### JAVA第二阶段—DAY19-JAVA案例

1. 过滤器快速入门

步骤：

定义一个类，实现接口Filter

复写方法

配置拦截路径：注解、web.xml

* 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day04\_demo

步骤二：创建过滤器

在src目录下创建包com.bailiban.web.filter,定义类Demo1Filter实现接口Filter，重写方法

|  |
| --- |
| **package** com.bailiban.web.filter;  **import** java.io.IOException;  **import** javax.servlet.Filter;  **import** javax.servlet.FilterChain;  **import** javax.servlet.FilterConfig;  **import** javax.servlet.ServletException;  **import** javax.servlet.ServletRequest;  **import** javax.servlet.ServletResponse;  **public** **class** Demo1Filter **implements** Filter {  @Override  **public** **void** destroy() {  // **TODO** Auto-generated method stub    }  @Override  **public** **void** doFilter(ServletRequest request, ServletResponse response, FilterChain chain)  **throws** IOException, ServletException {  System.***out***.println("过滤器执行了，打劫啦!!!");  //放行  chain.doFilter(request, response);    }  @Override  **public** **void** init(FilterConfig filterConfig) **throws** ServletException {  // **TODO** Auto-generated method stub    }  } |

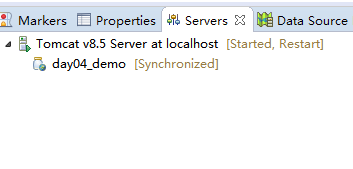
步骤三：配置拦截路径

|  |
| --- |
| //使用注解配置  @WebFilter("/\*")  public class Demo1Filter implements Filter |

Web.xml方式配置

|  |
| --- |
| <filter>  <filter-name>demo1</filter-name>  <filter-class>com.bailiban.web.filter.Demo1Filter</filter-class>  </filter>  <filter-mapping>  <filter-name>demo1</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping> |

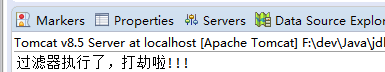
步骤四：将项目部署到Tomcat,启动服务器



步骤五：浏览器访问

在浏览器地址输入：http://localhost:8080/day04\_demo/index.jsp

控制台打印：



1. 使用过滤器解决图书管理项目全站字符乱码问题

步骤：

请求乱码：

get方式：tomcat 8 已经将get方式乱码问题解决了

post方式 ：request.setCharacterEncoding("utf-8");

响应的乱码问题：response.setCharacterEncoding("utf-8")。

* 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

复制day03\_book 取名 day04\_book

步骤二：创建过滤器解决全站字符乱码问题

在src目录下创建包com.bailiban.filter,定义过滤器EncodingFilter

|  |
| --- |
| **package** com.bailiban.filter;  **import** java.io.IOException;  **import** javax.servlet.Filter;  **import** javax.servlet.FilterChain;  **import** javax.servlet.FilterConfig;  **import** javax.servlet.ServletException;  **import** javax.servlet.ServletRequest;  **import** javax.servlet.ServletResponse;  **import** javax.servlet.annotation.WebFilter;  @WebFilter("/\*")  **public** **class** EncodingFilter **implements** Filter {      **public** **void** destroy() {    }    **public** **void** doFilter(ServletRequest request, ServletResponse response, FilterChain chain) **throws** IOException, ServletException {    //在放行之前设置编码  request.setCharacterEncoding("utf-8");  response.setCharacterEncoding("utf-8");  //放行  chain.doFilter(request, response);    }  **public** **void** init(FilterConfig fConfig) **throws** ServletException {    }  } |

