山东大学 计算机科学与技术 学院

现代软件开发技术 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：2020001300 | 姓名： | | 班级： |
| 实验题目：页面信息国际化、美化、表单数据校验、文件上传下载 | | | |
| 实验学时：2学时 | | 实验日期： 2023-10-31 | |
| 实验目的：  1.能够使用Spring Boot与Thymeleaf实现页面信息国际化  2.能够对表单数据进行校验  3.能够使用BootStrap美化前端页面，处理JSON数据；  4.能够使用Spring Boot进行文件上传与下载 | | | |
| 硬件环境： | | | |
| 软件环境：  操作系统：windows 10  开发平台与工具：IDEA开发平台、jdk-11.0.1、apache-tomcat-9.0.37 | | | |
| 实验内容与问题回答：  **1.练习Spring Boot与Thymeleaf实现页面信息国际化**  **（1）项目ch5-1, 分析src/main/resources/i18n三个子文件夹中的语言资源文件；**  **将鼠标移到相应的utf-8编码上，会显示中文。**  **Q：其中都有哪些文件夹？文件夹中有哪些语言资源文件？**  **（请查询字符的utf-8编码）**  主要的文件夹有admin、before、common三个文件夹。.properties文件就是对应的语言资源文件。  **（2）分析src/main/resources/中的application.porpertie资源属性文件，**  **Q:spring.messages.basename的设置是？**  代码分析：  server.servlet.context-path=/ch5\_1：这行配置指定了Servlet上下文路径。在Web应用中，这将是应用程序的URL根路径。在这里，它被设置为/ch5\_1，这意味着应用将在URL中添加/ch5\_1作为根路径。  spring.messages.basename：这行配置是与国际化（i18n）相关的。它指定了用于查找国际化消息的基本名称。在Spring Boot应用程序中，通常会有不同的消息文件用于不同的模块或部分，这些文件的基本名称在这里列出。  i18n/admin/adminMessages：这是用于管理员模块的消息文件的基本名称。  i18n/before/beforeMessages：这是用于"before"模块的消息文件的基本名称。  i18n/common/commonMessages：这是用于通用消息的消息文件的基本名称。  这些消息文件通常包含不同语言或区域的消息文本，以便应用程序能够根据用户的语言设置提供本地化的消息。这些配置通常与Spring Boot的国际化支持一起使用，允许应用程序提供多语言的用户界面。  spring.messages.basename的设置如下：    **（3）分析com.ch.ch5\_1包中的语言区域配置文件LocalConfig.java；**  **分析其实现的WebMvcConfigurer接口中的localeResolver()方法**  **和拦截器localeChangeInterceptor()及拦截器注册addInterceptors(InterceptorRegistry registry)**  **Q:拦截器中拦截了什么？**  localeResolver()方法创建并返回一个SessionLocaleResolver对象，它用于根据用户的会话设置语言区域。默认情况下，将语言区域设置为中国（Locale.CHINA），但用户可以在应用程序中的会话期间更改语言。  拦截器拦截了用户的语言变更请求，它拦截了用户请求中的语言变更参数。拦截器的作用是捕获特定类型的请求，执行某些操作，然后将请求继续传递给下一个处理程序（通常是控制器）。在这种情况下，LocaleChangeInterceptor 捕获语言变更请求，更改用户的语言区域，然后将请求继续传递给应用程序的其他部分，以根据新的语言设置提供相应的用户界面。这是一种支持国际化的机制，让用户能够以他们喜欢的语言使用应用程序。  **（4）分析com.ch.ch5\_1.controller包下的控制器I18nTestController.java文件；分析其中的@RequestMapping**  **Q: /i18n/first 对应哪个视图文件？**  在该文件中，@RequestMapping 注解用于标识处理器方法（Controller 中的方法）与特定 URL 路径之间的映射关系。它将指定的 URL 路径映射到相应的处理器在本项目中"/i18n/first" 对应的视图文件是src/main/resources/templates/i18n/first.html视图文件。  **（5）打开src/main/resources/templates/i18n下的视图网页文件：**  **分析before.html 前台页面，Q：th:text="#{XXXX}" 中XXXX来自哪儿？**  前台页面 before.html 中的 th:text="#{before}" 中的 before 消息键（Message Key）通常来自您应用的国际化资源文件（通常是 .properties 文件）。这个消息键定义在国际化资源文件中，根据用户的语言设置，Thymeleaf会查找相应的资源文件以获取消息文本。本项目中，程序会查找src/main/resources/i18n/before文件夹下的 beforeMessages\_zh\_CN.properties 文件，并在其中查找 before 消息键以获取相应的消息文本。因此，th:text="#{before}" 中的 before 是来自于 beforeMessages\_zh\_CN.properties 文件中的消息键。  **分析admin.html 后台页面； Q: th:href="@{/i18n/first}" th:text="#{return} 的含义是什么？