javaEE开发模式

2019年5月24日 星期五 14:07

1. 纯Servlet开发

   单纯的servlet开发，如果面对复杂页面将会难于处理，所以前台HTML内容还是应该放在页面中书写。

2. 纯JSP开发

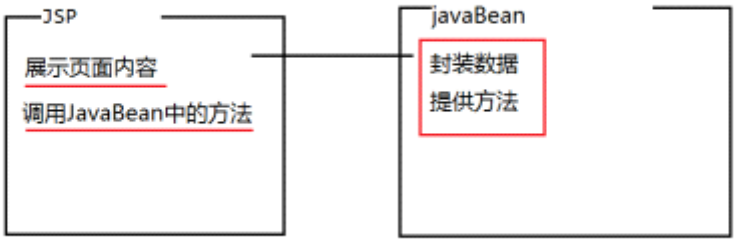
   相比servlet开发，纯jsp开发比较简单，但是面对复杂页面构成和代码逻辑的时候，HTML内容和Java语句内容相互嵌套，如果书写有误则会造成完全无法使用，代码可读性低，可维护性需求较高，便捷性较低。

3. JSP+JSP标签库

   利用JSP+JSP标签库的确简化了页面内容，但是面对复杂的逻辑需求是，在JSP页面中标签库可能无法处理，需要自行设计标签，而且这些标签域HTML标签相互嵌套，对于页面代码的可读性来说还是较低，由于逻辑代码和页面展示代码耦合在一起所以管理起来十分不便。

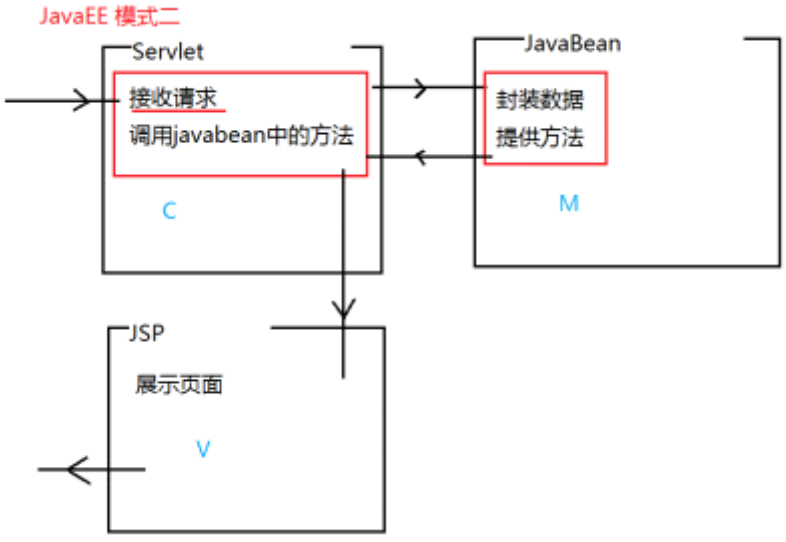
4. java+javaBean (Model One 模式一)

   javaEE模式一

   在模式一中JSP页面负责展示页面内容和调用javaBean中的方法。在javaBean中负责提供封装数据和提供方法。在JSP页面中既展示HTML内容也书写java代码逻辑，一个模块进行两个模块的工作，导致如果出现问题，模块之间会相互影响，所以应该将JSP页面中的两部分工作，拆分成两个模块，降低模块之内的耦合性。

5. Servlet+javaBean+JSP Model Two模式二

   模式二将javaEE处理的数据分成了三部分。在Servlet中主要负责接收请求，和根据请求调用javabean中的方法。在javaBean中主要负责封装数据和提供方法。杂

JSP页面中只负责数据和页面的展示。在这样的模型下，摸一个模块都进行各自模块功能，各个模块之间已经很大程度的降低了耦合性，各自执行各自的工作，当前模块产生异常，只需寻找当前模块的异常问题，无序寻找其他模块的问题。

