### JAVA第二阶段—DAY17-JAVA案例

1. 演示JSP三种脚本的使用

* 参考答案

实现此案例需要按照如下步骤进行。

步骤一：创建JavaEE项目

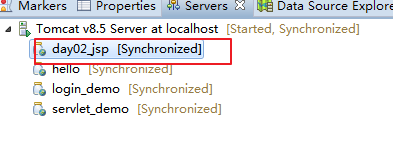
创建项目day02\_jsp

步骤二：在项目根目录WebContent下新建index.jsp文件

该文件代码如下

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=utf-8"*  pageEncoding=*"utf-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"utf-8"*>  <title>jsp</title>  </head>  <body>  Hi JSP!!!  </body>  </html> |
|  |

步骤三：将项目部署到Tomcat,启动服务器

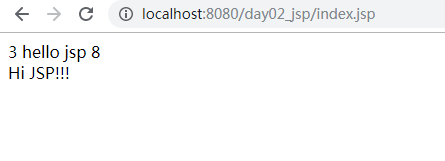


步骤四：在index.jsp页面书写jsp三种脚本

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html;charset=utf-8"*  pageEncoding=*"utf-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"utf-8"*>  <title>jsp</title>  </head>  <body>  <!-- 代码小脚本 -->  <%  **int** a = 3; //局部变量  %>  <!-- 输出小脚本 -->  <%=a%>  <%="hello jsp" %>  <!-- 声明 小脚本 -->  <%!  **int** b = 8;//成员变量  %>  <%=b %>  <br/>  Hi JSP!!!  </body>  </html> |

步骤五：浏览器访问

在浏览器地址输入：http://localhost:8080/day02\_jsp/index.jsp



步骤六：拓展

分别使用输出小脚本 和 声明小脚本 定义变量 让变量递增 观察变量的变化

|  |
| --- |
| <% **int** i = 0; %>  <%!**int** j = 0;%>  <%  i++;  j++;  %>  <%="i的值:"+i %>  <%="J的值:"+j %> |

1. 演示JSP注释和HTML注释区别。

* 参考答案

实现此案例需要按照如下步骤进行。

步骤一：创建JavaEE项目

直接使用案例1项目

步骤二：创建一个JSP页面zhushi.jsp

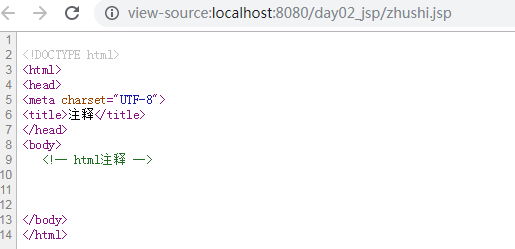
在该页面编写JSP注释 和HTML注释

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>注释</title>  </head>  <body>  <!-- html注释 -->  <%--jsp注释 --%>  <%--  System.out.println("hello");  --%>    </body>  </html> |

步骤四：浏览器访问

在浏览器地址输入：http://localhost:8080/day02\_jsp/zhushi.jsp

右键点击查看网页源代码 发现HTML注释可以看到 JSP注释看不到，在JSP页面推荐使用JSP注释



1. 演示page指令

* 参考答案

实现此案例需要按照如下步骤进行。

步骤一：创建JavaEE项目

直接使用案例1项目

步骤二：定义JSP页面 page.jsp

在JSP页面创建一个List集合 可以使用import 属性 导入相关的包

|  |
| --- |
| <%@page import=*"java.util.ArrayList"*%>  <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  import=*"java.util.List"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>page指令</title>  </head>  <body>  <%  //page指令 使用import 导入相关包  List list = **new** ArrayList();  %>  </body>  </html> |

步骤三：设置page.jsp页面的page指令的errorPage属性为true

在并写一条错误语句 int i = 3/0;

|  |
| --- |
| <%@page import=*"java.util.ArrayList"*%>  <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  import=*"java.util.List"* errorPage=*"500.jsp"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>page指令</title>  </head>  <body>  <%  //page指令 使用import 导入相关包  List list = **new** ArrayList();  **int** i = 3/0;  %>  </body>  </html> |