**  th:href="@{/i18n/first}"：这个部分使用 Thymeleaf 的 th:href 属性，用于设置链接的 URL。具体地，它将链接指向 /i18n/first，这是一个相对路径。在 Spring Boot 中，这通常表示映射到控制器 I18nTestController 的 testI18n 方法。  th:text="#{return}"：这个部分使用 Thymeleaf 的 th:text 属性，用于设置元素的文本内容。文本内容将从国际化资源文件中获取，具体的消息键为 return。  **分析first.html 首页页面； Q: th:href="@{/i18n/first(locale='zh\_CN')}" local对应的是什么？**  th:href="@{/i18n/first(locale='zh\_CN')}" 中的 locale='zh\_CN' 是自定义的参数，用于指定切换到中文（简体中文）的语言环境。这里的 locale 不是 Thymeleaf 的内建属性，而是在应用中自定义的。  **（6）测试运行：http://localhost:8080/ch5\_1/i18n/first，**  **Q: 点击其中的超级链接，尝试切换语言版本，分析其执行流程。**  在浏览器中输入<http://localhost:8080/ch5_1/i18n/first>后进行测试：      **2.练习使用类型注解进行表单验证。**  **（1）项目ch5\_1，分析src/main/java下，com.ch.ch5\_1.model包下的实体模型类Goods.java**  **Q: 其中有哪几个属性？各自加了什么验证注解？**  **gname 属性：**  @NotBlank@Length  **gprice 属性：**  @Range  **（2）分析src/main/java下，com.ch.ch5\_1.controller包中的TestValidatorController.java的代码**  **Q: @RequestMapping("/testValidator")注解的方法中，向Model中添加了什么对象实例？最后转向什么页面？**  在TestValidatorController.java文件中，在@RequestMapping("/testValidator") 注解的方法 testValidator 中，通过 @ModelAttribute("goodsInfo") 向 Model 中添加了名为 "goodsInfo" 的对象实例，该实例的类型为 Goods 类。最后，这个方法会转向名为 "testValidator" 的页面。  **（3）分析src/main/resources/templates中的testValidator.html页面代码**  **Q:表单标签<form ...>中绑定的对象是？表单提交到哪里？每个对象属性之后的 th:errors=...所代表的含义是什么？**  在testValidator.html页面代码中，表单标签 <form> 中绑定的对象是 ${goodsInfo}。该表单提交到地址 @{/add}。  **（4）分析src/main/java下，com.ch.ch5\_1.controller包中的 TestValidatorController.java 的代码**  **Q:@RequestMapping(value="/add")注解的方法中，起到关键性校验检查的注解是？绑定校验结果的对象是？**  @RequestMapping(value="/add") 注解的方法中，关键性校验检查的注解是 @Validated。绑定校验结果的对象是 BindingResult rs对象。  **（5）在浏览器地址栏输入：http://localhost:8080/ch5\_1/testValidator**  **在页面表单中试输入不合法的数据，页面提示什么信息？**    **3.练习基于Thymeleaf与BootStrap的Web开发实例**  **用STS4打开课本项目ch5\_2，**  **（1）分析com.ch.ch5\_2.model 包中的Book.java**  **Q:其中有哪几个属性？**  isbn  price  bname  publishing  author  picture  **（2）分析com.ch.ch5\_2.controller包中的ThymeleafController.java**  **Q:@RequestMapping("/")注解的方法中，分别向Model中添加了什么对象实例？最后转向哪个页面？**  ThymeleafController.java这个文件是一个Spring MVC中的控制器（Controller），处理请求映射为"/"的情况。public String index(Model model) 表示这是一个处理请求的方法，该方法接受一个Model对象作为参数，该对象用于向视图传递数据。在方法内部，创建了一个名为teacherGeng的Book对象，表示一本书的信息，并设置了其各个属性，如ISBN、价格、书名等。创建了一个名为chenHeng的ArrayList，用于存储多本书的信息。然后创建两个Book对象b1和b2，并将它们添加到chenHeng列表中。使用model.addAttribute()方法将teacherGeng对象添加到Model中，属性名为"aBook"。将chenHeng列表添加到Model中，属性名为"books"最后，通过return "index";语句表示，处理完成后将转向名为"index"的视图。  **（3）分析src/main/resources/templates中的页面index.html中的代码，注意其中的：**  **<link rel="stylesheet" th:href="@{css/bootstrap.min.css}" />**  **<!-- 默认访问 src/main/resources/static下的css文件夹-->**  **<link rel="stylesheet" th:href="@{css/bootstrap-theme.min.css}" />**  **Q: 网页中读取的两个变量是？