

    public void setChildren(List<Category> children) {

        this.children = children;

    }

}

# CategoryService.java

package cn.itcast.goods.category.service;

import cn.itcast.goods.category.dao.CategoryDao;

import cn.itcast.goods.category.domain.Category;

import java.sql.SQLException;

import java.util.List;

/\*\*

 \* 分类模块业务层

 \*/

public class CategoryService {

    private CategoryDao categoryDao = new CategoryDao();

    /\*\*

     \* 查询指定父分类下子分类的个数

     \*/

    public int findChildrenCountByParent(String pid) {

        try {

            return categoryDao.findChildrenCountByParent(pid);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 删除分类

     \*/

    public void delete(String cid) {

        try {

            categoryDao.delete(cid);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 修改分类

     \*/

    public void edit(Category category) {

        try {

            categoryDao.edit(category);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 加载分类

     \*/

    public Category load(String cid) {

        try {

            return categoryDao.load(cid);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 添加分类

     \*/

    public void add(Category category) {

        try {

            categoryDao.add(category);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 查询所有分类

     \*/

    public List<Category> findAll() {

        try {

            return categoryDao.findAll();

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 获取所有父分类，不带子分类

     \*/

    public List<Category> findParents() {

        try {

            return categoryDao.findParents();

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

}

# CategoryServlet.java

package cn.itcast.goods.category.web.servlet;

import cn.itcast.goods.category.domain.Category;

import cn.itcast.goods.category.service.CategoryService;

import cn.itcast.servlet.BaseServlet;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.IOException;

import java.util.List;

/\*\*

 \* 分类模块WEB层

 \*/

public class CategoryServlet extends BaseServlet {

    private CategoryService categoryService = new CategoryService();

    /\*\*

     \* 查询所有分类

     \*/

    public String findAll(HttpServletRequest req, HttpServletResponse resp)

            throws ServletException, IOException {

        List<Category> parents = categoryService.findAll();

        req.setAttribute("parents", parents);

        return "f:/jsps/left.jsp";

    }

 }

# CartItemDao.java

package cn.itcast.goods.cart.dao;

import cn.itcast.commons.CommonUtils;

import cn.itcast.goods.book.domain.Book;

import cn.itcast.goods.cart.domain.CartItem;

import cn.itcast.goods.user.domain.User;

import cn.itcast.jdbc.TxQueryRunner;

import org.apache.commons.dbutils.QueryRunner;

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

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

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

public class CartItemDao {

    private QueryRunner qr = new TxQueryRunner();

    private String toWhereSql(int len) {

        StringBuilder sb = new StringBuilder("cartItemId in(");

        for(int i = 0; i < len; i++) {

            sb.append("?");

            if(i < len - 1) {

                sb.append(",");

            }

        }

        sb.append(")");

        return sb.toString();

    }

    /\*\*

     \* 加载多个CartItem

     \*/

    public List<CartItem> loadCartItems(String cartItemIds) throws SQLException {

        Object[] cartItemIdArray = cartItemIds.split(",");

        String whereSql = toWhereSql(cartItemIdArray.length);

        String sql = "select \* from t\_cartitem c, t\_book b where c.bid=b.bid and " + whereSql;

        return toCartItemList(qr.query(sql, new MapListHandler(), cartItemIdArray));

    }

    /\*\*

     \* 按id查询

     \*/

    public CartItem findByCartItemId(String cartItemId) throws SQLException {

        String sql = "select \* from t\_cartItem c, t\_book b where c.bid=b.bid and c.cartItemId=?";

        Map<String,Object> map = qr.query(sql, new MapHandler(), cartItemId);

        return toCartItem(map);

    }

    /\*\*

     \* 批量删除

     \*/

    public void batchDelete(String cartItemIds) throws SQLException {

        Object[] cartItemIdArray = cartItemIds.split(",");

        String whereSql = toWhereSql(cartItemIdArray.length);

        String sql = "delete from t\_cartitem where " + whereSql;

        qr.update(sql, cartItemIdArray);//其中cartItemIdArray必须是Object类型的数组！

    }

    /\*\*

     \* 查询某个用户的某本图书的购物车条目是否存在

     \*/

    public CartItem findByUidAndBid(String uid, String bid) throws SQLException {

        String sql = "select \* from t\_cartitem where uid=? and bid=?";

        Map<String,Object> map = qr.query(sql, new MapHandler(), uid, bid);

        CartItem cartItem = toCartItem(map);

        return cartItem;

    }

    /\*\*

     \* 修改指定条目的数量

     \*/

    public void updateQuantity(String cartItemId, int quantity) throws SQLException {