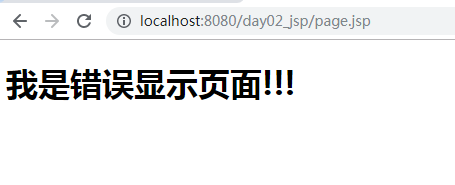
步骤四：创建一个JSP页面 500.jsp 用于显示错误

将该页面的isErrorPage属性设置为true

|  |
| --- |
| <%--将页面设置为错误显示页面 isErrorPage="true"--%>  <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"* isErrorPage=*"true"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>错误显示页面</title>  </head>  <body>  <h1>我是错误显示页面!!!</h1>  </body>  </html> |
|  |

步骤五：浏览器访问 page.jsp页面 查看效果

在浏览器地址输入：http://localhost:8080/day02\_jsp/page.jsp



1. 演示include指令

* 参考答案

实现此案例需要按照如下步骤进行。

步骤一：创建JavaEE项目

直接使用案例1项目

步骤二：定义JSP页面

在web根目录WebContent 中创建目录include,在该目录中分别创建三个JSP文件

body.jsp、footer.jsp、header.jsp,并在body.jsp使用include指令引入公共的头和底部

body.jsp

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>身体</title>  </head>  <body>  <!--使用include指令引入头部 -->  <%@include file=*"header.jsp"* %>  <h1>我是身体</h1>  <!--使用include指令引入底部 -->  <%@include file=*"footer.jsp"* %>  </body>  </html> |

header.jsp

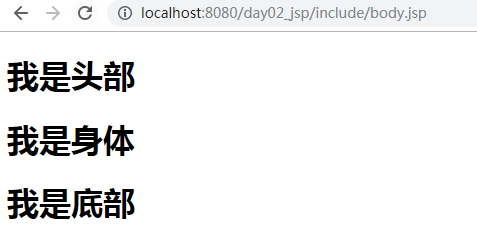
|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>头部</title>  </head>  <body>  <h1>我是头部 </h1>  </body>  </html> |

footer.jsp

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>我是底部</title>  </head>  <body>  <h1>我是底部</h1>  </body>  </html> |

步骤三：浏览器访问 body.jsp页面 查看效果

在浏览器地址输入：<http://localhost:8080/day02_jsp/include/body.jsp>



1. 演示include动作

* 参考答案

实现此案例需要按照如下步骤进行。

步骤一：创建JavaEE项目

直接使用案例1项目

步骤二：定义JSP页面

在web根目录WebContent 中创建目录include2,在该目录中分别创建三个JSP文件

body.jsp、footer.jsp、header.jsp,并在body.jsp使用include动作引入公共的头和底部

body.jsp

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>身体</title>  </head>  <body>  <!--使用include动作引入头部 -->  <jsp:include page=*"header.jsp"*></jsp:include>  <h1>我是身体 动作包含</h1>  <!--使用include动作引入底部 -->  <jsp:include page=*"footer.jsp"*></jsp:include>  </body>  </html> |

header.jsp

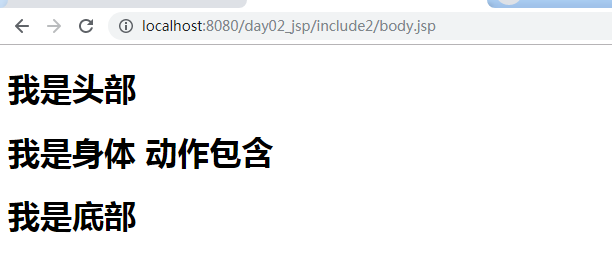
|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>头部</title>  </head>  <body>  <h1>我是头部 </h1>  </body>  </html> |

footer.jsp

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>我是底部</title>  </head>  <body>  <h1>我是底部</h1>  </body>  </html> |

步骤三：浏览器访问 body.jsp页面 查看效果

在浏览器地址输入：[http://localhost:8080/day02\_jsp/include2/body.jsp](http://localhost:8080/day02_jsp/include/body.jsp)