步骤三：去掉所有Servlet 中解决乱码的代码

1. 使用过滤器对图书管理项目进行权限控制

需求：

访问图书管理的资源。验证其是否登录

如果登录了，则直接放行。

如果没有登录，则跳转到登录页面，提示"您尚未登录，请先登录"。

* 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

基于day04\_book项目

步骤二：创建LoginFilter

在src目录下创建包com.bailiban.filter,定义过滤器LoginFilter

|  |
| --- |
| **package** com.bailiban.filter;  **import** javax.servlet.\*;  **import** javax.servlet.annotation.WebFilter;  **import** javax.servlet.http.HttpServletRequest;  **import** java.io.IOException;  /\*\*  \* 登录验证的过滤器  \*/  @WebFilter("/\*")  **public** **class** LoginFilter **implements** Filter {  **public** **void** doFilter(ServletRequest request, ServletResponse response, FilterChain chain) **throws** ServletException, IOException {  //强制转换  HttpServletRequest req = (HttpServletRequest) request;  //获取资源请求路径  String uri = req.getRequestURI();  //2.判断是否包含登录相关资源路径,要注意排除掉 css/js/图片 等资源  **if**(uri.contains("/login.jsp") || uri.contains("/login") || uri.contains("/css/") || uri.contains("/js/") || uri.contains("/fonts/")|| uri.contains("/img/") ){  //包含，用户就是想登录。放行  chain.doFilter(request, response);  }**else**{  //不包含，需要验证用户是否登录  //3.从获取session中获取user  Object user = req.getSession().getAttribute("u");  **if**(user != **null**){  //登录了。放行  chain.doFilter(request, response);  }**else**{  //如果没有登录 向request域中存放提示信息  request.setAttribute("msg","亲！请先登录再操作!!!");  //转发到login.jsp页面提示用户登录  request.getRequestDispatcher("/login.jsp").forward(request,response);  }  }    }  **public** **void** init(FilterConfig config) **throws** ServletException {  }  **public** **void** destroy() {  }  } |

步骤五：浏览器访问 查看效果

在浏览器地址输入：http://localhost:8080/day04\_book/book-add.jsp



1. 使用过滤器给图书管理项目添加过滤敏不文明用语功能

需求：

1. 对用户录入的数据进行不文明用语过滤

2. 如果是敏感词汇如：傻x、畜生 等，替换为 \*\*\*

* 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

基于day04\_book项目

步骤二：创建LoginFilter

在src目录下创建包com.bailiban.filter,定义过滤器DirtyFilter

|  |
| --- |
| **package** com.bailiban.filter;  **import** java.io.IOException;  **import** java.lang.reflect.InvocationHandler;  **import** java.lang.reflect.Method;  **import** java.lang.reflect.Proxy;  **import** java.util.Arrays;  **import** java.util.List;  **import** java.util.Map;  **import** java.util.Map.Entry;  **import** javax.servlet.Filter;  **import** javax.servlet.FilterChain;  **import** javax.servlet.FilterConfig;  **import** javax.servlet.ServletException;  **import** javax.servlet.ServletRequest;  **import** javax.servlet.ServletResponse;  **import** javax.servlet.annotation.WebFilter;  @WebFilter("/\*")  **public** **class** DirtyFilter **implements** Filter {    **public** **void** destroy() {    }    **public** **void** doFilter(ServletRequest request, ServletResponse response, FilterChain chain) **throws** IOException, ServletException {    //定义不文明词汇集合  List<String> list = Arrays.*asList*("傻x","畜生");  //创建代理对象  ServletRequest proxy\_request = (ServletRequest) Proxy.*newProxyInstance*(request.getClass().getClassLoader(), request.getClass().getInterfaces(),**new** InvocationHandler() {    @Override  **public** Object invoke(Object proxy, Method method, Object[] args) **throws** Throwable {  //增强getParameterMap方法  //判断是否是getParameter方法  **if**(method.getName().equals("getParameterMap")){  //增强返回值  //获取返回值    Map<String, String[]> map = (Map<String, String[]> ) method.invoke(request,args);    **for** (Entry<String, String[]> entry : map.entrySet()) {    //获取数据数组中的第一个元素  String value = entry.getValue()[0];  //判断数据数组中第一个元素是否包含敏感词汇  **for** (String dirty : list) {  //如果包含  **if**(value.contains(dirty)) {  //将敏感词替换为\*\*\*  entry.getValue()[0]= value.replace(dirty, "\*\*\*");  }  }  }  //返回处理好的map  **return** map;  }  **return** method.invoke(request,args);    }  });      chain.doFilter(proxy\_request, response);  }  **public** **void** init(FilterConfig fConfig) **throws** ServletException {  // **TODO** Auto-generated method stub  }  } |

