# 一、传统的文件上传

1. 首先是传统spring的文件上传。
   1. 引入fileupload的依赖

|  |
| --- |
| <**dependency**>  <**groupId**>commons-fileupload</**groupId**>  <**artifactId**>commons-fileupload</**artifactId**>  <**version**>1.3.1</**version**> </**dependency**> <**dependency**>  <**groupId**>commons-io</**groupId**>  <**artifactId**>commons-io</**artifactId**>  <**version**>2.4</**version**> </**dependency**> |

1. jsp页面

|  |
| --- |
| *<%--  Created by IntelliJ IDEA.  User: 86152  Date: 2020/4/4  Time: 9:48  To change this template use File | Settings | File Templates. --%>* <%@ **page contentType**="**text/html;charset=UTF-8**" **language**="**java**" %> <**html**> <**head**>  <**title**>文件上传学习</**title**> </**head**> <**body**>  <**h3**>文件上传</**h3**>   <**form method="post" action="user/fileUploadTest01" enctype="multipart/form-data"**>  传统文件上传：<**input type="file" name="file01"**>  <**br**/>  <**button type="submit"**>传统文件上传</**button**>  </**form**>   <**br**/><**hr**/>   <**form method="post" action="user/fileUploadTest02" enctype="multipart/form-data"**>  sprigMVC文件上传：<**input type="file" name="file02"**>  <**br**/>  <**button type="submit"**>sprigMVC文件上传</**button**>  </**form**>  </**body**> </**html**> |

