12 SpringMVC-文件上传 & 拦截器

`SpringMVC`

一：文件上传描述

> 1、Spring MVC 为文件上传提供了支持，通过MultipartResolver 实现的实现类：CommonsMultipartResovler
>
> 2、Spring MVC 上下文中默认没有装配 MultipartResovler如果想使用 Spring 的文件上传功能，需现在上下文中配置 MultipartResolver

1、配置 MultipartResolver

> 1、导入对应的jar包
> commons-fileupload-1.3.1.jar
> commons-io-2.4.jar
>
> 2、spring.xml中进行配置

```
    <!--配置上传文件的插件-->
    <bean id="multipartResolver"
class="org.springframework.web.multipart.commons.CommonsMultipartResolver
">
          <property name="defaultEncoding"  value="UTF-8"></property>
          <!-- 这一次上传总共文件大小 -->
          <property name="maxUploadSize"  value="1231234"></property>
             <!-- 上传每个文件的大小 -->
<property name="maxUploadSizePerFile" value="#{1024*1024}"></property>
<!-- 阈值(如果在这个范围,之内存在内存中之外存在临时文件中) -->
 <property name="maxInMemorySize" value="#{2*1024*1024}">
    </bean>
```

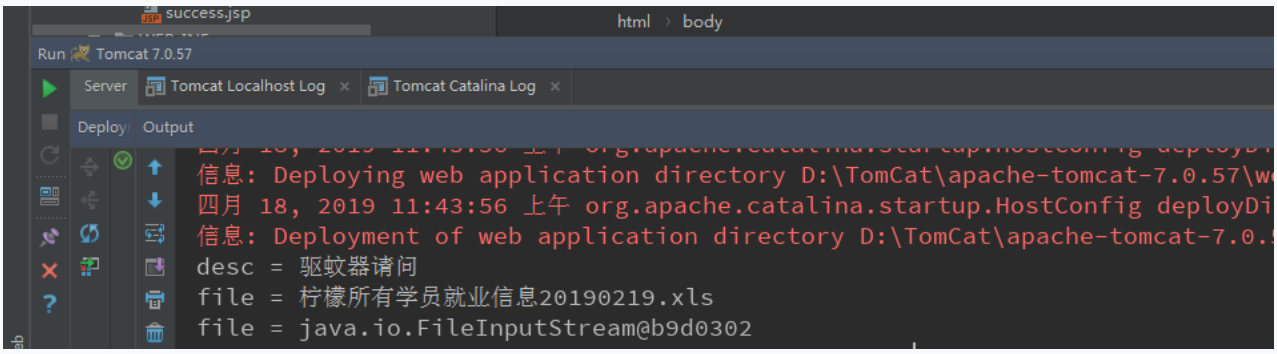
> 3、controller层请求

```
@RequestMapping("/testMultipartResolver")
    public String testMultipartResolver(@RequestParam("desc") String desc,
                                        @RequestParam("file") MultipartFile file) throws IOException {
  if (!file.isEmpty()) {
            System.out.println("desc = " + desc);
            System.out.println("file = " + file.getOriginalFilename());
            System.out.println("file = " + file.getInputStream());
        }
        return  "success";
    }
```

4、页面请求代码

```
  <FORM ACTION="testMultipartResolver" METHOD="post"  enctype="multipart/form-data">
    file:<INPUT TYPE="file" NAME="file"><br/>
    desc: <INPUT TYPE="text"  NAME="desc"><br/>
      <br/>
    <input type="submit" value="submint">
  </FORM>
```

5、执行结果



二：自定义拦截器

1、Spring MVC也可以使用拦截器对请求进行拦截处理，用户可以自定义拦截器来实现特定的功能必须实现HandlerInterceptor接口

1、preHandle()

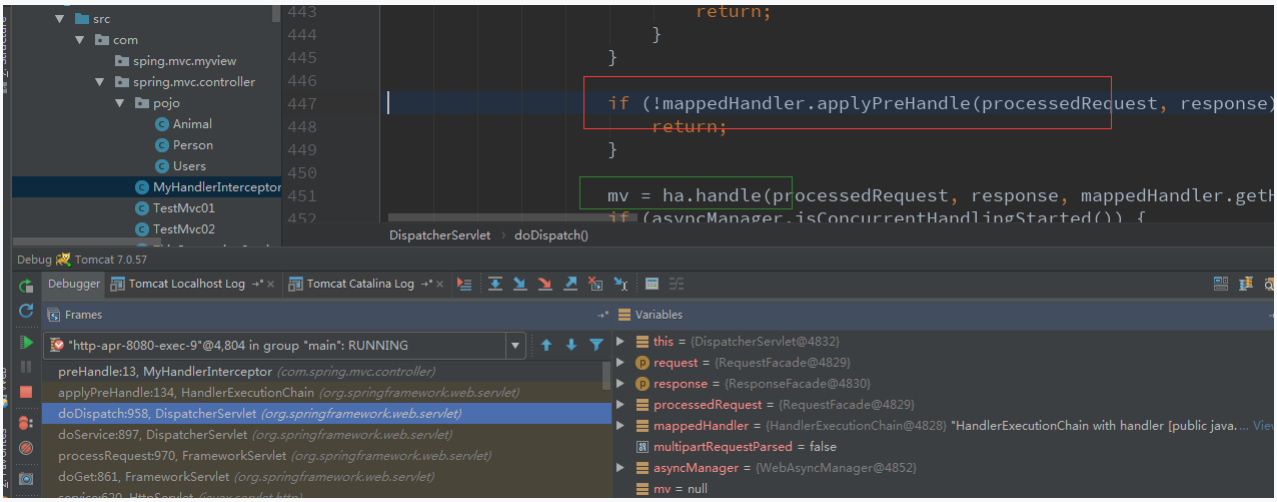
-这个方法在业务处理器处理请求之前被调用，在该方法中对用户请求 request 进行处理

