

# jBoot validator 完美实现+统一封装错误提示

| 19:23:21 catoop 阅读数 562 更多

主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。  
[/shanhhy.blog.csdn.net/article/details/95366348](#)

else 参数判断!!!  
式  
数方式

自动使用统一异常返回错误，例如：

```
code": "-3",  
message": "参数{age}最小18"
```

，主要包含如下几个Java类，拿过去拷贝到项目中就能用：

ontroller  
o  
alidatorService  
e  
onseBodyAdvice  
IGNORE

```
: javax.validation.constraints.Email;  
: javax.validation.constraints.NotNull;  
: javax.validation.constraints.Size;  
  
: org.springframework.beans.factory.annotation.Autowired;  
: org.springframework.validation.annotation.Validated;  
: org.springframework.web.bind.annotation.GetMapping;  
: org.springframework.web.bind.annotation.RequestMapping;  
: org.springframework.web.bind.annotation.RestController;  
  
: link.nuggets.user.module.example.model.ValidatorVo;  
: link.nuggets.user.module.example.service.ExampleValidatorService;
```

.idator示例

ithor 单红宇  
ite 2019年7月8日

```
lated // 注意! 1) 如果想在参数中使用 @NotNull 这种注解校验，就必须在类上添加 @Validated; 2) 如果方法中的参数是对象类型，则必须要在参数对象前  
ontroller  
stMapping("/example/valid")  
: class ValidatorController {
```

```
utowired  
ivate ExampleValidatorService exampleValidatorService;
```

```

/*
 * 直接参数校验
 * 要特别提醒的是，验证框架里面大部分都不需要我们显示设置message，每个注解框架都给了一个默认提示语，大多数提示还都比较友好
 *
 * @param email
 * @return
 */
@GetMapping("/test1")
public String test1(@NotNull(message = "不能为空") @Size(max = 32, min = 6, message = "长度需要在6-32之间") @Email String email) {
    return "OK";
}

/*
 * 实体类校验
 *
 * @param validatorVo
 * @return
 */
@GetMapping("/test2")
public String test2(@Validated ValidatorVo validatorVo) {
    return "Validator OK";
}

/*
 * 内部Service校验
 *
 * @return
 */
@GetMapping("/test3")
public String test3() {
    return exampleValidatorService.show("16");
}

import javax.validation.constraints.Max;
import javax.validation.constraints.Min;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;

import lombok.Data;
```

示例V0

作者 单红宇

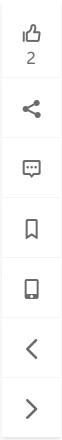
时间 2019年7月10日

```

class ValidatorVo {

    @NotNull(message = "不能为空")
    @Size(max = 16, min = 6, message = "字符串长度需要在6-16之间")
    private String name;

    @Max(value = 100, message = "最大100")
    @Min(value = 18, message = "最小18")
    private String age;
```



```
: javax.validation.constraints.Min;
: javax.validation.constraints.NotNull;

: org.slf4j.Logger;
: org.slf4j.LoggerFactory;
: org.springframework.stereotype.Service;
: org.springframework.validation.annotation.Validated;
```

]]

uthor 单红宇  
ite 2019年7月10日

```
lated
.ce
: class ExampleValidatorService {

ivate static final Logger logger = LoggerFactory.getLogger(ExampleValidatorService.class);

blic String show(@NotNull(message = "不能为空") @Min(value = 18, message = "最小18") String age) {
    logger.info("age = {}", age);
    return age;
}
```

-返回值

uthor 单红宇  
ite 2019年6月26日

```
: enum ResultCode {

/* 操作成功 */
UCCESS("1", "成功"),

/* 操作失败 */
IL("0", "失败"),

/* 操作失败 */
ILL("-1", "数据不存在"),


/* 系统发生异常 */
CEPTION("-2", "系统异常"),


/* 没有权限 */
RIBIDDEN("9403", "没有权限"),


/* 参数错误 */
RAM_INVALID("-3", "参数错误");


ivate String code;
ivate String msg;


ivate ResultCode(String code, String msg) {
    this.code = code;
    this.msg = msg;
}
```


