常用的SpringMVC异常处理方式主要有三种：

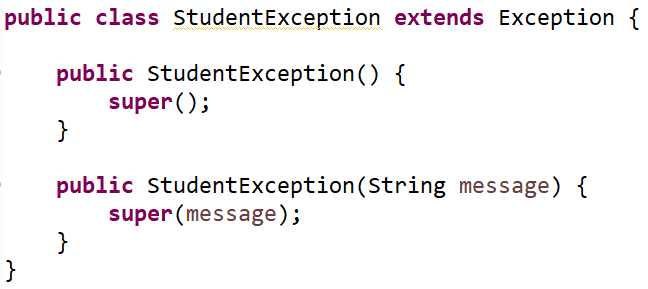
* 使用系统定义好的异常处理器SimpleMappingExceptionResolver
* 使用自定义异常处理器
* 使用异常处理注解

1. **SimpleMappingExceptionResolver 异常处理器**

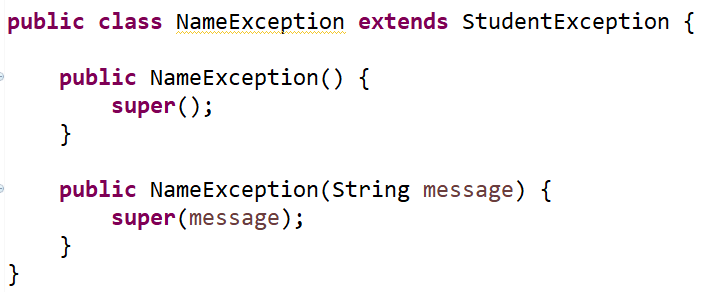
该方式只需要在SpringMVC配置文件中注册该异常处理器Bean即可。该Bean比较特殊，没有id属性，无需显式调用或被注入给其它<bean/>，当异常发生时会自动执行该类。

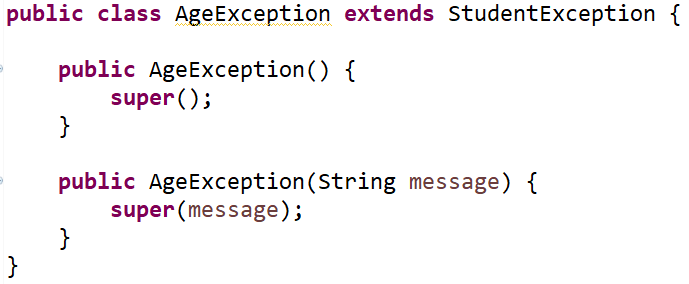
1. 自定义异常类

父类异常：

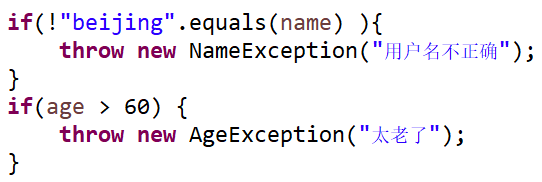


子类异常：





1. 修改Controller



1. 注册异常处理器



1. 定义异常响应页面



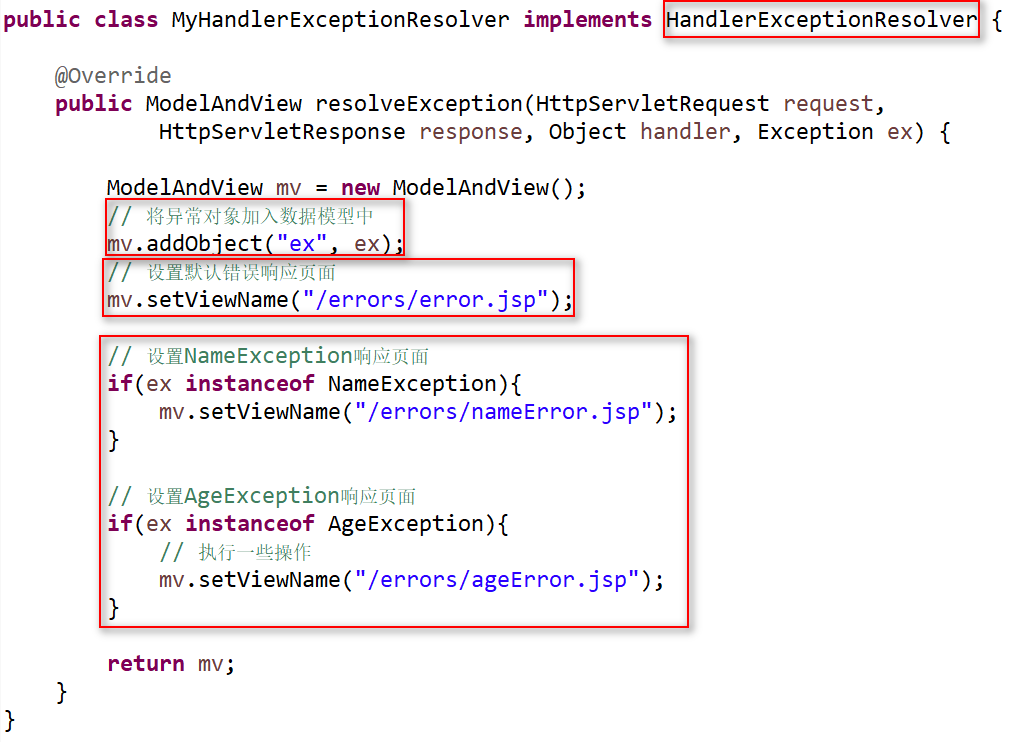
1. **自定义异常处理器**

使用SpringMVC定义好的SimpleMappingExceptionResolver异常处理器，可以实现发生指定异常后的跳转。但若要**实现在捕获到指定异常时，执行一些操作的目的**，它是完成不了的。此时，就需要自定义异常处理器。

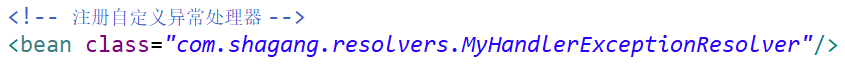
自定义异常处理器，需要实现HandlerExceptionResolver接口，并且该类需要在SpringMVC配置文件中进行注册。

1. 定义异常处理器

当一个类***实现HandlerExceptionResolver接口***后，只要有异常发生，无论什么异常，都会***自动执行接口方法resolveException()***。



1. 注册异常处理器



1. **异常处理注解**

使用***注解@ExceptionHandler可以将一个方法指定为异常处理方法***。该注解只有一个可选属性value，为一个Class<?>数组，用于指定该注解的方法所要处理的异常类，即所要匹配的异常。

而被注解的方法，其返回值可以是ModelAndView、String，或void，方法名随意，方法参数可以是Exception及其子类对象、HttpServletRequest、HttpServletResponse等。系统会自动为这些方法参数赋值。

对于异常处理注解的用法，也可以直接将异常处理方法注解于Controller之中。

不过，一般不这样使用。而是将异常处理方法专门定义在一个Controller中，让其它Controller继承该Controller即可。但是，这种用法的弊端也很明显：Java是“***单继承多实现***”的，这个Controller的继承将这唯一的一个继承机会使用了，使得若再有其它类需要继承，将无法直接实现。

1. 定义异常处理的Controller



1. 修改Controller

***让普通Controller继承自定义好的异常处理Controller***。