**  **Q: 型如<div class="XXX"> 中的XXX，来自哪里？**  网页中读取的两个变量是链接引用的两个CSS文件：  <link rel="stylesheet" th:href="@{css/bootstrap.min.css}" />：这是引用了名为"bootstrap.min.css"的CSS文件，使用Thymeleaf语法中的@{}表达式，它将被解析为实际的URL。这表示应用会去访问名为"bootstrap.min.css"的CSS文件。通常，这种语法是相对于当前应用的根路径的。  <link rel="stylesheet" th:href="@{css/bootstrap-theme.min.css}" />：类似地，这是引用了名为"bootstrap-theme.min.css"的CSS文件。  **对于<div class="XXX"> 中的XXX，来自哪里这个问题：**  <div class="panel panel-primary"> 中的 "panel panel-primary" 是Bootstrap框架中的样式类。这些样式类定义了面板（panel）的外观，其中"panel-primary"表示这是一个主要样式的面板。  <div class="container"> 中的 "container" 也是Bootstrap框架中的样式类，用于创建一个响应式容器。  <div class="row"> 中的 "row" 也是Bootstrap框架中的样式类，用于创建一个水平的行，通常与列（col）一起使用。  <div class="col-md-4 col-sm-6"> 中的 "col-md-4 col-sm-6" 是Bootstrap框架中的栅格系统样式类，表示在中等屏幕（md）上占用4列，而在小型屏幕（sm）上占用6列。这用于创建响应式的布局。  总之，这些样式类都是来自Bootstrap框架，通过使用这些类可以轻松地实现响应式和美观的界面设计。而Thymeleaf的表达式语法（比如${aBook.bname}）则用于在模板中动态地填充数据。  **（4）查看src/main/resources/static中，/css,/js等静态资源文件，**  **（5）运行com.ch.ch5\_2包中的 Ch52Application.java，run as application。**  **在浏览器地址栏用 http://localhost:8080/ch5\_2/, 测试。**    **4.练习 Spring Boot处理JSON数据 （对比第二章项目ch2\_5）**  **（1）分析src/main/java下，com.ch.ch5\_2.model 包中的实体类Person.java.**  **Q:其中包含哪几个属性？**  pname  password  page  **（2）分析src/main/resources/templates中的页面文件input.html，**  **Q:引入jQuery的代码是哪行？**  **Q: function testJson() { 中，var pname = $("#pname").val(); #pname对应是什么？**  **Q：$.ajax({ 中，发送的有哪5项关键数据变量？接收的数据变量是？**  引入jQuery的是这一行代码：<script type="text/javascript" th:src="@{js/jquery.min.js}"></script> 这行代码引入了jQuery库，它的路径相对于根目录的"js"文件夹。  **（3）分析com.ch.ch5\_2.controller包中的文件：TestJsonController.java中的代码，**  **Q：@RequestMapping("/testJson")注解，响应的请求由哪里发出？**  **Q：@ResponseBody注解的作用？**  **Q：@RequestBody注解在哪里？**  @RequestMapping("/testJson")注解，响应的请求由前端页面的JavaScript代码发出  @ResponseBody 注解告诉 Spring MVC 框架，不要视图解析器来解析返回结果，而是直接将方法的返回值写入 HTTP 响应体中。通常用于返回 JSON 或其他格式的数据。  @RequestBody 注解用于将 HTTP 请求的 body 部分的 JSON 数据绑定到方法的参数上，这里在 testJson 方法的参数中使用了 @RequestBody Person user，表示将接收到的 JSON 数据转换为 Person 类型的对象。这样可以方便地在方法中处理前端传递的 JSON 数据。  **（4）在浏览器地址栏用 http://localhost:8080/ch5\_2/input， 输入数据测试。**  **（5）根据代码分析，分别放开index.html和TestJsonController.java中的注释，测试更多的对象类型。**  **5. 练习Spring Boot文件上传与下载**  **（1）分析ch5\_2项目中的pom.xml中新添加的依赖，**  在pom.xml中新加入的依赖主要是下面这个依赖commons-fileupload。  这部分的作用是引入 commons-fileupload 组件，版本号为 1.3.3。它是一个用于处理文件上传的 Apache Commons 组件，通常在Web应用中用于支持文件上传功能。在 Spring Boot 项目中，它可以与Spring MVC一起使用，用于处理文件上传的相关操作。  **（2）分析src/main/resources/中的application.perpertie资源属性文件中，**  **Q:其中有关文件上传大小控制的属性设置有哪些？**  在上述配置文件中，有关文件上传大小控制的属性设置如下：  server.servlet.context-path=/ch5\_2：  这是设置应用的上下文路径，即访问该应用时的根路径。