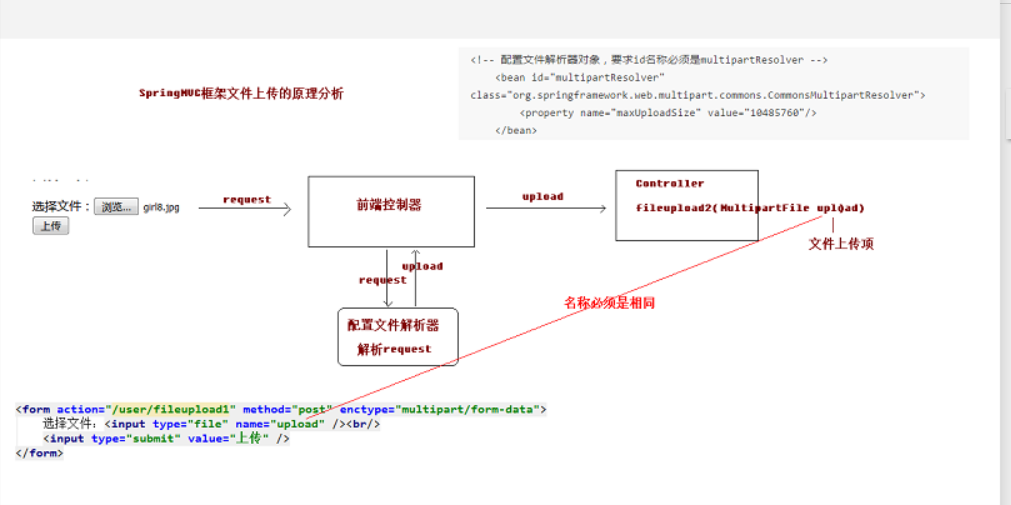
1. 传统方法对应的controller

|  |
| --- |
| **package** com.igeek.shop.controller;  **import** org.apache.commons.fileupload.FileItem; **import** org.apache.commons.fileupload.FileUploadException; **import** org.apache.commons.fileupload.disk.DiskFileItemFactory; **import** org.apache.commons.fileupload.servlet.ServletFileUpload; **import** org.springframework.stereotype.Controller; **import** org.springframework.web.bind.annotation.RequestMapping; **import** org.springframework.web.multipart.MultipartFile;  **import** javax.servlet.http.HttpServletRequest; **import** java.io.File; **import** java.io.IOException; **import** java.util.HashMap; **import** java.util.List; **import** java.util.Map; **import** java.util.UUID;  @Controller @RequestMapping(path = **"/user"**) **public class** FileUploadController {    */\*  \* 传统方法  \* \*/* @RequestMapping(path = **"/fileUploadTest01"**)  **public** String fileUploadTest01(HttpServletRequest request){  System.***out***.println(**"FileUploadController.fileUploadTest01()........"**);   *// 为了有针对性的获取表单中其他普通控件值* Map<String, String> params = **new** HashMap<String, String>();   *// 1.创建磁盘文件项工厂* DiskFileItemFactory factory = **new** DiskFileItemFactory();  *// 第一个参数：缓存区大小 以字节为单位 第二个参数：缓存区的地址 上传的位置* String temp\_path = request.getSession().getServletContext().getRealPath(**"/fileiploads/"**);   *// DiskFileItemFactory factory = new DiskFileItemFactory(1024\*1024,new  // File(temp\_path));* File tempFile = **new** File(temp\_path);  **if** (!tempFile.exists()) {  tempFile.mkdirs();  }   *// 获取服务器存放文件路径* String path = request.getSession().getServletContext().getRealPath(**"/file01/"**);   File uploadFile = **new** File(path);  **if** (!uploadFile.exists()) {  uploadFile.mkdirs();  }   *// 分别设置缓存区大小* factory.setSizeThreshold(1024 \* 1024);*// 1M  // 设置缓存的地址* factory.setRepository(tempFile);   *// 2.创建核心类* ServletFileUpload upload = **new** ServletFileUpload(factory);  *// 将上传文件名进行编码* upload.setHeaderEncoding(**"UTF-8"**);   *// 判断表单的内容是不是文件上传 true---是文件上传 false---普通表单* **boolean** multipartContent = ServletFileUpload.*isMultipartContent*(request);  **if** (multipartContent) {  *// 文件上传* **try** {  List<FileItem> parseRequest = upload.parseRequest(request);  **if** (parseRequest != **null**) {  *// 遍历* **for** (FileItem item : parseRequest) {  *// 判断item对象是不是普通表单对象，还是文件上传的对象* **if** (item.isFormField()) {  *// 获取表单的name属性值* String fieldName = item.getFieldName();  *// 获取表单中输入的值* String fieldValue = item.getString(**"UTF-8"**);*// 对表单中输入的值进行编码* System.***out***.println(fieldName + **"="** + fieldValue);   params.put(fieldName, fieldValue);  } **else** {  *// item是一个文件上传的对象  // 先获取文件名* String fileName = item.getName();  *// 获取文件内容  // InputStream in = item.getInputStream();  /\*  \* OutputStream out = new FileOutputStream(new File(path+"/" + fileName)); //拷贝  \* IOUtils.copy(in, out);  \*/* String newFileName = UUID.*randomUUID*().toString();   **int** lastIndexOfDot = fileName.lastIndexOf(**"."**);  **if** (lastIndexOfDot != -1) {  *// 有后缀,则要得到后缀* String subfix = fileName.substring(lastIndexOfDot);  newFileName = newFileName + subfix;  }   *// 把item中的文件流写入到目标位置* item.write(**new** File(uploadFile, newFileName));  *// 关闭流  // in.close();  /// out.close();  // 应该将缓存区的临时文件清空* item.delete();   params.put(item.getFieldName(), **"true"**);  }  }  }  } **catch** (FileUploadException e) {  e.printStackTrace();  } **catch** (Exception e) {  e.printStackTrace();  }   } **else** {  *// 以前的处理表单的方式* request.getParameter(**"name"**);  }    System.***out***.println(**"文件上传成功！！！"**);  **return "success"**;  } } |

