08 SpringMVC-重定向 & 转发 & 处理静态资源

`SpringMVC`

一:forward && redirect 描述

> 1、一般控制器方法返回字符串类型的值会被当成逻辑视图名处理
>
> 2、如果返回的字符串中带 `forward:` 或 `redirect:` 前缀
>
> 3、SpringMVC 会对他们进行特殊处理：将 `forward:` 和`redirect:` 当成指示符，其后的字符串作为 URL 来处理
>
> – redirect:success.jsp：重定向操作
>
> – forward:success.jsp：转发操作

1、controller层

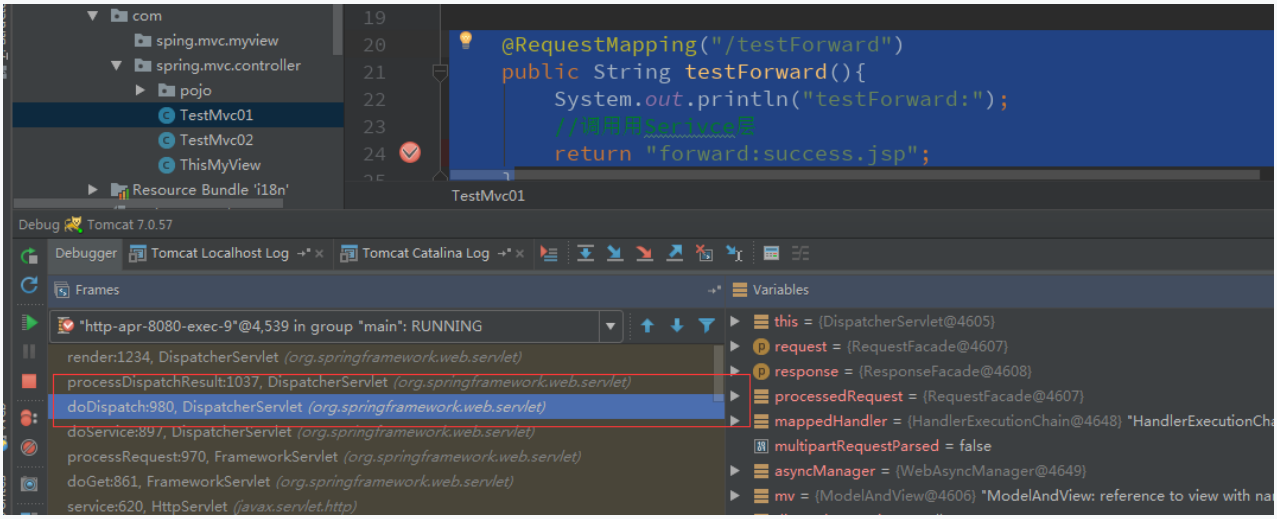
```
    @RequestMapping("/testForward")
    public String testForward(){
        System.out.println("testForward:");
        //调用用Serivce层
        return "forward:success.jsp";
    }

    @RequestMapping("/testRredirect")
    public String testRredirect(){
        System.out.println("testRredirect:");
        //调用用Serivce层
        return "redirect:success.jsp";
    }
==========================
  <HR>
  <a href="testRredirect">testRredirect</a>
  <HR>
  <HR>
  <a href="testForward">testForward</a>
  <HR>
```

2、`forward:` || `redirect:`源码解析

> 1、在请求的forward或者redirect处打上断点

```
    @RequestMapping("/testForward")
    public String testForward(){
        System.out.println("testForward:");
        //调用用Serivce层
        //打大花杠柳断点
        return "forward:success.jsp";
    }
```



> 2、在DispatcherServlet类中

```
  protected void doDispatch(HttpServletRequest request,...)
  {
  //其他代码省略.........
  /***
  打上断点，点击进入
  */
     this.processDispatchResult(processedRequest, response,
mappedHandler, mv, (Exception)dispatchException);
              } catch (Exception var22) {
  }
   //其他代码省略.........
```