步骤三：浏览器访问查看效果

在浏览器地址输入：http://localhost:8080/day04\_book/book/updateUI?id=1





1. 定义一个ServletContextListener监听器 ,监听ServletContext对象的创建和销毁,并加载类路径下的配置文件

* 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：创建一个类，实现指定的监听器接口

定义一个类MyServletContextListener，实现ServletContextListener接口

重写方法

|  |
| --- |
| **package** com.bailiban.web.listener;  **import** java.io.FileInputStream;  **import** java.io.FileNotFoundException;  **import** javax.servlet.ServletContext;  **import** javax.servlet.ServletContextEvent;  **import** javax.servlet.ServletContextListener;  **import** javax.servlet.annotation.WebListener;  /\*\*  \*  \* **@author** my  \* 定义一个ServletContextListener 监听器  \* 监听ServletContext的创建和销毁  \*/  @WebListener  **public** **class** MyServletContextListener **implements** ServletContextListener {    /\*  \* 监听 ServletContext对象的创建  \* ServletContext对象是在服务器启动后自动创建  \* 该方法在服务器启动后自动调用  \*  \*/  @Override  **public** **void** contextInitialized(ServletContextEvent sce) {    System.***out***.println("ServletContext对象被创建了。。。");  //加载类路径 classpath下的配置文件  //得到servletContext对象  ServletContext servletContext = sce.getServletContext();  //获取配置参数  String contextConfigLocation = servletContext.getInitParameter("contextConfigLocation");  //获取真实路径  String realPath = servletContext.getRealPath(contextConfigLocation);  System.***out***.println(realPath);  //将文件加载进内存  **try** {  FileInputStream fis = **new** FileInputStream(realPath);  System.***out***.println(fis) ;  } **catch** (FileNotFoundException e) {  // **TODO** Auto-generated catch block  e.printStackTrace();  }    }  /\*  \* 在服务器关闭后，ServletContext对象被销毁。  \* 当服务器正常关闭后该方法被调用  \*  \*/  @Override  **public** **void** contextDestroyed(ServletContextEvent sce) {    System.***out***.println("ServletContext对象被销毁了。。。");    }      } |

配置文件

|  |
| --- |
| <!-- <listener>  <listener-class>com.bailiban.web.listener.MyServletContextListener</listener-class>  </listener> -->  <!--配置初始化参数 -->  <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>/WEB-INF/classes/applicationContext.xml</param-value>  </context-param> |

步骤三：查看效果



1. 文件上传快速入门

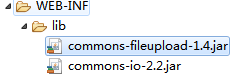
* 参考答案

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day04\_upload

导入文件上传相关jar包



步骤二：定义JSP页面

在web根目录WebContent 中创建upload.jsp页面

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>文件上传</title>  </head>  <body>  <form action=*"*${pageContext.request.contextPath}*/upload"* enctype=*"multipart/form-data"* method=*"post"*>  <input type=*"text"* name=*"username"* /><br/>  <input type=*"file"* name=*"pic"* /> <br/>  <input type=*"submit"* value=*"上传"* />    </form>  </body>  </html> |

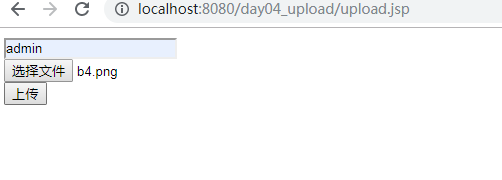
步骤三：

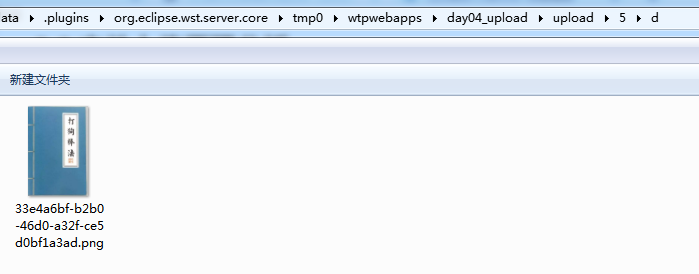
在src创建包com.bailiban.web.servlet，在包中创建UploadServlet