1. 演示forward动作

* 参考答案

实现此案例需要按照如下步骤进行。

步骤一：创建JavaEE项目

直接使用案例1项目

步骤二：定义JSP页面

在web根目录WebContent 中创建JSP文件forward.jsp 和target.jsp

forward.jsp 中使用转发动作，并携带参数

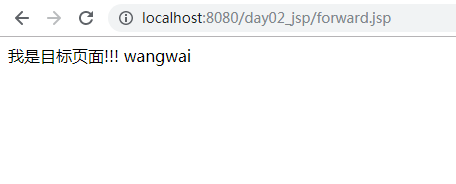
|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>转发动作</title>  </head>  <body>  <!-- 使用转发动作 转发请求 target.jsp页面 -->  <jsp:forward page=*"target.jsp"*>  <jsp:param value=*"wangwai"* name=*"username"*/>  </jsp:forward>  </body>  </html> |

target.jsp 使用request对象 接收参数

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>目标页面</title>  </head>  <body>  我是目标页面!!!  <!--获取带过来的参数 -->  <%=request.getParameter("username") %>  </body>  </html> |

步骤三：浏览器访问 forward.jsp页面 查看效果

在浏览器地址输入：http://localhost:8080/day02\_jsp/forward.jsp



1. 使用MVC模式实现登录和图书列表的显示

需求：

优化登录功能

1.登录界面使用提供的静态页面

2.用户登录失败，回显登录页面

3.用户登录成功 跳转到图书列表页面

* 参考答案

实现此案例需要按照如下步骤进行。

步骤一：创建JavaEE项目

创建项目day02\_book

步骤二：重写登录功能

基于login\_demo项目，将该项目代码拷贝到day02\_book

更新登录页面 , 并重写 登录成功和登录失败业务代码

LoginServlet

|  |
| --- |
| **package** com.bailiban.servlet;  **import** java.io.IOException;  **import** java.util.Map;  **import** javax.servlet.ServletException;  **import** javax.servlet.annotation.WebServlet;  **import** javax.servlet.http.HttpServlet;  **import** javax.servlet.http.HttpServletRequest;  **import** javax.servlet.http.HttpServletResponse;  **import** org.apache.commons.beanutils.BeanUtils;  **import** com.bailiban.domian.User;  **import** com.bailiban.service.UserService;  **import** com.bailiban.service.impl.UserServiceImpl;  /\*\*  \* Servlet implementation class LoginServlet  \*/  @WebServlet("/login")  **public** **class** LoginServlet **extends** HttpServlet {  **private** **static** **final** **long** ***serialVersionUID*** = 1L;  **public** LoginServlet() {  **super**();  }    **protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {    //使用request对象解决 post 提交中文乱码  request.setCharacterEncoding("utf-8");  // //使用request对象 接收表单提交的数据  // //getParameter(String name) name为表单空件的name属性值  // String username = request.getParameter("username");  // String password = request.getParameter("password");  // //接收到的数据封装进用户对象  // User loginUser = new User(username, password);  //使用BeanUtils优化参数的封装  //获取所有请求参数  Map<String, String[]> map = request.getParameterMap();  //将参数数据封装进实体类  User loginUser = **new** User();  **try** {  BeanUtils.*populate*(loginUser, map);  } **catch** (Exception e) {  // **TODO** Auto-generated catch block  e.printStackTrace();  }  //实例化UserService对象  UserService userService = **new** UserServiceImpl();  //调用登录方法 得到从数据库中查询的用户对象  User user = userService.login(loginUser);  //判断用户是否登录成功  **if**(user!=**null**) {  //如果登录成功 重定向到 图书列表页面  //request.getContextPath() 获取虚拟目录  response.sendRedirect(request.getContextPath()+"/book/list");    }**else** {  //出现错误 向request域中放错误信息  request.setAttribute("msg", "用户登录失败!!!");  //转发到login.jsp 显示错误提示  request.getRequestDispatcher("/login.jsp").forward(request, response);    }    }  **protected** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  //无论是get还是post请求 都执行doGet方法 避免写重复的代码  doGet(request, response);  }  } |