        String sql = "update t\_cartitem set quantity=? where cartItemId=?";

        qr.update(sql, quantity, cartItemId);

    }

    /\*\*

     \* 添加条目

     \*/

    public void addCartItem(CartItem cartItem) throws SQLException {

        String sql = "insert into t\_cartitem(cartItemId, quantity, bid, uid)" +

                " values(?,?,?,?)";

        Object[] params = {cartItem.getCartItemId(), cartItem.getQuantity(),

                cartItem.getBook().getBid(), cartItem.getUser().getUid()};

        qr.update(sql, params);

    }

    private CartItem toCartItem(Map<String,Object> map) {

        if(map == null || map.size() == 0) return null;

        CartItem cartItem = CommonUtils.toBean(map, CartItem.class);

        Book book = CommonUtils.toBean(map, Book.class);

        User user = CommonUtils.toBean(map, User.class);

        cartItem.setBook(book);

        cartItem.setUser(user);

        return cartItem;

    }

    private List<CartItem> toCartItemList(List<Map<String,Object>> mapList) {

        List<CartItem> cartItemList = new ArrayList<CartItem>();

        for(Map<String,Object> map : mapList) {

            CartItem cartItem = toCartItem(map);

            cartItemList.add(cartItem);

        }

        return cartItemList;

    }

    /\*\*

     \* 通过用户查询购物车条目

     \*/

    public List<CartItem> findByUser(String uid) throws SQLException {

        String sql = "select \* from t\_cartitem c, t\_book b where c.bid=b.bid and uid=? order by c.orderBy";

        List<Map<String,Object>> mapList = qr.query(sql, new MapListHandler(), uid);

        return toCartItemList(mapList);

    }

}

# CartItem.java

package cn.itcast.goods.cart.domain;

import cn.itcast.goods.book.domain.Book;

import cn.itcast.goods.user.domain.User;

import java.math.BigDecimal;

public class CartItem {

    private String cartItemId;// 主键

    private int quantity;// 数量

    private Book book;// 条目对应的图书

    private User user;// 所属用户

    public double getSubtotal() {

        BigDecimal b1 = new BigDecimal(book.getCurrPrice() + "");

        BigDecimal b2 = new BigDecimal(quantity + "");

        BigDecimal b3 = b1.multiply(b2);

        return b3.doubleValue();

    }

    public String getCartItemId() {

        return cartItemId;

    }

    public void setCartItemId(String cartItemId) {

        this.cartItemId = cartItemId;

    }

    public int getQuantity() {

        return quantity;

    }

    public void setQuantity(int quantity) {

        this.quantity = quantity;

    }

    public Book getBook() {

        return book;

    }

    public void setBook(Book book) {

        this.book = book;

    }

    public User getUser() {

        return user;

    }

    public void setUser(User user) {

        this.user = user;

    }

    public static void main(String[] args) {

        System.out.println(2.0-1.1);//0.8999999999999999

    }

}

# CartItemService.java

package cn.itcast.goods.cart.service;

import cn.itcast.commons.CommonUtils;

import cn.itcast.goods.cart.dao.CartItemDao;

import cn.itcast.goods.cart.domain.CartItem;

import java.sql.SQLException;

import java.util.List;

public class CartItemService {

    private CartItemDao cartItemDao = new CartItemDao();

    public List<CartItem> loadCartItems(String cartItemIds) {

        try {

            return cartItemDao.loadCartItems(cartItemIds);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 修改购物车条目数量

     \*/

    public CartItem updateQuantity(String cartItemId, int quantity) {

        try {

            cartItemDao.updateQuantity(cartItemId, quantity);

            return cartItemDao.findByCartItemId(cartItemId);

        } catch(SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 批量删除功能

     \*/

    public void batchDelete(String cartItemIds) {

        try {

            cartItemDao.batchDelete(cartItemIds);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 添加条目

     \*/

    public void add(CartItem cartItem) {

        try {

            CartItem \_cartItem = cartItemDao.findByUidAndBid(

                    cartItem.getUser().getUid(),

                    cartItem.getBook().getBid());

            if(\_cartItem == null) {

                cartItem.setCartItemId(CommonUtils.uuid());

                cartItemDao.addCartItem(cartItem);

            } else {//如果原来有这个条目，修改数量

                int quantity = cartItem.getQuantity() + \_cartItem.getQuantity();

                cartItemDao.updateQuantity(\_cartItem.getCartItemId(), quantity);

            }

        } catch(Exception e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 我的购物车功能

     \*/

    public List<CartItem> myCart(String uid) {

        try {

            return cartItemDao.findByUser(uid);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