在这里设置为 /ch5\_2。  spring.servlet.multipart.max-file-size=50MB：  这是设置单个文件上传的最大大小为 50MB。该属性定义了单个文件的最大允许大小。  spring.servlet.multipart.max-request-size=500MB：  这是设置总的上传请求大小为 500MB。该属性定义了整个请求中所有文件大小的最大允许值。  这两个属性的设置是为了限制文件上传的大小，以确保系统在处理文件上传时不会超出预定的阈值，从而保护系统免受潜在的滥用或意外上传的影响。  **（3）分析src/main/resources/templates中的页面文件uploadFile.html，**  **Q:下列表单标签提交的请求地址是？编码类型是？**  **<form class="form-horizontal" action="upload" method="post" enctype="multipart/form-data">**  **Q:两个<input >标签中，对应的属性变量名是？**  **表单提交的请求地址是：**  <form class="form-horizontal" action="upload" method="post" enctype="multipart/form-data"> 中的 action 属性指定了表单提交的请求地址为 "upload"。  **编码类型是：**  <form> 标签中的 enctype="multipart/form-data" 指定了表单数据以二进制形式进行传输，这是用于支持文件上传的一种编码类型。  **两个 <input> 标签中，对应的属性变量名是：**  第一个 <input> 标签：name="description"，对应的属性变量名是 "description"。  第二个 <input> 标签：name="myfile"，对应的属性变量名是 "myfile"。  **（4）分析com.ch.ch5\_2.controller包中的文件上传控制器：TestFileUpload.java，**  **Q：@RequestMapping("/upload")注解的方法中，怎样完成文件上传？**  1.接收请求参数：  使用HttpServletRequest request接收请求对象。  使用@RequestParam("description") String description接收文件描述。  使用@RequestParam("myfile") MultipartFile myfile接收上传的文件，这里的参数名必须与表单中文件上传控件的name属性一致。  2.输出文件描述信息：  打印文件描述信息，示例代码：System.out.println("文件描述：" + description);  3.判断文件是否为空：  使用if(!myfile.isEmpty())判断文件是否为空。  4.获取上传文件信息：  获取上传文件的原始文件名：String fileName = myfile.getOriginalFilename();  5.构建上传文件路径：  使用String path = request.getServletContext().getRealPath("/uploadFiles/");获取上传文件的保存路径。  使用File filePath = new File(path + File.separator + fileName);构建上传文件的完整路径。  6.判断目录是否存在：  使用if(!filePath.getParentFile().exists()) { filePath.getParentFile().mkdirs(); }判断目录是否存在，不存在则创建。  7.将上传文件保存到指定目录：  使用myfile.transferTo(filePath);将上传的文件保存到指定目录。  8.转发到查询文件列表的方法：  使用return "forward:/showDownLoad";将请求转发到查询文件列表的方法，即 /showDownLoad。  **Q：@RequestMapping("/showDownLoad")注解的方法中，怎样完成文件下载？**  使用 String path = request.getServletContext().getRealPath("/uploadFiles/"); 获取文件存储路径使用 File fileDir = new File(path); 构建文件目录对象。使用 File filesList[] = fileDir.listFiles(); 获取文件目录下的文件列表。使用 model.addAttribute("filesList", filesList); 将文件列表信息添加到 Model 中，以便在视图页面中展使用 return "showFile"; 返回视图页面的名称，这里是 "showFile"。在视图页面 "showFile.html" 中，可以通过 Thymeleaf 模板引擎遍历文件列表，并显示每个文件的信息。这样，当用户访问 "/showDownLoad" 路径时，系统会执行 showDownLoad 方法，该方法获取文件列表信息，将其添加到 Model 中，然后返回 "showFile" 视图页面。用户在浏览器中看到的页面将展示可下载文件的列表。  **（5）分析src/main/resources/templates中的页面文件showFile.html，**  **Q：下列代码中 @{download(filename=${file.name}，对应控制器中的哪个方法？**  **<a th:href="@{download(filename=${file.name})}">**  **<span th:text="${file.name}"></span>**  **</a>**  @{download(filename=${file.name})} 对应控制器中的 @RequestMapping("/download") 注解的方法， | | | |
| 思考题解答：  无 | | | |