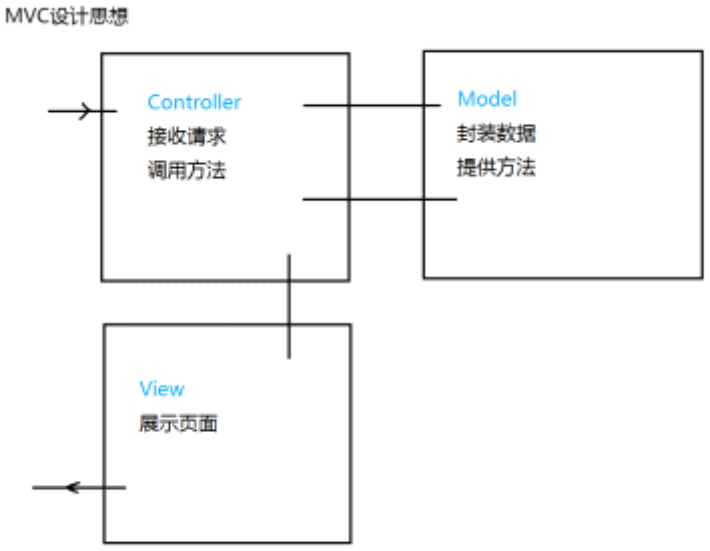
三个模块之间：

相互独立。虽然存在耦合，但是这个耦合是必要的存在，如果抛弃这个耦合将会使每个模块完全独立，导致无法一起工作。

模块之间相互协作，可读性已经最大程度的提升，代码量虽然有所上升，但是更加便于代码的维护和管理。

6. MVC设计思想

MVC设计思想并不是javaEE特有的设计思想，而是在设计程序代码的时候，一个通用的设计思想，在其他语言和程序中同样也适用。



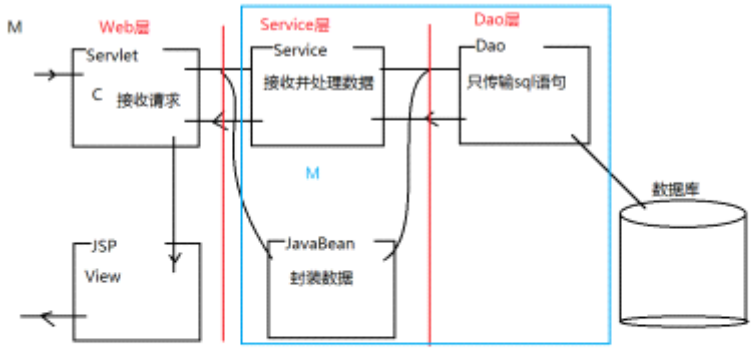
MVC设计思想一共分为三个部分，第一是负责接收用户发送请求的Controller(控制器)，接收请求之后，根据用户请求的内容，调用对应模型中的方法。第二是负责提供封装数据和方法的Model(模型)，模型中有用户请求发送来的数据，以及数据处理的方法，model处理完用户数据之后，将结果数据再次范围到Controller中。第三是负责展示用户结果数据View(视图)，View中，将Controller传递的结果数据不做任何处理直接展示到页面当中。

特点：

MVC中三个模块都各自执行各自模块的功能，模块之间相互独立。

正是因为模块之间相互独立，所以代码的管理更加便捷，可维护性更强。

7. JavaEE经典三层架构

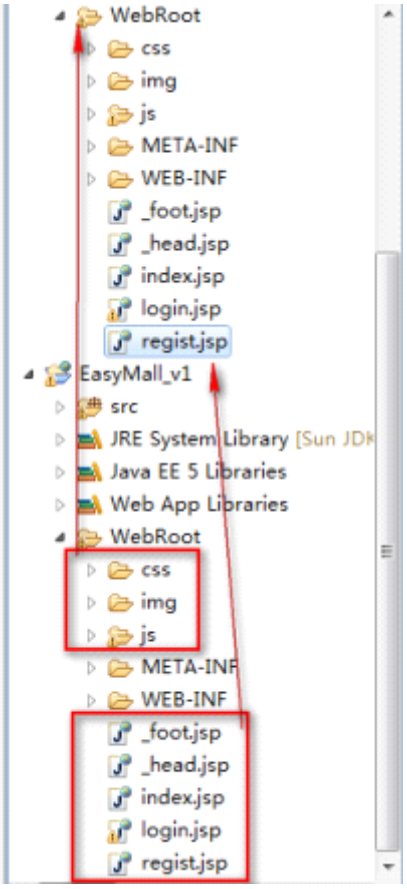

M
Web层
Servlet
C 接收请求
Service层
Service
接收并处理数据
Dao层
Dao
只传输sql语句
数据库
M
JSP
View
JavaBean
封装数据