}

# CartItemServlet.java

package cn.itcast.goods.cart.web.servlet;

import cn.itcast.commons.CommonUtils;

import cn.itcast.goods.book.domain.Book;

import cn.itcast.goods.cart.domain.CartItem;

import cn.itcast.goods.cart.service.CartItemService;

import cn.itcast.goods.user.domain.User;

import cn.itcast.servlet.BaseServlet;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.IOException;

import java.util.List;

import java.util.Map;

public class CartItemServlet extends BaseServlet {

    private CartItemService cartItemService = new CartItemService();

    /\*\*

     \* 添加购物车条目

     \*/

    public String add(HttpServletRequest req, HttpServletResponse resp)

            throws ServletException, IOException {

        Map map = req.getParameterMap();

        CartItem cartItem = CommonUtils.toBean(map, CartItem.class);

        Book book = CommonUtils.toBean(map, Book.class);

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

        cartItem.setBook(book);

        cartItem.setUser(user);

        cartItemService.add(cartItem);

        return myCart(req, resp);

    }

    /\*\*

     \* 我的购物车

     \*/

    public String myCart(HttpServletRequest req, HttpServletResponse resp)

            throws ServletException, IOException {

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

        String uid = user.getUid();

        List<CartItem> cartItemLIst = cartItemService.myCart(uid);

        req.setAttribute("cartItemList", cartItemLIst);

        return "f:/jsps/cart/list.jsp";

    }

}

# BookDao.java

package cn.itcast.goods.book.dao;

import cn.itcast.commons.CommonUtils;

import cn.itcast.goods.book.domain.Book;

import cn.itcast.goods.category.domain.Category;

import cn.itcast.goods.pager.Expression;

import cn.itcast.goods.pager.PageBean;

import cn.itcast.goods.pager.PageConstants;

import cn.itcast.jdbc.TxQueryRunner;

import org.apache.commons.dbutils.QueryRunner;

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

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

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

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import java.util.Map;

public class BookDao {

    private QueryRunner qr = new TxQueryRunner();

    /\*\*

     \* 删除图书

     \*/

    public void delete(String bid) throws SQLException {

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

        qr.update(sql, bid);

    }

    /\*\*

     \* 修改图书

     \*/

    public void edit(Book book) throws SQLException {

        String sql = "update t\_book set bname=?,author=?,price=?,currPrice=?," +

                "discount=?,press=?,publishtime=?,edition=?,pageNum=?,wordNum=?," +

                "printtime=?,booksize=?,paper=?,cid=? where bid=?";

        Object[] params = {book.getBname(),book.getAuthor(),

                book.getPrice(),book.getCurrPrice(),book.getDiscount(),

                book.getPress(),book.getPublishtime(),book.getEdition(),

                book.getPageNum(),book.getWordNum(),book.getPrinttime(),

                book.getBooksize(),book.getPaper(),

                book.getCategory().getCid(),book.getBid()};

        qr.update(sql, params);

    }

    /\*\*

     \* 按bid查询

     \*/

    public Book findByBid(String bid) throws SQLException {

        String sql = "SELECT \* FROM t\_book b, t\_category c WHERE b.cid=c.cid AND b.bid=?";

        Map<String,Object> map = qr.query(sql, new MapHandler(), bid);

        Book book = CommonUtils.toBean(map, Book.class);

        Category category = CommonUtils.toBean(map, Category.class);

        book.setCategory(category);

        // 把pid获取出来，创建一个Category parnet，把pid赋给它，然后再把parent赋给category

        if(map.get("pid") != null) {

            Category parent = new Category();

            parent.setCid((String)map.get("pid"));

            category.setParent(parent);

        }

        return book;

    }

    /\*\*

     \* 查询指定分类下图书的个数

     \*/

    public int findBookCountByCategory(String cid) throws SQLException {

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

        Number cnt = (Number)qr.query(sql, new ScalarHandler(), cid);

        return cnt == null ? 0 : cnt.intValue();

    }

    /\*\*

     \* 按分类查询

     \*/

    public PageBean<Book> findByCategory(String cid, int pc) throws SQLException {

        List<Expression> exprList = new ArrayList<Expression>();

        exprList.add(new Expression("cid", "=", cid));

        return findByCriteria(exprList, pc);

    }

    /\*\*

     \* 按书名模糊查询

     \*/

    public PageBean<Book> findByBname(String bname, int pc) throws SQLException {

        List<Expression> exprList = new ArrayList<Expression>();

        exprList.add(new Expression("bname", "like", "%" + bname + "%"));

        return findByCriteria(exprList, pc);