2






















```

public String code() {
    return code;

public String msg() {
    return msg;

: java.util.Set;

: javax.servlet.http.HttpServletRequest;
: javax.validation.ConstraintViolation;
: javax.validation.ConstraintViolationException;

: org.hibernate.validator.internal.engine.path.PathImpl;
: org.slf4j.Logger;
: org.slf4j.LoggerFactory;
: org.springframework.boot.autoconfigure.web.servlet.error.BasicErrorController;
: org.springframework.core.MethodParameter;
: org.springframework.http.HttpStatus;
: org.springframework.http.MediaType;
: org.springframework.http.ResponseEntity;
: org.springframework.http.converter.HttpMessageConverter;
: org.springframework.http.converter.json.MappingJackson2HttpMessageConverter;
: org.springframework.http.server.ServerHttpRequest;
: org.springframework.http.server.ServerHttpResponse;
: org.springframework.http.server.ServletServerHttpRequest;
: org.springframework.validation.BindException;
: org.springframework.validation.FieldError;
: org.springframework.web.bind.annotation.ControllerAdvice;
: org.springframework.web.bind.annotation.ExceptionHandler;
: org.springframework.web.servlet.mvc.method.annotation.ResponseBodyAdvice;

```

controller 统一返回包装

author 单红宇

date 2019年6月26日

```

@ControllerAdvice
: class ResultResponseBodyAdvice implements ResponseBodyAdvice<Object> {

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

@Override
public Object beforeBodyWrite(Object body, MethodParameter returnType, MediaType selectedContentType,
    Class<?> extends HttpMessageConverter<?>> selectedConverterType, ServerHttpRequest request,
    ServerHttpResponse response) {
    // 此处获取到request 为特殊需要的时候处理使用
    HttpServletRequest req = ((ServletServerHttpRequest) serverHttpRequest).getServletRequest();
    // 下面处理统一返回结果 (统一code、msg、sign 加密等)
    if (selectedConverterType == MappingJackson2HttpMessageConverter.class
        && (selectedContentType.equals(MediaType.APPLICATION_JSON)
            || selectedContentType.equals(MediaType.APPLICATION_JSON_UTF8))) {
        if (body == null) {
            return ResultV0.NULL;
        } else if (body instanceof ResultV0) {
            return body;
        } else {
            // 异常

```



2



```

        if (returnType.getExecutable().getDeclaringClass().isAssignableFrom(BasicErrorController.class)) {
            ResultVO vo = new ResultVO(ResultCode.EXCEPTION);
            HttpServletRequest req = ((ServletServerHttpRequest) request).getServletRequest();
            if (req.getRequestURL().toString().contains("localhost")
                || req.getRequestURL().toString().contains("127.0.0.1"))
                vo.setData(body);
            return vo;
        } else {
            return new ResultVO(body);
        }
    }
}
return body;

```

```

@Override
public boolean supports(MethodParameter returnType, Class<? extends HttpMessageConverter<?>> converterType) {
    if (returnType.hasMethodAnnotation(ResultVO_IGNORE.class))
        return false;
    return true;
}

```

```

/*
 * Validator 参数校验异常处理
 */
@param ex
@return
/

ExceptionHandler(value = BindException.class)
public ResponseEntity<Object> handleMethodVoArgumentNotValidException(BindException ex) {
    FieldError err = ex.getFieldError();
    // err.getField() 读取参数字段
    // err.getDefaultMessage() 读取验证注解中的message值
    String message = "参数{" .concat(err.getField()).concat("}").concat(err.getDefaultMessage());
    logger.info("{} -> {}", err.getObjectName(), message);
    return new ResponseEntity<Object>(new ResultVO(ResultCode.PARAM_INVALID, message), HttpStatus.OK);
}

/*
 * Validator 参数校验异常处理
 */
@param ex
@return
/

ExceptionHandler(value = ConstraintViolationException.class)
public ResponseEntity<Object> handleMethodArgumentNotValidException(ConstraintViolationException ex) {
    Set<ConstraintViolation<?>> constraintViolations = ex.getConstraintViolations();
    for (ConstraintViolation<?> constraintViolation : constraintViolations) {
        PathImpl pathImpl = (PathImpl) constraintViolation.getPropertyPath();
        // 读取参数字段, constraintViolation.getMessage() 读取验证注解中的message值
        String paramName = pathImpl.getLeafNode().getName();
        String message = "参数{" .concat(paramName).concat("}").concat(constraintViolation.getMessage());
        logger.info("{} -> {} -> {}", constraintViolation.getRootBeanClass().getName(), pathImpl.toString(), message);
        return new ResponseEntity<Object>(new ResultVO(ResultCode.PARAM_INVALID, message), HttpStatus.OK);
    }
}
return new ResponseEntity<Object>(new ResultVO(ResultCode.PARAM_INVALID, ex.getMessage()), HttpStatus.OK);
}