重构EasyMall项目

2019年5月24日 星期五 15:07

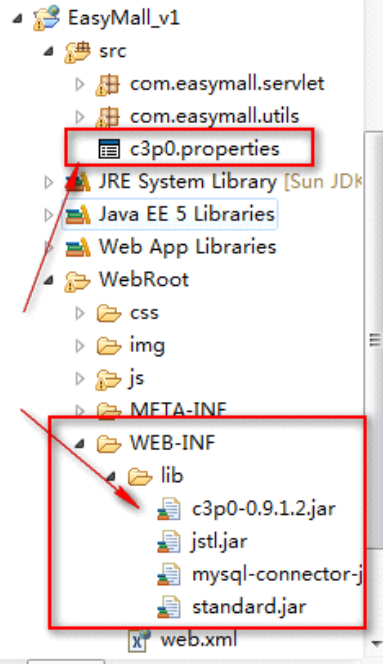
1. 创建EasyMall web应用
2. 将原有EasyMall的静态资源引入

   将原有web应用全部静态资源复制到新的EasyMallweb应用中



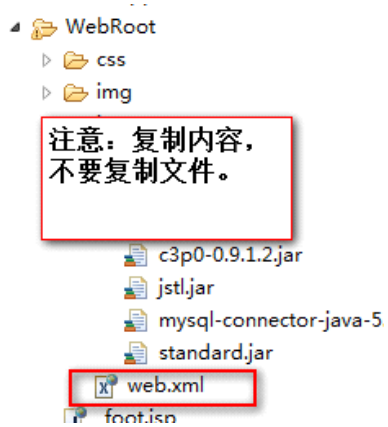
3. 导入开发包以及配置文件

   在WEB-INF的lib中有开发所需包

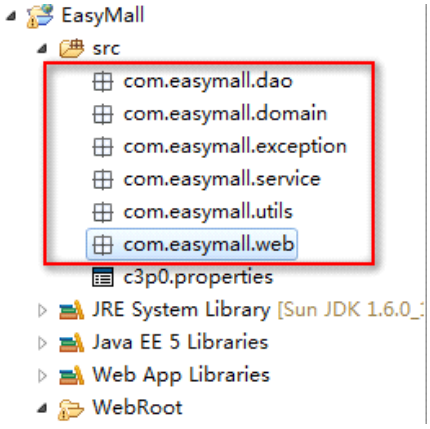


4. 复制web.xml文件内容

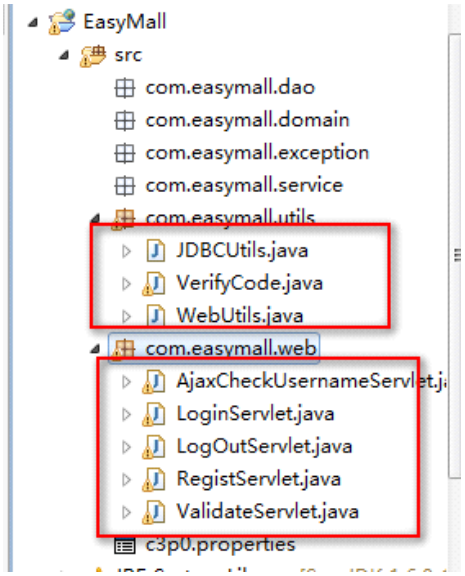
   将原有web应用的web.xmln内容复制到新的EasyMall 的web.xml中



5. 根据经典架构---构建新的目录结构



6. 导入动态资源和工具类



7. 修改web.xml文件中的配置信息

将原有的com.easymall.servlet 更改为com.easymall.web

8. 创建JavaBean--User 用来存储用户信息数据

```
package com.easymall.domain;

public class User{
```

```
    private int id;
    private String username;
    private String password;
    private String nickname;
    private String email;


    public User() {
        super();
    }
    public User(int id, String username, String password, String nickname,
        String email) {
        super();
        this.id = id;
        this.username = username;
        this.password = password;
        this.nickname = nickname;
        this.email = email;
    }
    public int getId() {
        return id;
    }
    public void setId(int 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 getNickname() {
        return nickname;
    }
    public void setNickname(String nickname) {
        this.nickname = nickname;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }


}
```

