Springmvc的统一全局异常处理

把Controller类中的所有处理器方法中的异常处理都统一到一个地方进行处理。

采用的是AOP的思想。

这样就把业务逻辑和异常处理代码分开，实现解耦合。

SpringMVC框架实现该框架需要两个注解：1、@ExceptionHandler 2、@ControllerAdvice

```
步骤：
1、新建maven项目依赖
2、添加各种所需要的依赖
3、创建一个exception包，其中新建一个自定义异常类MyUserException，再定义它的子类
NameException，AgeException
4、在Controller类中的方法中根据条件创建并抛出NameException和AgeException异常
5、创建一个handler包，其中创建一个普通类作为全局异常处理类
    1）、在该类的上面加入@ControllerAdvice
    2）、在类中定义方法，方法的上面添加@Exceptionhandler注解
6、创建处理异常的视图jsp页面
7、创建SpringMVC的配置文件
    1）、声明组件扫描器，扫描@Controller注解，所以要指定该注解所在包名
    2）、声明组件扫描器，扫描@ControllerAdvice所在的包名
    3）、声明注解驱动
```

1）创建项目进行测试，在idea-maven-springmvc工程中创建一个项目模块e-springmvc-exceptionhandler-05，内容使用02项目模块中的代码和配置文件，具体需要修改和添加什么如下面过程所示

2）、创建一个handler包，其中新建一个自定义异常类MyUserException，再定义它的子类NameException，AgeException

MyUserException

```
package com.studymyself.exception;

public class MyUserException extends Exception{

    public MyUserException() {
        super();
    }

    public MyUserException(String message) {
        super(message);
    }
}
```

NameException

```
package com.studymyself.exception;

//表示当用户的姓名有异常时，抛出NameException异常
public class NameException extends MyUserException{

    public NameException() {
        super();
    }

    public NameException(String message) {
        super(message);
    }
}
```

AgeException

```
package com.studymyself.exception;

//表示当输入的age属性不符合要求时，抛出AgeException异常
public class AgeException extends MyUserException{

    public AgeException() {
        super();
    }

    public AgeException(String message) {
        super(message);
    }
}
```

3）、在Controller类中的方法中根据条件创建并抛出NameException和AgeException异常

OtherController类如下

```java
package com.studymyself.controller;

import com.studymyself.exception.AgeException;
import com.studymyself.exception.MyUserException;
import com.studymyself.exception.NameException;
import com.studymyself.vo.Student;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.ModelAndView;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

@Controller
@RequestMapping(value = "/test")
public class OtherController {


    @RequestMapping(value = "/otherOne.do",method = RequestMethod.POST)
    public ModelAndView doOtherOne(String name,int age) throws MyUserException {

        ModelAndView modelAndView = new ModelAndView();

        //根据请求参数抛出异常
        if (!"niu".equals(name)){
            throw new NameException("用户名不合法");
        }
        if ((age < 0 || age>80)){
            throw new AgeException("年龄不合法");
        }
        modelAndView.addObject("myName",name);
        modelAndView.addObject("myAge",age);
        modelAndView.setViewName("other");

        return modelAndView;

    }

}
```