登录页面 login.jsp

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>管理员登录</title>  <!--导入框架样式文件-->  <link rel=*"stylesheet"* href=*"css/bootstrap.min.css"* />  <link rel=*"stylesheet"* href=*"css/base.css"* />  <script type=*"text/javascript"* src=*"js/jquery1.12.4.min.js"*></script>  <script type=*"text/javascript"* src=*"js/bootstrap.min.js"*></script>  </head>  <body>  <div class=*"container"*>  <h2>管理员登录</h2>  <form method=*"post"* action=*"/day02\_book/login"*>  <div class=*"form-group"*>  <label class=*"control-label"*>用户名</label>  <input type=*"text"* class=*"form-control"* placeholder=*"请输入用户名"* name=*"username"* required />  </div>  <div class=*"form-group"*>  <label class=*"control-label"*>密码</label>  <input type=*"password"* class=*"form-control"* placeholder=*"请输入密码"* name=*"password"* required />  </div>  <button type=*"submit"* class=*"btn btn-success btn-block"*>登录</button>  <button type=*"submit"* class=*"btn btn-danger btn-block"*>取消</button>  </form>  <!-- 使用小脚本 从request域中 取出 错误信息-->  <%  String msg = (String)request.getAttribute("msg");  **if**(msg!=**null**){  %>  <div class=*"alert alert-warning alert-dismissible"* role=*"alert"*>  <button type=*"button"* class=*"close"* data-dismiss=*"alert"* aria-label=*"Close"*>  <span aria-hidden=*"true"*>&times;</span>  </button>  <%=msg %>  </div>  <%  }  %>    </div>  </body>  </html> |

步骤三：创建图书表

在javaweb\_day01数据库中创建 图书表并初始化数据

|  |
| --- |
| #图书表  CREATE TABLE t\_book(    id INT PRIMARY KEY AUTO\_INCREMENT,  NAME VARCHAR(32) UNIQUE NOT NULL,  price DOUBLE NOT NULL,  author VARCHAR(32) NOT NULL,  image VARCHAR(32) DEFAULT NULL, -- 图片  publish\_date DATETIME NOT NULL    );  INSERT INTO t\_book VALUES (NULL ,'葵花宝典',99.9,'东方不败','b1.png',NOW());  INSERT INTO t\_book VALUES (NULL ,'九阴真经',9.9,'周芷若','b2.png',NOW());  INSERT INTO t\_book VALUES (NULL ,'独孤九剑',888,'令狐冲' , 'b3.png',NOW()); |

步骤四：创建实体类 Book

创建包com.bailiban.domain,在该包下创建User类

|  |
| --- |
| **package** com.bailiban.domian;  **public** **class** Book {  **private** **int** id; //编号  **private** String name;//书籍名称  **private** **double** price; //价格  **private** String author; //作者  **private** String image; //图片  **private** String publishDate;//出版日期  **public** **int** getId() {  **return** id;  }  **public** **void** setId(**int** id) {  **this**.id = id;  }  **public** String getName() {  **return** name;  }  **public** **void** setName(String name) {  **this**.name = name;  }  **public** **double** getPrice() {  **return** price;  }  **public** **void** setPrice(**double** price) {  **this**.price = price;  }  **public** String getAuthor() {  **return** author;  }  **public** **void** setAuthor(String author) {  **this**.author = author;  }  **public** String getImage() {  **return** image;  }  **public** **void** setImage(String image) {  **this**.image = image;  }  **public** String getPublishDate() {  **return** publishDate;  }  **public** **void** setPublishDate(String publishDate) {  **this**.publishDate = publishDate;  }    } |

步骤五：编写图书 dao层

创建包com.bailiban.dao,在该包下创建BookDao

|  |
| --- |
| **package** com.bailiban.dao;  **import** java.util.List;  **import** com.bailiban.domian.Book;  **public** **interface** BookDao {    //查询所有书籍  List<Book> selecAll();  } |

创建包com.bailiban.dao.impl,在该包下创建BookDaoImpl实现类

|  |
| --- |
| **package** com.bailiban.dao.impl;  **import** java.util.List;  **import** org.springframework.jdbc.core.BeanPropertyRowMapper;  **import** org.springframework.jdbc.core.JdbcTemplate;  **import** com.bailiban.dao.BookDao;  **import** com.bailiban.domian.Book;  **import** com.bailiban.utils.JDBCUtils;  **public** **class** BookDaoImpl **implements** BookDao {  //声明JDBCTemplate对象共用  **private** JdbcTemplate template = **new** JdbcTemplate(JDBCUtils.*getDataSource*());  //查询所有图书  @Override  **public** List<Book> selecAll() {    //查询所有书籍  List<Book> list = template.query("select \* from t\_book", **new** BeanPropertyRowMapper<Book>(Book.**class**));    **return** list;  }      } |

