SpringBoot统一异常处理

01、分析

- 统一跳转：/error 这是一种全局的机制
- 配置类：补充状态进行跳转 -- 局部的机制
- 自定义页面的方式，方便我们可以把error.html随心所欲的进行存放

问题：

- 其实给开发增大的工作量，
- 不明确具体异常信息，如果要追求细粒度的控制。
- 内部定义的状态 HttpStatus.状态枚举，是一个大方向的错误指定
    - 比如：INTERNAL_SERVER_ERROR 它是服务器只要任何方法执行报任何异常Exception 都会是500。这就会给开发者带来困扰，给用户一个错误就够了。对开发者来说就不够细粒度，因为未来程序的开发大部分是一种前后端分离的开发方式，如果不给接口调用者，具体的错误信息提示的话，可能会造成很多的沟通成本，开发的时间成本。
    - 在开发中越具体的错误捕获对于开发者来说排除错误是非常有利的。

举例：

比如开发用户注册的接口：

比如：用户名不为空，密码格式不对

- 不友好的统一返回：{status:500,msg:"未知错误”}
- 友好的统一返回：`{status:501,msg:"用户名不为空"}`
  `{status:502,msg:"密码格式不对"}`

02、异常规则

- 具体异常优先级要高于大异常。
- 在try/catch的具体异常一定写在大异常上面

```
 try {
            Connection con = null;
            PreparedStatement preparedStatement
= con.prepareStatement("");
        }catch (SQLException sqlex){
            sqlex.printStackTrace();
        } catch(Exception exx){

        }
```

03、springboot如何做到细粒度自定异常返回呢？

01、用@ControllerAdvice

```
package com.kuangstudy.web.error.config2;

import
org.springframework.web.bind.annotation.ControlleRAdvice;
import
org.springframework.web.bind.annotation.ExceptionHandler;

import javax.servlet.http.HttpServletRequest;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;

/**
 * Description:
 * Author: yykk Administrator
 * Version: 1.0
 * Create Date Time: 2021/12/15 22:37.
 * Update Date Time:
 *
 * @see
```

```
 */
@ControllerAdvice
public class GlobalExceptionControllerHandler {

    /**
     * 拦截所有程序异常
     * @param request
     * @param ex
     * @return
     */
    @ExceptionHandler(value=Exception.class)
    public String errorHandler(HttpServletRequest request,Exception ex){
        return "err2/noError";
    }

    /**
     * SQLException异常
     * @param request
     * @param ex
     * @return
     */
    @ExceptionHandler(value=SQLException.class)
    public String errorHandlerSQL(HttpServletRequest request,Exception ex){
        return "err2/sqlError";
    }

    /**
     * MyException异常
```

```
47          * @param request
48          * @param ex
49          * @return
50          */
51         @ExceptionHandler(value=MyException.class)
52         public String
   errorHandlerMy(HttpServletRequest
   request,Exception ex){
53             return "err2/myError";
54         }
55
56 }
```

@ControllerAdvice 和 @RestControllerAdvice底层原理是：AOP主要用于开启全局异常处理一种机制，对后面的统一返回，统一异常处理，统一参数注入都会用这个@ControllerAdvice。

02、@RestControllerAdvice

有了@ControllerAdvice 为什么还出现@RestControllerAdvice，其实和@Controller和@RestController一个道理。因为在程序开发中，不仅仅只有页面返回处理。如果单体项目，有freemarker 和 thymeleaf的话其实使用@ControllerAdvice做统一异常处理能够满足错误处理机制。