-如果该拦截器对请求进行拦截处理后还要调用其他的拦截器，或者是业务处理器去进行处理，则返回true

-如果不需要再调用其他的组件去处理请求，则返回false

1、源码查询

在preHandle方法中打上断点，进行DUG测试是在ha.handle前调用applyPreHandle

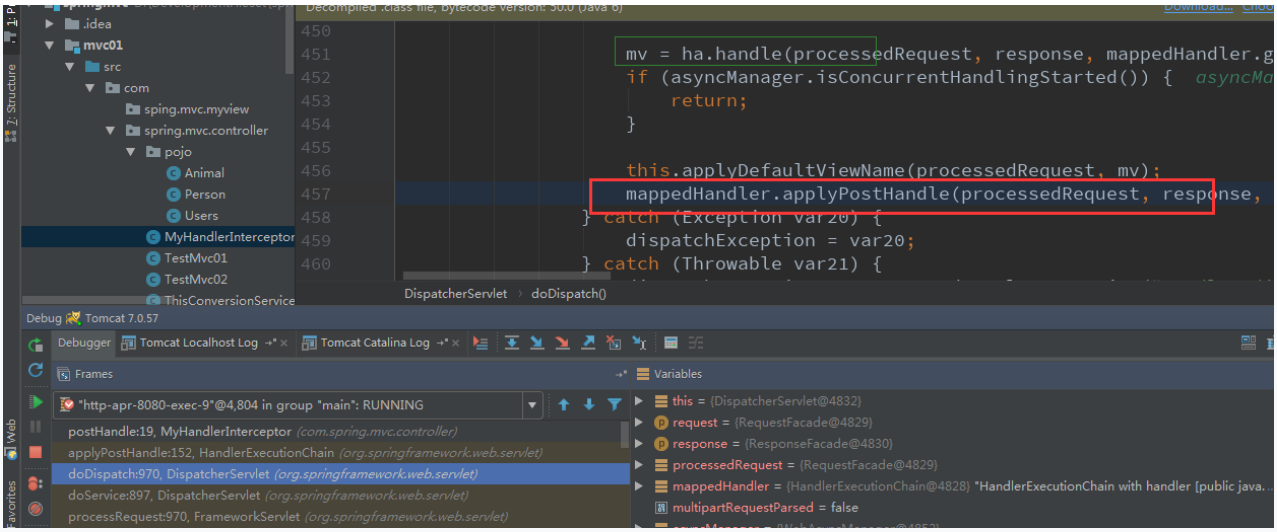


2、postHandle()

-这个方法在业务处理器处理完请求后，但是DispatcherServlet 向客户端返回响应前被调用，在该方法中对用户请求request进行处理

1、源码查询

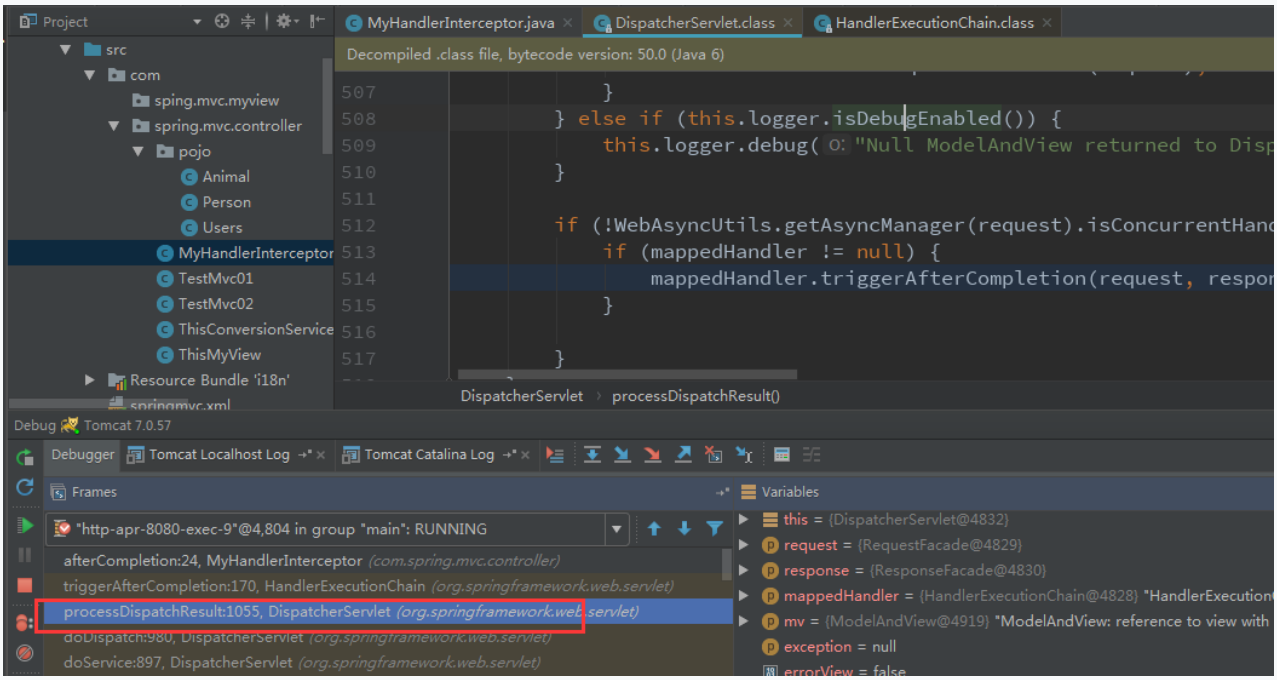
在postHandle方法中打上断点，进行DUG测试是在ha.handle后并在渲染是图之前this.processDispatchResult，调用applyPostHandle