|  |
| --- |
| **package** com.bailiban.web.servlet;  **import** java.io.File;  **import** java.io.IOException;  **import** java.util.List;  **import** java.util.UUID;  **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.fileupload.FileItem;  **import** org.apache.commons.fileupload.disk.DiskFileItemFactory;  **import** org.apache.commons.fileupload.servlet.ServletFileUpload;  @WebServlet("/upload")  **public** **class** UploadServlet **extends** HttpServlet {  **private** **static** **final** **long** ***serialVersionUID*** = 1L;  **protected** **void** doGet(HttpServletRequest request, HttpServletResponse response)  **throws** ServletException, IOException {  // 创建一个DiskFileItemfactory工厂类  DiskFileItemFactory factory = **new** DiskFileItemFactory();  // 创建一个ServletFileUpload核心对象  ServletFileUpload sfu = **new** ServletFileUpload(factory);  // 解决上传文件名中文乱码  sfu.setHeaderEncoding("utf-8");  // 解析request对象，并得到一个表单项的集合  **try** {  List<FileItem> fileItems = sfu.parseRequest(request);  // 遍历表单项数据  **for** (FileItem fileitem : fileItems) {  // 如果是普通表单项  **if** (fileitem.isFormField()) {  // 如果不是  // 获取该控件的value值  String value = fileitem.getString("utf-8");  // 获取该控件的name属性值  String name = fileitem.getFieldName();  System.***out***.println(name + "--" + value);  } **else** {  // 上传文件  uploadFile(fileitem);  }  }  } **catch** (Exception e) {  // **TODO** Auto-generated catch block  e.printStackTrace();  }  }  **protected** **void** doPost(HttpServletRequest request, HttpServletResponse response)  **throws** ServletException, IOException {  // **TODO** Auto-generated method stub  doGet(request, response);  }  // 将上传文件表单项封装  **private** **void** uploadFile(FileItem fileItem) {  // 如果上传表单项  // 得到文件输入流  // 创建物理目录路径  String realPath = **this**.getServletContext().getRealPath("/upload");  // 根据该路径创建一个目录对象  File dir = **new** File(realPath);  **if** (!dir.exists()) {  dir.mkdirs();// 创建一个指定的目录  }  // 得到上传的名子  String filename = fileItem.getName();// 美女.jpg  **if** (filename != **null**) {  // 得到文件后缀  String extend = filename.substring(filename.indexOf("."));  System.***out***.println(extend);  // 重写生成一个唯一的文件名  filename = UUID.*randomUUID*() + extend;  }  // 目录打撒  String childDir = getChildDir(filename);  // 上传文件,自动删除临时文件  **try** {  fileItem.write(**new** File(realPath, childDir + "/" + filename));  } **catch** (Exception e) {  // **TODO** Auto-generated catch block  e.printStackTrace();  }  }  // 目录打撒方法  **private** String getChildDir(String filename) {  // 获取文件名的hashCode  **int** hashCode = filename.hashCode();  // 转换为16进制  String code = Integer.*toHexString*(hashCode);  // 取前2个字符拼成一个子目录  String childDir = code.charAt(0) + "/" + code.charAt(1);  **return** childDir;  }  } |

步骤四：浏览器访问 测试

在浏览器地址输入：http://localhost:8080/day04\_upload/upload.jsp





1. 文件下载快速入门

* 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

基于项目day04\_upload

步骤二：定义JSP页面

在web根目录WebContent 中创建download.jsp页面

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>文件下载</title>  </head>  <body>  <a href=*"*${pageContext.request.contextPath}*/download?filename=帅哥.jpg"*>下载图片</a>  </body>  </html> |