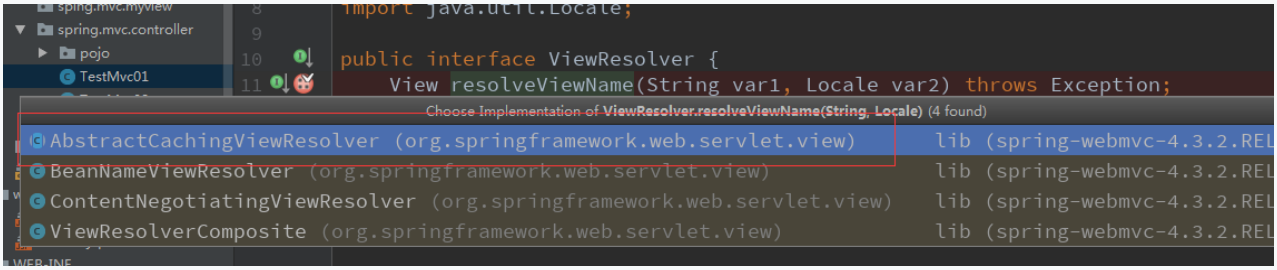
2、在processDispatchResult方法中

```
 private void processDispatchResult(HttpServletRequest request,...) {
  //其他代码省略..........
 /***
 打上断点，点击进入
 */
 if (mv != null && !mv.wasCleared()) {
             this.render(mv, request, response);
             if (errorView) {
 }
   //其他代码省略..........
```

2、在render方法中

```
protected void render(ModelAndView mv,........
  //其他代码省略..........
           if (mv.isReference()) {
   /***
 打上断点，点击进入
 */
             view = this.resolveViewName(mv.getViewName(), mv.getModelIn
ternal(), locale, request);
             if (view == null) {
  //其他代码省略..........
```

3、resolveViewName中点击实现类(Ctrl+Alt+B)



1、resolveViewName方法

```
public View resolveViewName(String viewName, Locale locale) throws
Exception {
        if (!this.isCache()) {
```

```
                /***
 打上断点，点击进入
 */
            return this.createView(viewName, locale);
        } else {
       //其他代码省略.....
```

2、createView

```
//Ctrl+Alt+B查询实现类
protected View createView(String viewName, Locale locale) throws
Exception {
        return this.loadView(viewName, locale);
    }
```

1、AbstractCachingViewResolver抽象类

1、UrlBasedViewResolver类中
2、进行判断请求的redirect还是forward

```
protected View createView(String viewName, Locale locale) throws
Exception {
        if (!this.canHandle(viewName, locale)) {
            return null;
        } else {
            String forwardUrl;
            if (viewName.startsWith("redirect:")) {
                forwardUrl = viewName.substring("redirect:".length());
                RedirectView view = new RedirectView(forwardUrl, this.i
sRedirectContextRelative(), this.isRedirectHttp10Compatible());
                view.setHosts(this.getRedirectHosts());
                return this.applyLifecycleMethods(viewName, view);
            } else if (viewName.startsWith("forward:")) {
                forwardUrl = viewName.substring("forward:".length());
                return new InternalResourceView(forwardUrl);
            } else {
                return super.createView(viewName, locale);
            }
        }
    }
```

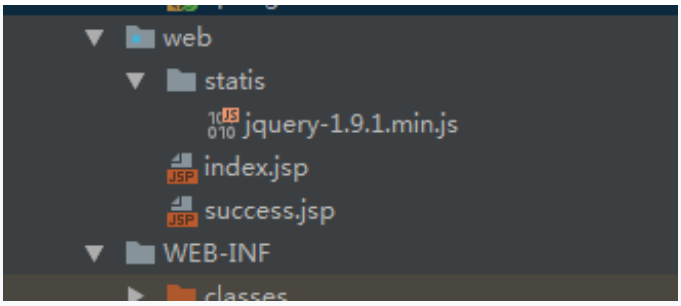
二:处理静态资源

> 1、SpringMVC 的配置文件中配置 `<mvc:default-servlet-handler/>` 的方式解决静态资源的问题
>
> 2、`<mvc:default-servlet-handler/>`将在 SpringMVC 上下文中定义一个 DefaultServletHttpRequestHandler，它会对进入 DispatcherServlet的请求进行筛查，如果发现是没有经过映射的请求
>
> 3、就将该请求交由WEB应用服务器默认的Servlet处理，如果不是静态资源的请求，才由DispatcherServlet 继续处理
>
> 4、一般 WEB 应用服务器默认的 Servlet 的名称都是default，若所使用的 WEB 服务器的默认 Servlet 名称不是 default，则需要通过 default-servlet-name 属性显式指定

```
//tomcat的xml中的进行处理
<servlet>
        <servlet-name>default</servlet-name>
        <servlet-class>org.apache.catalina.servlets.DefaultServlet</servlet-class>
        <init-param>
            <param-name>debug</param-name>
            <param-value>0</param-value>
        </init-param>
        <init-param>
            <param-name>listings</param-name>
            <param-value>false</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
```

1、放入Jquery.jsjar包

> 1、导入jar包



2、请求http://localhost:8080/statis/jquery-1.9.1.min.js

3、在SpringMVC的xml中配置静态资源

```
<!--配置静态资源-->
    <mvc:default-servlet-handler/>
```

4、请求