**《Spring Boot基础教程》**

**第9节**

**web应用开发-错误处理**

1. **错误的处理**

**方法一：Spring Boot 将所有的错误默认映射到/error，** **实现ErrorController**

@Controller

@RequestMapping(value = "error")

public class BaseErrorController implements ErrorController {

private static final Logger logger = LoggerFactory.getLogger(BaseErrorController.class);

@Override

public String getErrorPath() {

logger.info("出错啦！进入自定义错误控制器");

return "error/error";

}

@RequestMapping

public String error() {

return getErrorPath();

}

}

**方法二：添加自定义的错误页面**

2.1 html静态页面：在resources/public/error/ 下定义

如添加404页面： resources/public/error/404.html页面，中文注意页面编码

2.2 模板引擎页面：在templates/error/下定义

如添加5xx页面： templates/error/5xx.ftl

注：templates/error/ 这个的优先级比较 resources/public/error/高

**方法三：使用注解@ControllerAdvice**

/\*\*

\* 统一异常处理

\*

\* @param exception

\* exception

\* @return

\*/

@ExceptionHandler({ RuntimeException.class })

@ResponseStatus(HttpStatus.OK)

public ModelAndView processException(RuntimeException exception) {

logger.info("自定义异常处理-RuntimeException");

ModelAndView m = new ModelAndView();

m.addObject("roncooException", exception.getMessage());

m.setViewName("error/500");

return m;

}

/\*\*

\* 统一异常处理

\*

\* @param exception

\* exception

\* @return

\*/

@ExceptionHandler({ Exception.class })

@ResponseStatus(HttpStatus.OK)

public ModelAndView processException(Exception exception) {

logger.info("自定义异常处理-Exception");

ModelAndView m = new ModelAndView();

m.addObject("roncooException", exception.getMessage());

m.setViewName("error/500");

return m;

}

**更多课程信息，请关注 龙果学院 官方网站**[**http://www.roncoo.com/**](http://www.roncoo.com/)

**或关注 龙果 微信公众号 RonCoo\_com**

****