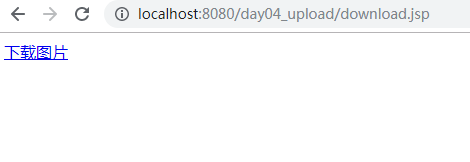
步骤三：

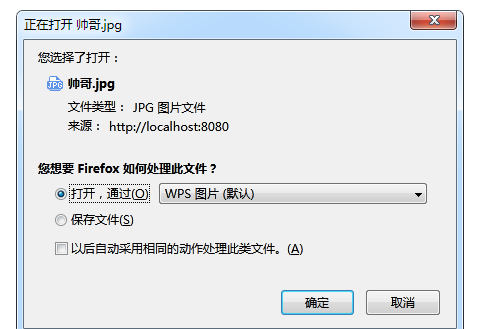
在src创建包com.bailiban.web.servlet，在包中创建DownloadServlet

|  |
| --- |
| **package** com.bailiban.web.servlet;  **import** java.io.FileInputStream;  **import** java.io.IOException;  **import** java.net.URLEncoder;  **import** javax.servlet.ServletContext;  **import** javax.servlet.ServletException;  **import** javax.servlet.ServletOutputStream;  **import** javax.servlet.annotation.WebServlet;  **import** javax.servlet.http.HttpServlet;  **import** javax.servlet.http.HttpServletRequest;  **import** javax.servlet.http.HttpServletResponse;  **import** sun.misc.BASE64Encoder;  @WebServlet("/download")  **public** **class** DownloadServlet **extends** HttpServlet {    **private** **static** **final** **long** ***serialVersionUID*** = 1L;  **protected** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  //获取请求参数，文件名称  String filename = request.getParameter("filename");  //使用字节输入流加载文件进内存  //找到文件服务器路径  ServletContext servletContext = **this**.getServletContext();  String realPath = servletContext.getRealPath("/" + filename);  //创建字节流  FileInputStream fis = **new** FileInputStream(realPath);  //设置response的响应头  //设置响应头类型：content-type  String mimeType = servletContext.getMimeType(filename);//获取文件的mime类型  response.setHeader("content-type",mimeType);  //设置响应头打开方式:content-disposition  //觉得中文乱码问题  //1.获取user-agent请求头、  String agent = request.getHeader("user-agent");  //2.根据头信息判定  **if** (agent.contains("MSIE")) {  // IE浏览器  filename = URLEncoder.*encode*(filename, "utf-8");  filename = filename.replace("+", " ");  } **else** **if** (agent.contains("Firefox")) {  // 火狐浏览器  BASE64Encoder base64Encoder = **new** BASE64Encoder();  filename = "=?utf-8?B?" + base64Encoder.encode(filename.getBytes("utf-8")) + "?=";  } **else** {  // 其它浏览器  filename = URLEncoder.*encode*(filename, "utf-8");  }  response.setHeader("content-disposition","attachment;filename="+filename);  //将输入流的数据写出到输出流中  ServletOutputStream sos = response.getOutputStream();  **byte**[] buff = **new** **byte**[1024 \* 8];  **int** len = 0;  **while**((len = fis.read(buff)) != -1){  sos.write(buff,0,len);  }  fis.close();  sos.close();  }  **protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  **this**.doPost(request,response);  }  } |