3、afterCompletion()

-这个方法在 DispatcherServlet完全处理完请求后被调用，可以在该方法中进行一些资源清理的操作

1、源码查询



4、创建一个自定义拦截器的类

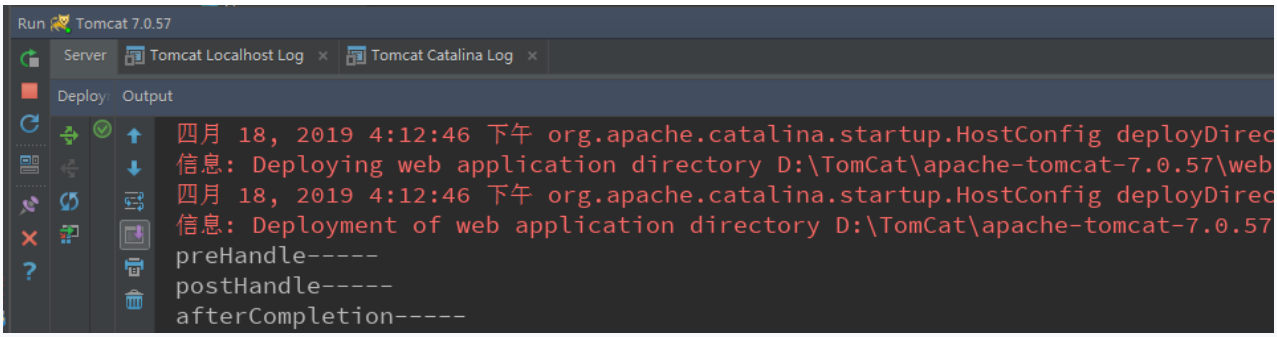
1、并且实现HandlerInterceptor接口

```
public class MyHandlerInterceptor implements HandlerInterceptor {
    @Override
    public boolean preHandle(HttpServletRequest httpServletRequest, HttpServletResponse httpServletResponse, Object o) throws Exception {
        System.out.println("preHandle-----");
        return true;
    }

    @Override
    public void postHandle(HttpServletRequest httpServletRequest, HttpServletResponse httpServletResponse, Object o, ModelAndView modelAndView) throws Exception {
        System.out.println("postHandle-----");
    }

    @Override
    public void afterCompletion(HttpServletRequest httpServletRequest, HttpServletResponse httpServletResponse, Object o, Exception e) throws Exception {
        System.out.println("afterCompletion-----");
    }
}
```

2、Spring.xml配置文件

```
    <!--配置localChanceInterceptor-->
    <mvc:interceptors>
           <!--配置自动拦截器-->
        <bean class="com.spring.mvc.controller.MyHandlerInterceptor" ></bean>
        <bean class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">
        </bean>
    </mvc:interceptors>
```

3、执行的结果



1、指定某地址生效的拦截器

1、创建一个自定义的拦截器(如上)，输出语句不一样

2、在Spring.xml文件进行配置

```
 <!--配置localChanceInterceptor-->
    <mvc:interceptors>
           <!--配置自动拦截器-->
        <bean class="com.spring.mvc.controller.MyHandlerInterceptor" ></bean>
        <bean class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">
        </bean>
         <mvc:interceptor>
             <!--
                <mvc:exclude-mapping path=""生效的的路径
              <mvc:mapping path=""  不生效的的路径
               在当前路径下,对当前的拦截器生效,其他路径不生效
             -->
             <mvc:mapping path="/testMVC/**"/>
             <bean class="com.spring.mvc.controller.NullMyHandlerInterceptor"  ></bean>
         </mvc:interceptor>
    </mvc:interceptors>
```