9. 实现登录功能：

a. 修改LoginServlet

```
//登录功能
public class LoginServlet extends HttpServlet {
    //代表User信息的Service层
    private UserService userService = new UserService();
    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        //1.乱码处理
        //2.获取用户信息
```

```
            //如果remname值为true则要求记住用户名
            //3.访问数据库，比对用户信息


            //UserService负责接收javabean提供的数据
            //UserService提供处理数据的方法
            //将登陆功能的返回结果作为保存用户登陆信息使用。所以需要返回值对象
            User user =null;
            try {
                user = userService.loginUser(username,password);

            } catch (MsgException e) {
                request.setAttribute("msg", e.getMessage());
                request.getRequestDispatcher("/login.jsp").forward(request, response);
                return;
            }
            //将用户信息保存仅入session域
            request.getSession().setAttribute("user", user);
            //回到首页
            response.sendRedirect(request.getContextPath()+"/");

        }

        public void doPost(HttpServletRequest request, HttpServletResponse response)
                throws ServletException, IOException {
            doGet(request, response);

        }

}
```

b. 创建UserService

```
public class UserService {
    private UserDao userDao = new UserDao();
    /**
     * 根据用户名和密码实现登陆功能，并将用户信息对象返回
     * @param username 用户名
     * @param password 密码
     * @return 用户信息对象User
     */
    public User loginUser(String username, String password) {

        User user = userDao.findUserByUsernameAndPassword(username,password);

        if(user != null){
            //如果用户名和密码匹配，则正常登陆，并跳转到首页
            return user;
        }else{
            //如果用户名和密码不匹配，则提示错误信息
            throw new MsgException("用户名或密码不正确");
        }
    }
}
```

c. 创建异常类MsgException

```
package com.easymall.exception;

public class MsgException extends RuntimeException {
    public MsgException(){

    }
    public MsgException(String e){
```

```
            super(e);
        }
    }
```

d. 创建UserDao

```
public class UserDao {

    public User findUserByUsernameAndPassword(String username, String password) {
        Connection conn = null;
        PreparedStatement stat = null;
        ResultSet rs = null;
        try {
            conn = JDBCUtils.getConnection();
            stat = conn.prepareStatement("select * from user where username=? and password=?");
            stat.setString(1, username);
            stat.setString(2, password);
            rs = stat.executeQuery();
            if(rs.next()){
                //查询到用户信息，返回User对象
                User user = new User();
                user.setId(rs.getInt("id"));
                user.setUsername(rs.getString("username"));
                user.setPassword(rs.getString("password"));
                user.setNickname(rs.getString("nickname"));
                user.setEmail(rs.getString("email"));
                return user;
            }else{
                //没有查询到用户信息
                return null;
            }
        } catch (SQLException e) {
            e.printStackTrace();
            throw new RuntimeException(e);
        }finally{
            JDBCUtils.close(conn, stat, rs);
        }
    }
}
```

e. head.jsp修改：

```
<c:if test="${sessionScope.user.username == null }">
            <a href="${pageContext.request.contextPath }/login.jsp">登录</a>  |  
        <a href="${pageContext.request.contextPath }/regist.jsp">注册</a>
    </c:if>
    <c:if test="${sessionScope.user.username != null }">
            <a href="#">欢迎 ${sessionScope.user.username } 回来</a>  |  
        <a href="${pageContext.request.contextPath }/LogOutServlet">注销</a>
    </c:if>
```

10. 实现注册功能

修改RegistServlet：

```
/**
 * 注册功能
 * @author Administrator
 *
 */
public class RegistServlet extends HttpServlet {
    private UserService userService = new UserService();
    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        //乱码处理
        //请求乱码和响应乱码处理
        //UserService
```

```
            User user = new User(0,username,password,nickname,email);
            try {
                userService.registUser(user);
            } catch (MsgException e) {
                request.setAttribute("msg",e.getMessage());
                request.getRequestDispatcher("/regis.jsp").forward(request, response);
                return;
            }

            //
            //跳转回首页
            response.getWriter().write(
                "<h1 align='center'><font color='red'>" +
                "恭喜注册成功，3秒之后跳转回首页</font></h1>");
            response.setHeader("refresh", "3;url=http://www.easymall.com");
        }

        public void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
            doGet(request, response);

        }

}
```

