课程目标:

1. 显示用户下的app信息
2. 会springmvc实现文件上传
3. 实现app信息的添加
4. 测试显示用户下的app 信息

查询当前用户信息

|  |
| --- |
| @RequestMapping(**"/searchAppInfo"**) **public** String searchAppInfo(AppInfoCondition appInfoCondition, HttpSession session, Model model){  *//appInfoCondition接收表单的条件  //1.获取当前登入的用户编号* DevUser devUser=(DevUser)session.getAttribute(**"loginInfo"**);  appInfoCondition.setDevid(devUser.getId()); *//设置用户编号  //appInfoCondition.setDevid(new Long("1")); //设置固定用户编号  //2.调用业务实现搜索* PageInfo<AppInfo> pageInfo=**appInfoService**.getAppInfoByCondition(appInfoCondition);  *//3.将获取的信息填充到页面* model.addAttribute(**"pageInfo"**,pageInfo);  model.addAttribute(**"condition"**,appInfoCondition);*//回显  //4.返回页面* **return "developer/appinfolist"**; } |

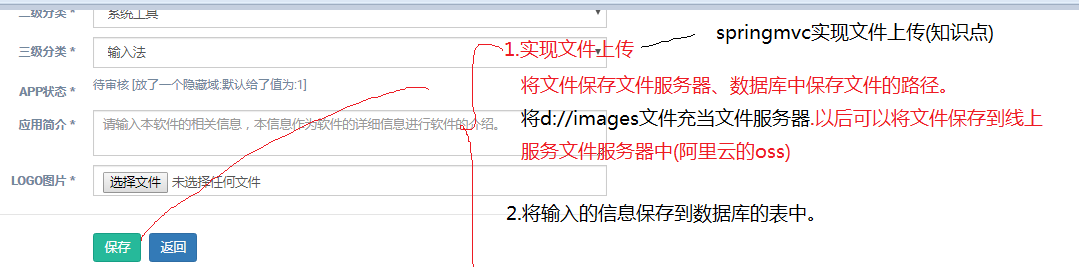
1. 添加app信息页面
2. 制作添加app信息的页面:

略

1. 实现app信息页面的平台、分类的异步加载

略

1. 分析实现添加功能



注意:上传文件保存到文件服务器中,常用文件服务器(fastdfs搭建文件服务器或者选择线上的文件服务器如:阿里云的oss)

三、(重点:)springmvc实现文件上传的步骤：

注意:springmvc文件上传是默认封装的是commons-fileuplad组件

1. 制作文件上传的表单

|  |
| --- |
| <**form class="form-horizontal   action="appinfoaddsave" method="post” enctype="multipart/form-data"**>  略  </form> |

要求:表单必需以post方式提交，且添加**enctype="multipart/form-data"**

1. springmvc控制器实现文件上传的步骤:

2.1:导入文件上传的依赖:commons-fileuplad包

|  |
| --- |
| *<!-- 文件上传 start -->* <**dependency**> <**groupId**>commons-fileupload</**groupId**> <**artifactId**>commons-fileupload</**artifactId**>  <**version**>1.2.1</**version**> </**dependency**> *<!-- 文件上传 end -->* |

2.2:在springmvc配置文件中添加文件上传解析器

|  |
| --- |
| *<!--添加上传文件的支持-->* <**bean id="multipartResolver" class="org.springframework.web.multipart.commons.CommonsMultipartResolver"**>  <**property name="defaultEncoding" value="UTF-8"**/>  <**property name="maxUploadSize" value="5242880"**/> *<!-- 5M -->* </**bean**> |

* 1. 在控制器中处理文件上传(编写代码)

|  |
| --- |
| *//实现app信息的添加 文件上传* @RequestMapping(**"/addAppInfo"**) **public** String addAppInfo(  @RequestParam(value = **"a\_logoPicPath"**,required = **false**) CommonsMultipartFile file,  HttpServletRequest request){  *//一个CommonsMultipartFile对象对应一个文件上传的表单对象  //1.获取文件上传的相关信息  /\*System.out.println("文件名称:"+file.getOriginalFilename());  System.out.println("文件大小:"+file.getSize());  System.out.println("文件类型:"+file.getContentType());\*/  //2.实现文件上传 将文件保存到服务器* **try** {  //获取文件的保存位置  String path= request.getSession().getServletContext()  .getRealPath("upl**oad/logo"**);  *//生成文件的唯一名称* String filename=file.getOriginalFilename(); *//上传文件名* String extname=filename.substring(filename.lastIndexOf(**"."**)); *//扩展名* String saveFileName=System.*currentTimeMillis*()+extname; *//生成的随机文件名  //创建保存的文件对象* File saveFile=**new** File(path+**"/"**+saveFileName); *//保存的文件* file.transferTo(saveFile); *//上传* System.***out***.println(path);  }**catch** (Exception e){  e.printStackTrace();  }   **return ""**; } |