5）、创建一个handler包，其中创建一个普通类作为全局异常处理类

1）、在该类的上面加入@ControllerAdvice
2）、在类中定义方法，方法的上面添加@Exceptionhandler注解

```java
package com.studymyself.handler;

import com.studymyself.exception.NameException;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.servlet.ModelAndView;
```

```java
/**
 * @ControllerAdvice:
 *  该注解放在类上面，顾名思义，这个类就是一个控制器增强
 *  因为之前我们使用AOP技术给业务方法进行功能增强时，用的就是Advice通知注解
 *  Advice可以看做增强的意思，这里就是给控制器类进行功能增强，新功能就是
 *  给所有控制器的所有方法增加异常处理
 *  位置：定义类的上面
 *  特点：需要让框架知道这个注解所在的包名，在SpringMVC配置文件中声明组件扫描器指定包名
 *
 */
@ControllerAdvice
public class GlobalExceptionHandler {

    //在该类中定义方法，处理发生的异常
    /**
     * 处理异常的方法和定义控制器方法一样，方法中可以有多个参数，返回值
     * 可以是ModelAndView、String、void、对象类型
     *
     *  当参数是异常类型时，可以是Exception对象，
     *  方法之中可以通过该异常对象获取抛出的异常信息
     *
     *  注解@ExceptionHandler(XxxException.class)
     *    其中有个value属性，是Class类型的，填写某个异常，表示
     *    当我们的Controller类中的某个方法发生所填写的异常时
     *    就由当前方法处理
     *    value的值我们可以填写控制器方法中具体的某个异常类型，
     *    创建多个方法分别处理控制器方法发生的异常，但是我们一般
     *    都是定义一个方法，value的值填Exception.class，这样抛出
     *    的所有异常都交给这个方法处理
     */
    @ExceptionHandler(value = NameException.class)
    public ModelAndView doOtherOneNameException(Exception e){
        //处理NameException异常
        /**
         * 在该方法中异常发生的处理逻辑：
         * 1、需要把异常记入下来，记入到数据库，日志文件
         *  记入发生的时间、哪个方法、异常的错误内容到日志
         * 2、发送通知，把异常信息通过邮件，短信，微信等方式发送给相关人员
         * 3、给用户提供友好提示
         * 1和2实现比较复杂，这里只是实现3
         */
        ModelAndView modelAndView = new ModelAndView();
        modelAndView.addObject("msg","姓名必须是niu，其他输入无法访问");
        modelAndView.addObject("e",e);
        modelAndView.setViewName("nameError");
        return modelAndView;
    }

    @ExceptionHandler(value = NameException.class)
    public ModelAndView doOtherOneAgeException(Exception e){
        //处理AgeException异常
        ModelAndView modelAndView = new ModelAndView();
        modelAndView.addObject("msg","年龄必须大于0小于130");
        modelAndView.addObject("e",e);
        modelAndView.setViewName("ageError");
        return modelAndView;
    }
```

```
    @ExceptionHandler(value = Exception.class)
    public ModelAndView doOtherOneException(Exception e){
        //处理非NameException和AgeException的其他异常
        //我们可以在整型的age属性填一个abc，这样的异常交给该方法进行处理
        ModelAndView modelAndView = new ModelAndView();
        modelAndView.addObject("msg","其他非法输入");
        modelAndView.addObject("e",e);
        modelAndView.setViewName("defaultError");
        return modelAndView;
    }

    /**
     * 处理流程：当我们的程序出现异常后，要么是向上抛，要么是
     *   而框架的异常处理就是将try/catch中catch语句块中内容和业务代码分开来，
     *   方法出现异常，框架将异常根据@Exceptionhandler中的值进行比较，符合哪一个
     *   就调用哪一个方法进行处理，这相当于是一个切面，只是增加在控制器Controller
     *   类中的，相当于运行在catch语句块中的内容
     */


}
```

6）、创建处理异常的视图jsp页面，

nameError.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>NameException发生页面</title>
</head>
<body>
    nameError.jsp<br/>
    提示信息：${msg}<br/>
    异常信息：${e.message}<br/>
</body>
</html>
```

ageError.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>AgeException发生页面</title>
</head>
<body>
ageError.jsp<br/>
提示信息：${msg}<br/>
异常信息：${e.message}<br/>
</body>
</html>
```

defaultError.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>其他发生页面</title>
</head>
<body>
defaultError.jsp<br/>
提示信息：${msg}<br/>
异常信息：${e.message}<br/>
</body>
</html>
```

index.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>Welcome Page</title>
</head>
<body>
    <P>提交参数给Controller，接收请求参数</P>
    <form action="test/otherOne.do" method="post">
        姓名:<input type="text" name="name"><br>
        年龄:<input type="text" name="age"><br>
        <input type="submit" value="提交参数">
    </form>

</body>
</html>
```

7）、创建SpringMVC的配置文件

1）、声明组件扫描器，扫描@Controller注解，所以要指定该注解所在包名
2）、声明组件扫描器，扫描@ControllerAdvice所在的包名
3）、声明注解驱动

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:mvc="http://www.springframework.org/schema/mvc"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context
https://www.springframework.org/schema/context/spring-context.xsd
http://www.springframework.org/schema/mvc
https://www.springframework.org/schema/mvc/spring-mvc.xsd">
    <!--声明组件扫描器-->
    <context:component-scan base-package="com.studymyself.controller"/>

    <!--声明配置视图解析器,框架提供的一个类-->
```

```
    <bean
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <!--前缀属性：视图文件的路径，前后要加/，前一个/表示项目根路径：
http://localhost:8080/a_springmvc_web_01-->
        <property name="prefix" value="/WEB-INF/view/"/>
        <!--后缀属性：视图文件的类型-->
        <property name="suffix" value=".jsp"/>
    </bean>

    <!--处理异常的两个声明组件-->
    <context:component-scan base-package="com.studymyself.handler"/>
    <mvc:annotation-driven/>
</beans>
```

8）、发布项目，启动服务器进行测试

```
java.lang.IllegalStateException: Ambiguous @ExceptionHandler method mapped for
[class com.studymyself.exception.NameException]: {public
org.springframework.web.servlet.ModelAndView
com.studymyself.handler.GlobalExceptionHandler.doOtherOneNameException(java.lang
.Exception), public org.springframework.web.servlet.ModelAndView
com.studymyself.handler.GlobalExceptionHandler.doOtherOneAgeException(java.lang.
Exception)}
```

测试时出现上面的异常：

表示我在全局异常处理类中定义的两个异常处理方法的`@ExceptionHandler`的`value`使用了相同的异常`Class`

虽然以后只会定义一个处理方法，但是还是记住为好

输入：

name：niu

age：10

输入：

name：ss

age：10

输入：

name：niu

age：150

输入：

name：niu

age：abc