    }

    /\*\*

     \* 按作者查

     \*/

    public PageBean<Book> findByAuthor(String author, int pc) throws SQLException {

        List<Expression> exprList = new ArrayList<Expression>();

        exprList.add(new Expression("author", "like", "%" + author + "%"));

        return findByCriteria(exprList, pc);

    }

    /\*\*

     \* 按出版社查

     \*/

    public PageBean<Book> findByPress(String press, int pc) throws SQLException {

        List<Expression> exprList = new ArrayList<Expression>();

        exprList.add(new Expression("press", "like", "%" + press + "%"));

        return findByCriteria(exprList, pc);

    }

    /\*\*

     \* 多条件组合查询

     \*/

    public PageBean<Book> findByCombination(Book criteria, int pc) throws SQLException {

        List<Expression> exprList = new ArrayList<Expression>();

        exprList.add(new Expression("bname", "like", "%" + criteria.getBname() + "%"));

        exprList.add(new Expression("author", "like", "%" + criteria.getAuthor() + "%"));

        exprList.add(new Expression("press", "like", "%" + criteria.getPress() + "%"));

        return findByCriteria(exprList, pc);

    }

    /\*\*

     \* 通用的查询方法

     \*/

    private PageBean<Book> findByCriteria(List<Expression> exprList, int pc) throws SQLException {

        int ps = PageConstants.BOOK\_PAGE\_SIZE;//每页记录数

        StringBuilder whereSql = new StringBuilder(" where 1=1");

        List<Object> params = new ArrayList<Object>();//SQL中有问号，它是对应问号的值

        for(Expression expr : exprList) {

            whereSql.append(" and ").append(expr.getName())

                .append(" ").append(expr.getOperator()).append(" ");

            // where 1=1 and bid = ?

            if(!expr.getOperator().equals("is null")) {

                whereSql.append("?");

                params.add(expr.getValue());

            }

        }

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

        Number number = (Number)qr.query(sql, new ScalarHandler(), params.toArray());

        int tr = number.intValue();//得到了总记录数

        sql = "select \* from t\_book" + whereSql + " order by orderBy limit ?,?";

        params.add((pc-1) \* ps);

        params.add(ps);

        List<Book> beanList = qr.query(sql, new BeanListHandler<Book>(Book.class),

                params.toArray());

        PageBean<Book> pb = new PageBean<Book>();

        pb.setBeanList(beanList);

        pb.setPc(pc);

        pb.setPs(ps);

        pb.setTr(tr);

        return pb;

    }

    /\*\*

     \* 添加图书

     \*/

    public void add(Book book) throws SQLException {

        String sql = "insert into t\_book(bid,bname,author,price,currPrice," +

                "discount,press,publishtime,edition,pageNum,wordNum,printtime," +

                "booksize,paper,cid,image\_w,image\_b)" +

                " values(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)";

        Object[] params = {book.getBid(),book.getBname(),book.getAuthor(),

                book.getPrice(),book.getCurrPrice(),book.getDiscount(),

                book.getPress(),book.getPublishtime(),book.getEdition(),

                book.getPageNum(),book.getWordNum(),book.getPrinttime(),

                book.getBooksize(),book.getPaper(), book.getCategory().getCid(),

                book.getImage\_w(),book.getImage\_b()};

        qr.update(sql, params);

    }

}

# Book.java

package cn.itcast.goods.book.domain;

import cn.itcast.goods.category.domain.Category;

public class Book {

    private String bid;//主键

    private String bname;//图名

    private String author;//作者

    private double price;//定价

    private double currPrice;//当前价

    private double discount;//折扣

    private String press;//出版社

    private String publishtime;//出版时间

    private int edition;//版次

    private int pageNum;//页数

    private int wordNum;//字数

    private String printtime;//刷新时间

    private int booksize;//开本

    private String paper;//纸质

    private Category category;//所属分类

    private String image\_w;//大图路径

    private String image\_b;//小图路径

    public String getBid() {

        return bid;

    }

    public void setBid(String bid) {

        this.bid = bid;

    }

    public String getBname() {

        return bname;

    }

    public void setBname(String bname) {

        this.bname = bname;

    }

    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 double getCurrPrice() {

        return currPrice;

    }

    public void setCurrPrice(double currPrice) {

        this.currPrice = currPrice;

    }

    public double getDiscount() {

        return discount;

    }

    public void setDiscount(double discount) {

        this.discount = discount;

    }

    public String getPress() {

        return press;

    }

    public void setPress(String press) {

        this.press = press;

    }

    public String getPublishtime() {

        return publishtime;

    }

    public void setPublishtime(String publishtime) {

        this.publishtime = publishtime;

    }

    public int getEdition() {

        return edition;