如果在有freemarker 和 thymeleaf的使用@RestControllerAdvice 会怎么样呢？

```java
package com.kuangstudy.web.error.config2;

import org.springframework.web.bind.annotation.ControllerAdvice;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestControllerAdvice;

import javax.servlet.http.HttpServletRequest;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;

/**
 * Description:
 * Author: yykk Administrator
 * Version: 1.0
 * Create Date Time: 2021/12/15 22:37.
 * Update Date Time:
 *
 * @see
 */
//@ControllerAdvice
@RestControllerAdvice
```

```
24 public class GlobalExceptionControllerHandler {
25
26     /**
27      * 拦截所有程序异常
28      * @param request
29      * @param ex
30      * @return
31      */
32     @ExceptionHandler(value=Exception.class)
33     @ResponseBody
34     public String errorHandler(HttpServletRequest
   request,Exception ex){
35         return "err2/noError";
36     }
37
38     /**
39      * SQLException异常
40      * @param request
41      * @param ex
42      * @return
43      */
44     @ExceptionHandler(value=SQLException.class)
45     @ResponseBody
46     public String
   errorHandlerSQL(HttpServletRequest
   request,Exception ex){
47         return "err2/sqlError";
48     }
49
50     /**
51      * MyException异常
```

```
52          * @param request
53          * @param ex
54          * @return
55          */
56         @ExceptionHandler(value=MyException.class)
57         @ResponseBody
58         public String
   errorHandlerMy(HttpServletRequest
   request,Exception ex){
59             return "err2/myError";
60         }
61
62     }
63
```

@ControllerAdvice 和 @RestControllerAdvice的区别：

- 通过上面的分析，得出结论@ControllerAdvice根据你的返回值找页面。@RestControllerAdvice直接把方法的内容输出
- 其实和@Controller和@RestController是一个含义。所以我们把统一异常处理的类GlobalExceptionControllerHandler当做Controller去对待就对了。它只不过是一个特殊的Controller 就出现异常以后就交给这个特殊GlobalExceptionControllerHandler来处理。

04、开发中我到底使用那种会更好呢？

- 如果是前后端分离的方式，只能使用@RestControllerAdvice。为什么：因为前后端分离压根就没有freemarker或者

thymeleaf，也就说没有页面，也没有静态资源。

- 如果是单体项目存在freemarker或者 thymeleaf，你想跳页面给用户呈现你就使用：@ControllerAdvice 如果你想返回状态和具体信息：你就使用@RestControllerAdvice

01、@ControllerAdvice + 页面跳转方式呈现具体细粒度错误信息在页面

```
1  package com.kuangstudy.web.error.config2;
2
3  import org.springframework.http.HttpStatus;
4  import
   org.springframework.web.bind.annotation.Controlle
   rAdvice;
5  import
   org.springframework.web.bind.annotation.Exception
   Handler;
6  import
   org.springframework.web.bind.annotation.ResponseB
   ody;
7  import
   org.springframework.web.bind.annotation.RestContr
   ollerAdvice;
8  import
   org.springframework.web.servlet.ModelAndView;
9
10 import javax.servlet.http.HttpServletRequest;
11 import java.sql.Connection;
12 import java.sql.PreparedStatement;
```

```
13 import java.sql.SQLException;
14
15 /**
16  * Description:
17  * Author: yykk Administrator
18  * Version: 1.0
19  * Create Date Time: 2021/12/15 22:37.
20  * Update Date Time:
21  *
22  * @see
23  */
24 @ControllerAdvice
25 //@RestControllerAdvice
26 public class GlobalExceptionControllerHandler {
27
28     /**
29      * 拦截所有程序异常
30      * @param request
31      * @param ex
32      * @return
33      */
34     @ExceptionHandler(value=Exception.class)
35     public ModelAndView
   errorHandler(HttpServletRequest request,
   Exception ex ){
36         ModelAndView modelAndView = new
   ModelAndView();
37         modelAndView.setViewName("err2/noError");
38         modelAndView.addObject("status",
   HttpStatus.INTERNAL_SERVER_ERROR);
```

```
39
    modelAndView.addObject("msg",ex.getMessage());
40
    modelAndView.addObject("url",request.getRequestURL().toString());
41          return modelAndView;
42      }
43
44      /**
45       * SQLException异常
46       * @param request
47       * @param ex
48       * @return
49       */
50      @ExceptionHandler(value=SQLException.class)
51      public String errorHandlerSQL(HttpServletRequest request,Exception ex){
52          return "err2/sqlError";
53      }
54
55      /**
56       * MyException异常
57       * @param request
58       * @param ex
59       * @return
60       */
61      @ExceptionHandler(value=MyException.class)
62      @ResponseBody
```

```
63      public String
    errorHandlerMy(HttpServletRequest
    request,Exception ex){
64          return "err2/myError";
65      }
66
67  }
68
```