修改UserService：

```
public class UserService {
    private UserDao userDao = new UserDao();
    /**
     * 根据用户名和密码实现登陆功能，并将用户信息对象返回
     * @param username 用户名
     * @param password 密码
     * @return 用户信息对象User
     */
    public User loginUser(String username, String password) {

        User user = userDao.findUserByUsernameAndPassword(username,password);

        if(user != null){
            //如果用户名和密码匹配，则正常登陆，并跳转到首页
            return user;
        }else{
            //如果用户名和密码不匹配，则提示错误信息
            throw new MsgException("用户名或密码不正确");
        }




    }
    /**
     * 注册用户
     * @param user 用户信息对象，其中包含全部的用户信息参数
     */
    public void registUser(User user) {
        //判断用户名是否存在
        User findUser = userDao.findUserByUsername(user.getUsername());

        if(findUser != null){
            //如果存在则提示错误信息用户名已存在
            throw new MsgException("用户名已存在");
        }else{
            //如果不存在则正常添加用户
```

```
                userDao.addUser(user);
            }


        }



}
```

修改UserDao：

```
package com.easymall.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

import com.easymall.domain.User;
import com.easymall.utils.JDBCUtils;

public class UserDao {

    public User findUserByUsernameAndPassword(String username, String password) {
        Connection conn = null;
        PreparedStatement stat = null;
        ResultSet rs = null;
        try {
            conn = JDBCUtils.getConnection();
            stat = conn.prepareStatement("select * from user where username=? and password=?");
            stat.setString(1, username);
            stat.setString(2, password);
            rs = stat.executeQuery();
            if(rs.next()){
                //查询到用户信息，返回User对象
                User user = new User();
                user.setId(rs.getInt("id"));
                user.setUsername(rs.getString("username"));
                user.setPassword(rs.getString("password"));
                user.setNickname(rs.getString("nickname"));
                user.setEmail(rs.getString("email"));
                return user;
            }else{
                //没有查询到用户信息
                return null;
            }
        } catch (SQLException e) {
            e.printStackTrace();
            throw new RuntimeException(e);
        }finally{
            JDBCUtils.close(conn, stat, rs);
        }


    }
    /**
     * 向user表中插入一条用户数据
     * @param user 用户信息对象
     */
    public void addUser(User user) {
        Connection conn = null;
        PreparedStatement stat = null;
        try {
            conn = JDBCUtils.getConnection();
```

```
                stat = conn.prepareStatement("insert  into user values(null,?,?,?,?)");
                stat.setString(1, user.getUsername());
                stat.setString(2, user.getPassword());
                stat.setString(3, user.getNickname());
                stat.setString(4,user.getEmail());
                stat.executeUpdate();

            } catch (SQLException e) {
                e.printStackTrace();
                throw new RuntimeException(e);
            }finally{
                JDBCUtils.close(conn, stat, null);
            }

        }
        /**
         * 根据用户名查询用户信息
         * @param username 用户名
         */
        public User findUserByUsername(String username) {
            Connection conn = null;
            PreparedStatement  stat = null;
            ResultSet rs = null;
            try {
                conn = JDBCUtils.getConnection();
                stat = conn.prepareStatement("select  * from user where username=? ");
                stat.setString(1, username);
                rs = stat.executeQuery();
                if(rs.next()){
                    //查询到用户信息，返回User对象
                    User user = new User();
                    user.setId(rs.getInt("id"));
                    user.setUsername(rs.getString("username"));
                    user.setPassword(rs.getString("password"));
                    user.setNickname(rs.getString("nickname"));
                    user.setEmail(rs.getString("email"));
                    return user;
                }else{
                    //没有查询到用户信息
                    return null;
                }
            } catch (SQLException e) {
                e.printStackTrace();
                throw new RuntimeException(e);
            }finally{
                JDBCUtils.close(conn, stat, rs);
            }
        }
}
```

11. 注销功能

```
package com.easymall.web;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
//注销用户登录状态
public class LogOutServlet extends HttpServlet {

    public void doGet(HttpServletRequest  request, HttpServletResponse response)
            throws ServletException, IOException {
```

```
            //注销用户登录状态可以将session删除，删除session后其中的属性值也会销毁
            //实现用户注销的功能
            //先判断session对象是否为空，如果为空则不需要释放。
            //不为空再释放。
            if(request.getSession(false)!=null){
                request.getSession().invalidate();
            }

            response.sendRedirect("http://www.easymall.com");
        }

        public void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
            doGet(request, response);

        }

}
```

12. 作业：
    a. 配置一个错误友好提示页面。<error-page></error-page>