    }

    public void setEdition(int edition) {

        this.edition = edition;

    }

    public int getPageNum() {

        return pageNum;

    }

    public void setPageNum(int pageNum) {

        this.pageNum = pageNum;

    }

    public int getWordNum() {

        return wordNum;

    }

    public void setWordNum(int wordNum) {

        this.wordNum = wordNum;

    }

    public String getPrinttime() {

        return printtime;

    }

    public void setPrinttime(String printtime) {

        this.printtime = printtime;

    }

    public int getBooksize() {

        return booksize;

    }

    public void setBooksize(int booksize) {

        this.booksize = booksize;

    }

    public String getPaper() {

        return paper;

    }

    public void setPaper(String paper) {

        this.paper = paper;

    }

    public Category getCategory() {

        return category;

    }

    public void setCategory(Category category) {

        this.category = category;

    }

    public String getImage\_w() {

        return image\_w;

    }

    public void setImage\_w(String image\_w) {

        this.image\_w = image\_w;

    }

    public String getImage\_b() {

        return image\_b;

    }

    public void setImage\_b(String image\_b) {

        this.image\_b = image\_b;

    }

}

# BookServlet.java

package cn.itcast.goods.book.web.servlet;

import cn.itcast.commons.CommonUtils;

import cn.itcast.goods.book.domain.Book;

import cn.itcast.goods.book.service.BookService;

import cn.itcast.goods.pager.PageBean;

import cn.itcast.servlet.BaseServlet;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import java.io.IOException;

public class BookServlet extends BaseServlet {

    private BookService bookService = new BookService();

    /\*\*

     \* 获取当前页码

     \*/

    private int getPc(HttpServletRequest req) {

        int pc = 1;

        String param = req.getParameter("pc");

        if(param != null && !param.trim().isEmpty()) {

            try {

                pc = Integer.parseInt(param);

            } catch(RuntimeException e) {}

        }

        return pc;

    }

    /\*\*

     \* 截取url，页面中的分页导航中需要使用它做为超链接的目标！

     \*/

    private String getUrl(HttpServletRequest req) {

        String url = req.getRequestURI() + "?" + req.getQueryString();

        int index = url.lastIndexOf("&pc=");

        if(index != -1) {

            url = url.substring(0, index);

        }

        return url;

    }

    /\*\*

     \* 按bid查询

     \*/

    public String load(HttpServletRequest req, HttpServletResponse resp)

            throws ServletException, IOException {

        String bid = req.getParameter("bid");//获取链接的参数bid

        Book book = bookService.load(bid);//通过bid得到book对象

        req.setAttribute("book", book);//保存到req中

        return "f:/jsps/book/desc.jsp";//转发到desc.jsp

    }

}

# BookService.java

package cn.itcast.goods.book.service;

import cn.itcast.goods.book.dao.BookDao;

import cn.itcast.goods.book.domain.Book;

import cn.itcast.goods.pager.PageBean;

import java.sql.SQLException;

public class BookService {

    private BookDao bookDao = new BookDao();

    /\*\*

     \* 删除图书

     \* @param bid

     \*/

    public void delete(String bid) {

        try {

            bookDao.delete(bid);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 修改图书

     \* @param book

     \*/

    public void edit(Book book) {

        try {

            bookDao.edit(book);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 返回当前分类下图书个数

     \* @param cid

     \* @return

     \*/

    public int findBookCountByCategory(String cid) {

        try {

            return bookDao.findBookCountByCategory(cid);

        } catch(SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 加载图书

     \* @param bid

     \* @return

     \*/

    public Book load(String bid) {

        try {

            return bookDao.findByBid(bid);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 按分类查

     \*/

    public PageBean<Book> findByCategory(String cid, int pc) {

        try {

            return bookDao.findByCategory(cid, pc);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 按书名查

     \*/

    public PageBean<Book> findByBname(String bname, int pc) {

        try {

            return bookDao.findByBname(bname, pc);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 按作者查

     \*/

    public PageBean<Book> findByAuthor(String author, int pc) {

        try {

            return bookDao.findByAuthor(author, pc);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 按出版社查

     \*/

    public PageBean<Book> findByPress(String press, int pc) {

        try {

            return bookDao.findByPress(press, pc);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 多条件组合查询

     \*/

    public PageBean<Book> findByCombination(Book criteria, int pc) {

        try {

            return bookDao.findByCombination(criteria, pc);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

    /\*\*

     \* 添加图书

     \*/

    public void add(Book book) {

        try {

            bookDao.add(book);

        } catch (SQLException e) {

            throw new RuntimeException(e);

        }

    }

}