>>>>>理解文件上传文件代码封装

|  |
| --- |
| **public class** FileUploadUtil {  */\*\*  \* 实现文件上传  \** ***@param savePath*** *保存的绝对路径  \** ***@param file*** *上传文件  \** ***@return*** *上传的文件名称,如果是null表示上传失败  \*/* **public static** String uploadFile(String savePath, CommonsMultipartFile file){  **try** {  *//生成唯一文件名* String uploadFileName=file.getOriginalFilename();  String expname=uploadFileName.substring(uploadFileName.lastIndexOf(**"."**));  String fileName=System.*currentTimeMillis*()+expname;  *//上传文件* File saveFile=**new** File(savePath+**"/"**+fileName);  file.transferTo(saveFile); *//上传* **return** fileName;  }**catch** (Exception e){  e.printStackTrace();  }  **return null**;  } }  //利用工具实现文件上传 在控制器中添加方法  @RequestMapping(**"/addAppInfoAction"**) **public** String addAppInfoAction(  HttpServletRequest request,  @RequestParam(value = **"a\_logoPicPath"**,required = **false**) CommonsMultipartFile logoFile){  *//1.实现文件上传  //一个文件域对应一个CommonsMultipartFile类的参数对象* **try** {  *//1.2 上传文件  //获取上传文件的保存位置* String savePath=request.getSession().getServletContext().getRealPath(**"upload/logo"**);  *//上传* String fileName=FileUploadUtil.*uploadFile*(savePath,logoFile);  System.***out***.println(**"保存成功"**);  }**catch** (Exception e){  }  **return ""**; } |

1. 实现app信息的添加
2. 编写dao层

略

1. 编写业务层

接口:

|  |
| --- |
| **public int** addAppInfo(AppInfo appInfo); |

实现:

|  |
| --- |
| @Override **public int** addAppInfo(AppInfo appInfo) {  **return this**.**appInfoMapper**.insertSelective(appInfo); } |

1. 编写控制器层:

上传图片、并保存到数据库

|  |
| --- |
| *//添加app信息* @RequestMapping(**"/addAppInfo"**) **public** String addAppInfo(  AppInfo appInfo,  @RequestParam(value = **"a\_logoPicPath"**) CommonsMultipartFile file,  HttpServletRequest request,  Model model,  HttpSession session){  *//注意:一个上传文件域对应一个cmmonsMultipartFile类或者MultipartFile的对象* **try** {  *//1.实现app的图片上传  //1.1获取上传文件的信息  /\*System.out.println("上传文件的名称:"+file.getOriginalFilename());  System.out.println("上传文件的大小:"+file.getSize()+"字节");  System.out.println("上传文件的类型:"+file.getContentType());\*/  //1.2上传图片* String relativePath=**"upload/logo"**;  String path=request.getSession().getServletContext().getRealPath(relativePath);  String fileName=FileUploadUtil.*uploadFile*(path,file); *//上传* **if**(fileName!=**null**){  *//2.将信息写入到数据库进行存储  //设置上传文件* appInfo.setLogopicpath(relativePath+**"/"**+fileName);  *//appInfo.setLogolocpath(path+"/"+fileName);   //设置当前用记的编号* DevUser devUser=(DevUser)session.getAttribute(**"loginInfo"**);  appInfo.setDevid(devUser.getId()); *//获取登入人的编号* **appInfoService**.addAppInfo(appInfo); *//调用业务保存  //返回页面* model.addAttribute(**"info"**,**"添加成功"**);  }  }**catch** (Exception e){  model.addAttribute(**"info"**,**"添加失败"**);  e.printStackTrace();  }  **return "developer/appinfoadd"**; } |

1. 编写前端页面

略[表单对象的名称与控制器实体的属性对应OK]