步骤六：编写图书service层

创建包com.bailiban.service,在该包下创建BookService接口

|  |
| --- |
| **package** com.bailiban.service;  **import** java.util.List;  **import** com.bailiban.domian.Book;  **public** **interface** BookService {  //查询所有书籍  List<Book> selecAll();  } |

创建包com.bailiban.service.impl,在该包下创建BookServiceImpl实现类

|  |
| --- |
| **package** com.bailiban.service.impl;  **import** java.util.List;  **import** com.bailiban.dao.BookDao;  **import** com.bailiban.dao.impl.BookDaoImpl;  **import** com.bailiban.domian.Book;  **import** com.bailiban.service.BookService;  **public** **class** BookServiceImpl **implements** BookService {    //创建dao  **private** BookDao bookDao = **new** BookDaoImpl();  //查询所有数据  @Override  **public** List<Book> selecAll() {  // **TODO** Auto-generated method stub  **return** bookDao.selecAll();  }  } |

步骤七：编写图书列表BookListServlet中的代码

|  |
| --- |
| **package** com.bailiban.servlet;  **import** java.io.IOException;  **import** java.util.List;  **import** javax.servlet.ServletException;  **import** javax.servlet.annotation.WebServlet;  **import** javax.servlet.http.HttpServlet;  **import** javax.servlet.http.HttpServletRequest;  **import** javax.servlet.http.HttpServletResponse;  **import** com.bailiban.domian.Book;  **import** com.bailiban.service.BookService;  **import** com.bailiban.service.impl.BookServiceImpl;  /\*\*  \* Servlet implementation class BookListServlet  \*/  @WebServlet("/book/list")  **public** **class** BookListServlet **extends** HttpServlet {  **private** **static** **final** **long** ***serialVersionUID*** = 1L;      **protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {    //创建service  BookService bookService = **new** BookServiceImpl();    //查询所有书籍  List<Book> list = bookService.selecAll();    //将数据放入request域  request.setAttribute("list", list);  //转发到 book-list.jsp页面显示  request.getRequestDispatcher("/book-list.jsp").forward(request, response);      }  **protected** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  // **TODO** Auto-generated method stub  doGet(request, response);  }  } |

步骤八：浏览器访问

在浏览器地址输入：<http://localhost:8080/day02_book/login.jsp>



登录失败



登录成功 显示图书列表



1. 完成图书列表的添加和删除

需求：

1.点击添加书籍，跳转到添加页面，输入数据完成书籍添加后回到图书列表

2.点击删除书籍，删除成功回到图书列表

步骤一：创建添加页面 book-add.jsp

根据提供的静态页面book-add.html创建

步骤二：编写图书 dao层

在dao中 增加 添加 删除方法

|  |
| --- |
| **package** com.bailiban.dao.impl;  **import** java.util.List;  **import** org.springframework.jdbc.core.BeanPropertyRowMapper;  **import** org.springframework.jdbc.core.JdbcTemplate;  **import** com.bailiban.dao.BookDao;  **import** com.bailiban.domian.Book;  **import** com.bailiban.utils.JDBCUtils;  **public** **class** BookDaoImpl **implements** BookDao {  //声明JDBCTemplate对象共用  **private** JdbcTemplate template = **new** JdbcTemplate(JDBCUtils.*getDataSource*());  //查询所有图书  @Override  **public** List<Book> selecAll() {    //查询所有书籍  List<Book> list = template.query("select \* from t\_book", **new** BeanPropertyRowMapper<Book>(Book.**class**));    **return** list;  }  //添加图书  @Override  **public** **void** add(Book book) {    //定义添加sql语句  String sql = "INSERT INTO t\_book VALUES (NULL ,?,?,?,?,?)";  //定义添加参数  Object[] args = {  book.getName(),  book.getPrice(),  book.getAuthor(),  book.getImage(),  book.getPublishDate()    };  template.update(sql,args );    }  //删除图书  @Override  **public** **void** delete(**int** id) {    //定义sql语句  String sql = "DELETE From t\_book WHERE id = ? ";    //调用更新方法  template.update(sql, id);    }      } |