# 二、springmvc的文件上传

1. springmvc的文件上传

|  |
| --- |
| **package** com.igeek.shop.controller;  **import** org.apache.commons.fileupload.FileItem; **import** org.apache.commons.fileupload.FileUploadException; **import** org.apache.commons.fileupload.disk.DiskFileItemFactory; **import** org.apache.commons.fileupload.servlet.ServletFileUpload; **import** org.springframework.stereotype.Controller; **import** org.springframework.web.bind.annotation.RequestMapping; **import** org.springframework.web.multipart.MultipartFile;  **import** javax.servlet.http.HttpServletRequest; **import** java.io.File; **import** java.io.IOException; **import** java.util.HashMap; **import** java.util.List; **import** java.util.Map; **import** java.util.UUID;  @Controller @RequestMapping(path = **"/user"**) **public class** FileUploadController {   */\*  \* SpringMVC方法  \* \*/* @RequestMapping(path = **"/fileUploadTest02"**)  **public** String fileUploadTest02(HttpServletRequest request, MultipartFile file02) **throws** IOException { *//file02 必须和页面上的上名字一致 sprigMVC文件上传：<input type="file" name="file02">* System.***out***.println(**"FileUploadController.fileUploadTest02()........"**);   *// 获取服务器存放文件路径* String path = request.getSession().getServletContext().getRealPath(**"/file02\_mvc/"**);  File uploadFile = **new** File(path);  **if** (!uploadFile.exists()) {  uploadFile.mkdirs();  }  String uploadFileName = file02.getOriginalFilename();  *// 获取文件内容   //重命名* String newFileName = UUID.*randomUUID*().toString();   **int** lastIndexOfDot = uploadFileName.lastIndexOf(**"."**);  **if** (lastIndexOfDot != -1) {  *// 有后缀,则要得到后缀* String subfix = uploadFileName.substring(lastIndexOfDot);  newFileName = newFileName + subfix;  }   *//上传文件* file02.transferTo(**new** File(path,newFileName));  System.***out***.println(**"文件上传成功！！！"**);  **return "success"**;  } |



# 三、文件跨服务器上传

1. 文件跨服务器上传。
   1. 新加一个服务器（tomcat 9999）,开一个项目，随便内容，用来存放文件（文件服务器）。最好不要用已经有功能的项目，可能是web.xml里前端控制器有些配置会阻断上传导致404错误。最后改成一个空的项目就好了。
   2. 再打开自己的文件上传的工程服务器（9090），即同时开两个服务器。模拟跨服务器上传。

首先是需要引入依赖：jersey

|  |
| --- |
| *<!--文件跨服务器上传-->* <**dependency**>  <**groupId**>com.sun.jersey</**groupId**>  <**artifactId**>jersey-core</**artifactId**>  <**version**>1.18.1</**version**> </**dependency**> <**dependency**>  <**groupId**>com.sun.jersey</**groupId**>  <**artifactId**>jersey-client</**artifactId**>  <**version**>1.18.1</**version**> </**dependency**> |

1. Jsp

|  |
| --- |
| <**form method="post" action="user/fileUploadTest03" enctype="multipart/form-data"**>  sprigMVC文件跨服务器上传：<**input type="file" name="file03"**>  <**br**/>  <**button type="submit"**>sprigMVC文件跨服务器上传</**button**> </**form**> |

1. controller编写。Client 的包要导对

**import** com.sun.jersey.api.client.Client;

@RequestMapping(path = **"/fileUploadTest03"**)不能重复

文件目录未找到 是409错误

我用自己原有工程作为文件上传的服务器。会出现404错误

|  |
| --- |
| */\*  \* SpringMVC 跨服务器文件上传方法 即开两个服务器（2个tomcat端口不同，同时启动）。  \* 一个服务器是文件服务器，不需要别的功能，接收存放文件就行。9999  \* 另一个服务器进行文件上传。  \* \*/* @RequestMapping(path = **"/fileUploadTest03"**) **public** String fileUploadTest03(HttpServletRequest request, MultipartFile file03) **throws** IOException { *//file02 必须和页面上的上名字一致 sprigMVC文件上传：<input type="file" name="file02">* System.***out***.println(**"FileUploadController.fileUploadTest03()........"**);   *// 获取服务器存放文件路径* String path = **"http://localhost:9999/fileServer/uploads/"**; *//http://localhost:9999/fileServer 放入了浏览器可以访问到项目。/uploads/是存放文件的文件夹。target目录下一定要有不然409错误* String uploadFileName = file03.getOriginalFilename();  *// 获取文件内容   //重命名* String newFileName = UUID.*randomUUID*().toString();   **int** lastIndexOfDot = uploadFileName.lastIndexOf(**"."**);  **if** (lastIndexOfDot != -1) {  *// 有后缀,则要得到后缀* String subfix = uploadFileName.substring(lastIndexOfDot);  newFileName = newFileName + subfix;  }   *//上传文件 跨服务器上传文件  //创建客户端* Client client = Client.*create*();  *//和服务器进行连接* WebResource resource = client.resource(path + newFileName);  *//上传文件* resource.put(file03.getBytes()); *//通过字节传过去* System.***out***.println(**"跨服务器文件上传成功！！！"**);  **return "success"**; } |