步骤四：浏览器访问 测试

在浏览器地址输入：http://localhost:8080/day04\_upload/download.jsp





1. 完善图书管理系统的图书添加，增加图书封面的上传功能

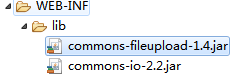
* 参考答案

实现此案例需要按照如下步骤进行。

步骤一：创建JavaEE项目

基于day04\_book项目,创建day05\_book项目

导入文件上传jar包



步骤二：修改添加页面form表单满足上传文件的要求

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>图书添加</title>  <!--导入框架样式文件-->  <link rel=*"stylesheet"* href=*"*${pageContext.request.contextPath}*/css/bootstrap.min.css"* />  <script type=*"text/javascript"* src=*"*${pageContext.request.contextPath}*/js/jquery1.12.4.min.js"*></script>  <script type=*"text/javascript"* src=*"*${pageContext.request.contextPath}*/js/bootstrap.min.js"*></script>  </head>  <body>  <div class=*"container"*>  <div class=*"panel panel-default"*>  <div class=*"panel-heading"*>  <h3 class=*"panel-title"*>图书添加</h3>  </div>  <div class=*"panel-body"*>  <form class=*"form-horizontal"* enctype=*"multipart/form-data"* method=*"post"* action=*"*${pageContext.request.contextPath}*/book/add"*>  <div class=*"form-group"*>  <label for=*"name"* class=*"col-sm-2 control-label"*>图书名称</label>  <div class=*"col-sm-10"*>  <input type=*"text"* class=*"form-control"* name=*"name"* placeholder=*"请输入图书名称"* required >  </div>  </div>  <div class=*"form-group"*>  <label for=*"inputEmail3"* class=*"col-sm-2 control-label"*>图书价格</label>  <div class=*"col-sm-10"*>  <input type=*"text"* class=*"form-control"* name=*"price"* placeholder=*"请输入图书价格"* required >  </div>  </div>  <div class=*"form-group"*>  <label for=*"inputEmail3"* class=*"col-sm-2 control-label"*>图书作者</label>  <div class=*"col-sm-10"*>  <input type=*"text"* class=*"form-control"* name=*"author"* placeholder=*"请输入图书作者"* required >  </div>  </div>  <div class=*"form-group"*>  <label for=*"inputEmail3"* class=*"col-sm-2 control-label"*>出版日期</label>  <div class=*"col-sm-10"*>  <input type=*"date"* class=*"form-control"* name=*"publishDate"* placeholder=*"请输入出版日期"* required>  </div>  </div>  <div class=*"form-group"*>  <label for=*"inputEmail3"* class=*"col-sm-2 control-label"*>图书图片</label>  <div class=*"col-sm-10"*>  <input type=*"file"* class=*"form-control"* name=*"image"* placeholder=*"请输入图书图片"* required>  </div>  </div>    <div class=*"form-group"*>  <div class=*"col-sm-offset-2 col-sm-10"*>  <button type=*"submit"* class=*"btn btn-default"*>添加</button>  </div>  </div>  </form>  </div>  </div>  </div>  </body>  </html> |

步骤四：修改添加servlet

|  |
| --- |
| **package** com.bailiban.servlet;  **import** java.io.File;  **import** java.io.IOException;  **import** java.lang.reflect.InvocationTargetException;  **import** java.util.Date;  **import** java.util.HashMap;  **import** java.util.List;  **import** java.util.Map;  **import** java.util.UUID;  **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** org.apache.commons.fileupload.FileItem;  **import** org.apache.commons.fileupload.disk.DiskFileItemFactory;  **import** org.apache.commons.fileupload.servlet.ServletFileUpload;  **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 {    // 创建一个DiskFileItemfactory工厂类  DiskFileItemFactory factory = **new** DiskFileItemFactory();  // 创建一个ServletFileUpload核心对象  ServletFileUpload sfu = **new** ServletFileUpload(factory);  // 解决上传文件名中文乱码  sfu.setHeaderEncoding("utf-8");  // 解析request对象，并得到一个表单项的集合  //用于封装普通表单项数据的map  Map<String, String[]> map = **new** HashMap<String, String[]>();  **try** {  List<FileItem> fileItems = sfu.parseRequest(request);  // 遍历表单项数据  **for** (FileItem fileitem : fileItems) {  // 如果是普通表单项  **if** (fileitem.isFormField()) {  // 获取该控件的value值  String value = fileitem.getString("utf-8");  // 获取该控件的name属性值  String name = fileitem.getFieldName();  //将数据添加到map  map.put(name, **new** String[] {value});  } **else** {  // 上传文件  String filename = uploadFile(fileitem);  //设置图片表单项的数据到map中  map.put(fileitem.getFieldName(), **new** String[] {filename});  }  }  //创建图书对象  Book book = **new** Book();  //将 map数据封装进对象  BeanUtils.*populate*(book, map);  //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");  } **catch** (Exception e) {  // **TODO** Auto-generated catch block  e.printStackTrace();  }    }  // 将上传文件表单项封装  //返回文件路径  **private** String uploadFile(FileItem fileItem) {  // 如果上传表单项  // 创建物理目录路径  String realPath = **this**.getServletContext().getRealPath("/img");  // 根据该路径创建一个目录对象  File dir = **new** File(realPath);  **if** (!dir.exists()) {  dir.mkdirs();// 创建一个指定的目录  }  // 得到上传的名子  String filename = fileItem.getName();// 美女.jpg  **if** (filename != **null**) {  // 得到文件后缀  String extend = filename.substring(filename.indexOf("."));  System.***out***.println(extend);  // 重写生成一个唯一的文件名  filename = **new** Date().getTime() + extend;  }  // 目录打撒  String childDir = getChildDir(filename);  //得到新的文件名  filename = childDir + "/" + filename;  **try** {  // 上传文件,自动删除临时文件  fileItem.write(**new** File(realPath, filename));  **return** filename;    } **catch** (Exception e) {  // **TODO** Auto-generated catch block  e.printStackTrace();  **return** **null**;  }  }  // 目录打撒方法  **private** String getChildDir(String filename) {  // 获取文件名的hashCode  **int** hashCode = filename.hashCode();  // 转换为16进制  String code = Integer.*toHexString*(hashCode);  // 取前2个字符拼成一个子目录  String childDir = code.charAt(0) + "/" + code.charAt(1);  **return** childDir;  }  **protected** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  // **TODO** Auto-generated method stub  doGet(request, response);  }  } |