02、@RestControllerAdvice 返回json错误信息给用户和开发者

```
1  package com.kuangstudy.web.error.config2;
2
3  import org.springframework.http.HttpStatus;
4  import
    org.springframework.web.bind.annotation.Controlle
    rAdvice;
5  import
    org.springframework.web.bind.annotation.Exception
    Handler;
6  import
    org.springframework.web.bind.annotation.ResponseB
    ody;
7  import
    org.springframework.web.bind.annotation.RestContr
    ollerAdvice;
8  import
    org.springframework.web.servlet.ModelAndView;
9
```

```
10 import javax.servlet.http.HttpServletRequest;
11 import java.sql.Connection;
12 import java.sql.PreparedStatement;
13 import java.sql.SQLException;
14 import java.util.HashMap;
15 import java.util.Map;
16
17 /**
18  * Description:
19  * Author: yykk Administrator
20  * Version: 1.0
21  * Create Date Time: 2021/12/15 22:37.
22  * Update Date Time:
23  *
24  * @see
25  */
26 @RestControllerAdvice
27 public class GlobalExceptionControllerHandler {
28
29     /**
30      * 拦截所有程序异常
31      * @param request
32      * @param ex
33      * @return
34      */
35     @ExceptionHandler(value=Exception.class)
36     public Map<String,Object>
   errorHandler(HttpServletRequest request,
   Exception ex ){
37         Map<String,Object> map = new HashMap<>();
```

```
38
 map.put("status",HttpStatus.INTERNAL_SERVER_ERRO
R);
39          map.put("msg",ex.getMessage());
40
 map.put("url",request.getRequestURL().toString()
);
41          return map;
42      }
43
44      /**
45       * SQLException异常
46       * @param request
47       * @param ex
48       * @return
49       */
50      @ExceptionHandler(value=SQLException.class)
51      public Map<String,Object>
errorHandlerSQL(HttpServletRequest
request,Exception ex){
52          Map<String,Object> map = new HashMap<>();
53          map.put("status",601);
54          map.put("msg",ex.getMessage());
55
 map.put("url",request.getRequestURL().toString()
);
56          return map;
57      }
58
59      /**
60       * MyException异常
```

```
61          * @param request
62          * @param ex
63          * @return
64          */
65         @ExceptionHandler(value=MyException.class)
66         @ResponseBody
67         public Map<String,Object>
   errorHandlerMy(HttpServletRequest
   request,Exception ex){
68             Map<String,Object> map = new HashMap<>();
69             map.put("status",602);
70             map.put("msg",ex.getMessage());
71
    map.put("url",request.getRequestURL().toString()
   );
72             return map;
73         }
74
75 }
76
```

- 使用@RestControllerAdvice，它的返回值建议在是String和ModelAndView，如果你返回ModelAndView就会指定setViewName页面的源码通过fm和th渲染以后返回。如果String直接返回字符串，对于用户和开发者来说，没有意义。特别用户者看不懂，对开发者信息不方便解析。

05、统一返回为什么是R类，而不是Map或者Object

原因如:

- Map不具备面向对象的特征
- Object 不明确类型。
- 建议自己去定义一个统一返回来处理统一异常。

命名方式:

- R
- ResponseResult
- ApiResponse
- Result

无论用那种，都是一种面向封装的思想。

06、总结

理解：全局异常处理就很像另外一个controller,由异常去触发