步骤三：编写图书 service层

在service中 增加 添加 删除方法

|  |
| --- |
| **package** com.bailiban.service.impl;  **import** java.util.List;  **import** com.bailiban.dao.BookDao;  **import** com.bailiban.dao.impl.BookDaoImpl;  **import** com.bailiban.domian.Book;  **import** com.bailiban.service.BookService;  **public** **class** BookServiceImpl **implements** BookService {    //创建dao  **private** BookDao bookDao = **new** BookDaoImpl();  //查询所有数据  @Override  **public** List<Book> selecAll() {  // **TODO** Auto-generated method stub  **return** bookDao.selecAll();  }  //添加图书  @Override  **public** **void** add(Book book) {    bookDao.add(book);  }  //根据id删除图书  @Override  **public** **void** delete(**int** id) {    bookDao.delete(id);  }  } |

步骤四：编写图书列表BookAddServle和BookDeleteServlet中的代码

|  |
| --- |
| **package** com.bailiban.servlet;  **import** java.io.IOException;  **import** java.lang.reflect.InvocationTargetException;  **import** java.util.Map;  **import** javax.servlet.ServletException;  **import** javax.servlet.annotation.WebServlet;  **import** javax.servlet.http.HttpServlet;  **import** javax.servlet.http.HttpServletRequest;  **import** javax.servlet.http.HttpServletResponse;  **import** org.apache.commons.beanutils.BeanUtils;  **import** com.bailiban.domian.Book;  **import** com.bailiban.service.BookService;  **import** com.bailiban.service.impl.BookServiceImpl;  @WebServlet("/book/add")  **public** **class** BookAddServlet **extends** HttpServlet {  **private** **static** **final** **long** ***serialVersionUID*** = 1L;      **protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {    //设置请求编码  request.setCharacterEncoding("utf-8");  //获取表单提交所有参数的map集合  Map<String, String[]> map = request.getParameterMap();  //创建图书对象  Book book = **new** Book();  //将参数封装进对象  **try** {  BeanUtils.*populate*(book, map);  } **catch** (IllegalAccessException e) {  // **TODO** Auto-generated catch block  e.printStackTrace();  } **catch** (InvocationTargetException e) {  // **TODO** Auto-generated catch block  e.printStackTrace();  }  //创建service  BookService bookService = **new** BookServiceImpl();  //完成添加  bookService.add(book);  //添加成功跳转到图书列表页面  response.sendRedirect(request.getContextPath()+"/book/list");      }  **protected** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  // **TODO** Auto-generated method stub  doGet(request, response);  }  }  **package** com.bailiban.servlet;  **import** java.io.IOException;  **import** javax.servlet.ServletException;  **import** javax.servlet.annotation.WebServlet;  **import** javax.servlet.http.HttpServlet;  **import** javax.servlet.http.HttpServletRequest;  **import** javax.servlet.http.HttpServletResponse;  **import** com.bailiban.service.BookService;  **import** com.bailiban.service.impl.BookServiceImpl;  @WebServlet("/book/delete")  **public** **class** BookDeleteServlet **extends** HttpServlet {  **private** **static** **final** **long** ***serialVersionUID*** = 1L;  **protected** **void** doGet(HttpServletRequest request, HttpServletResponse response)  **throws** ServletException, IOException {  // 接收数据  String str\_id = request.getParameter("id");  // 将id转换为int  **int** id = Integer.*parseInt*(str\_id);  // 创建service  BookService bookService = **new** BookServiceImpl();  //调用删除方法  bookService.delete(id);  //删除成功跳转到列表页面  response.sendRedirect(request.getContextPath()+"/book/list");  }  **protected** **void** doPost(HttpServletRequest request, HttpServletResponse response)  **throws** ServletException, IOException {  // **TODO** Auto-generated method stub  doGet(request, response);  }  } |