步骤七：浏览器访问

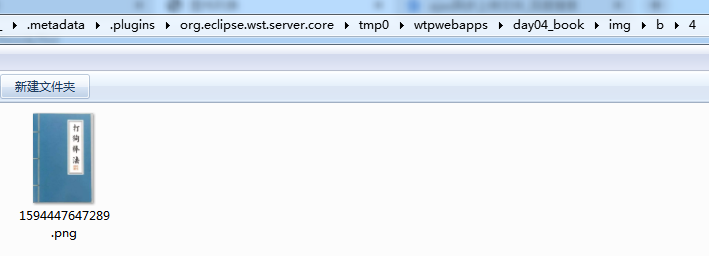
在浏览器地址输入：[http://localhost:8080/day04\_book/book-add.jsp](http://localhost:8080/day02_book/login.jsp)



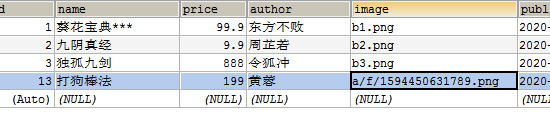
添加



服务器上图片



数据库中



1. 在图书列表增加图书封面下载功能

* 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

基于day05\_book项目

步骤二：给图书列表添加下载按钮

|  |
| --- |
| <%@page import=*"com.bailiban.domain.Book"*%>  <%@page import=*"java.util.List"*%>  <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*  pageEncoding=*"UTF-8"*%>  <!-- 使用taglib 指令导入JSTL标签库 -->  <%@taglib prefix=*"c"* uri=*"http://java.sun.com/jsp/jstl/core"* %>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"UTF-8"*>  <title>图书列表</title>  <!--导入框架样式文件-->  <link rel=*"stylesheet"* href=*"*${pageContext.request.contextPath}*/css/bootstrap.min.css"* />  <script type=*"text/javascript"* src=*"*${pageContext.request.contextPath}*/js/jquery1.12.4.min.js"*></script>  <script type=*"text/javascript"* src=*"*${pageContext.request.contextPath}*/js/bootstrap.min.js"*></script>  </head>  <body>  <div class=*"container"*>  <!-- 从session取出用户信息 显示欢迎信息 -->  <h1>欢迎您:${u.username}</h1>  <div class=*"panel panel-default"*>  <div class=*"panel-heading clearfix"*>  <a href=*"*${pageContext.request.contextPath}*/book-add.jsp"* target=*"main"* class=*"btn btn-primary pull-right"* >添加书籍</a>  </div>    <table class=*"table table-bordered table-hover"*>  <tr class=*"success"*>  <th>编号</th>  <th>书名</th>  <th>图片</th>  <th>价格</th>  <th>作者</th>  <th>出版日期</th>  <th>操作</th>  </tr>  <!-- 使用foreach标签 遍历 -->  <c:forEach items=*"*${list}*"* var=*"book"*>  <tr>  <td>${book.id}</td>  <td>${book.name}</td>  <td ><img src=*"*${pageContext.request.contextPath}*/img/*${book.image}*"* width=*"50"* /></td>  <td>${book.price}</td>  <td>${book.author}</td>  <td>${book.publishDate}</td>  <td>  <a class=*"btn btn-default btn-sm"* href=*"*${pageContext.request.contextPath}*/book/updateUI?id=*${book.id}*"*>修改</a>&nbsp;  <a class=*"btn btn-default btn-sm"* onclick="return confirm('确定删除吗？')" href=*"*${pageContext.request.contextPath}*/book/delete?id=*${book.id}*"*>删除</a>&nbsp;  <a class=*"btn btn-success btn-sm"* onclick="return confirm('确定下载吗？')" href=*"*${pageContext.request.contextPath}*/book/picDownload?filename=*${book.image}*"*>图书封面下载</a>  </td>  </tr>  </c:forEach>    </table>  </div>  </div>  </body>  </html> |

步骤四：创建图图片下载Servlet

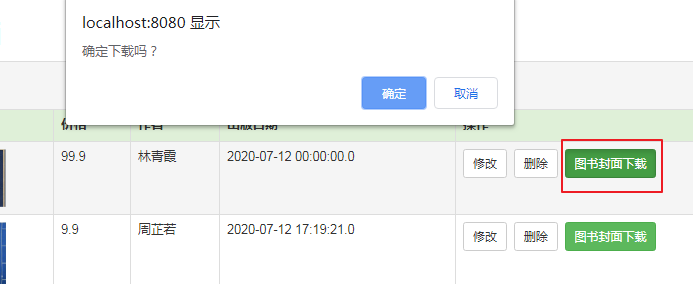
|  |
| --- |
| **package** com.bailiban.servlet;  **import** java.io.FileInputStream;  **import** java.io.IOException;  **import** java.net.URLEncoder;  **import** javax.servlet.ServletContext;  **import** javax.servlet.ServletException;  **import** javax.servlet.ServletOutputStream;  **import** javax.servlet.annotation.WebServlet;  **import** javax.servlet.http.HttpServlet;  **import** javax.servlet.http.HttpServletRequest;  **import** javax.servlet.http.HttpServletResponse;  **import** sun.misc.BASE64Encoder;  /\*\*  \*图书封面下载  \*/  @WebServlet("/book/picDownload")  **public** **class** BookPicDownLoadServlet **extends** HttpServlet {  **private** **static** **final** **long** ***serialVersionUID*** = 1L;    /\*\*  \* **@see** HttpServlet#HttpServlet()  \*/  **public** BookPicDownLoadServlet() {  **super**();  // **TODO** Auto-generated constructor stub  }  **protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  //获取请求参数，文件名称  String filename = request.getParameter("filename");  //使用字节输入流加载文件进内存  //找到文件服务器路径  ServletContext servletContext = **this**.getServletContext();  String realPath = servletContext.getRealPath("/img/" + filename);  //创建字节流  FileInputStream fis = **new** FileInputStream(realPath);  //设置response的响应头  //设置响应头类型：content-type  String mimeType = servletContext.getMimeType(filename);//获取文件的mime类型  response.setHeader("content-type",mimeType);  //设置响应头打开方式:content-disposition  //觉得中文乱码问题  //1.获取user-agent请求头  String agent = request.getHeader("user-agent");  //2.根据头信息判定  **if** (agent.contains("MSIE")) {  // IE浏览器  filename = URLEncoder.*encode*(filename, "utf-8");  filename = filename.replace("+", " ");  } **else** **if** (agent.contains("Firefox")) {  // 火狐浏览器  BASE64Encoder base64Encoder = **new** BASE64Encoder();  filename = "=?utf-8?B?" + base64Encoder.encode(filename.getBytes("utf-8")) + "?=";  } **else** {  // 其它浏览器  filename = URLEncoder.*encode*(filename, "utf-8");  }  response.setHeader("content-disposition","attachment;filename="+filename);  //将输入流的数据写出到输出流中  ServletOutputStream sos = response.getOutputStream();  **byte**[] buff = **new** **byte**[1024 \* 8];  **int** len = 0;  **while**((len = fis.read(buff)) != -1){  sos.write(buff,0,len);  }  fis.close();  sos.close();  }  **protected** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  // **TODO** Auto-generated method stub  doGet(request, response);  }  } |

步骤七：浏览器访问

点击下载按钮



弹出确认下载框



下载成功