```

```

: java.lang.annotation.Documented;
: java.lang.annotation.ElementType;
: java.lang.annotation.Retention;
: java.lang.annotation.RetentionPolicy;

```



2



```
: java.lang.annotation.Target;
```

Controller默认进行ResultVO包装，对于特殊不需要的，使用该注解可以忽略包装

作者 单红宇  
时间 2019年6月26日

```
@Target(ElementType.METHOD)
@Retention(RetentionPolicy.RUNTIME)
@Inherited
@interface ResultVO_IGNORE {
```

```
: org.slf4j.Logger;
: org.slf4j.LoggerFactory;
```

```
: com.google.gson.Gson;
```

Controller 统一定义返回类

作者 单红宇  
时间 2019年6月26日

```
: class ResultVO {

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

public static final ResultVO SUCCESS = new ResultVO(ResultCode.SUCCESS);
public static final ResultVO FAIL = new ResultVO(ResultCode.FAIL);
public static final ResultVO FORBIDDEN = new ResultVO(ResultCode.FORBIDDEN);
public static final ResultVO NULL = new ResultVO(ResultCode.NULL);
public static final ResultVO EXCEPTION = new ResultVO(ResultCode.EXCEPTION);
public static final ResultVO PARAM_INVALID = new ResultVO(ResultCode.PARAM_INVALID);

/**
 * 返回代码
 */
private String code;

/**
 * 返回信息
 */
private String message;

/**
 * 返回数据
 */
private Object data;

public Object getData() {
    return data;
}

public void setData(Object data) {
    this.data = data;
}

/**
 * 默认构造，返回操作正确的返回代码和信息
```



2



```

    /
    ublic ResultV0() {
        this.setCode(ResultCode.SUCCESS.code());
        this.setMessage(ResultCode.SUCCESS.msg());
    }

    /**
     * 构造一个返回特定代码的ResultV0对象
     */
    @param code
    /
    ublic ResultV0(ResultCode code) {
        this.setCode(code.code());
        this.setMessage(code.msg());
    }

    ublic ResultV0(String code, String message) {
        super();
        this.setCode(code);
        this.setMessage(message);
    }

    /**
     * 默认值返回，默认返回正确的code和message
     */
    @param data
    /
    ublic ResultV0(Object data) {
        ResultCode rc = data == null ? ResultCode.NULL : ResultCode.SUCCESS;
        this.setCode(rc.code());
        this.setMessage(rc.msg());
        this.setData(data);
    }

    /**
     * 构造返回代码，以及自定义的错误信息
     */
    @param code
    @param message
    /
    ublic ResultV0(ResultCode code, String message) {
        this.setCode(code.code());
        this.setMessage(message);
    }

    /**
     * 构造自定义的code, message, 以及data
     */
    @param code
    @param message
    @param data
    /
    ublic ResultV0(ResultCode code, String message, Object data) {
        this.setCode(code.code());
        this.setMessage(message);
        this.setData(data);
    }

    ublic String getMessage() {
        return message;
    }

    ublic void setMessage(String message) {
        this.message = message;
    }

```



2




```
public String getCode() {  
    // request请求响应的时候，一定会走到这里，判断如果code不是成功状态，就输出日志  
    if (!ResultCode.SUCCESS.code().equals(code))  
        logger.info("ResultV0={}", new Gson().toJson(this));  
    return code;  
  
public void setCode(String code) {  
    this.code = code;  
}
```


的提示，springboot 的web项目正常都是已经包含如下依赖了，该依赖里已经包含了validation-api


```
<dependency>  
    <groupId>org.springframework.boot</groupId>  
    <artifactId>spring-boot-starter-web</artifactId>  
</dependency>
```


的注解清单


	描述
	被注释的元素必须为 false
	同@AssertFalse
	被注释的元素必须是一个数字，其值必须小于等于指定的最大值
	同DecimalMax
	带批注的元素必须是一个在可接受范围内的数字
	顾名思义
	将来的日期
sent	现在或将来
	被注释的元素必须是一个数字，其值必须小于等于指定的最大值
	被注释的元素必须是一个数字，其值必须大于等于指定的最小值
	带注释的元素必须是一个严格的负数（0为无效值）
ero	带注释的元素必须是一个严格的负数（包含0）
	同StringUtils.isNotBlank
	同StringUtils.isNotEmpty
	不能是Null
	元素是Null
	被注释的元素必须是一个过去的日期
nt	过去和现在
	被注释的元素必须符合指定的正则表达式
	被注释的元素必须严格的正数（0为无效值）
ro	被注释的元素必须严格的正数（包含0）
	带注释的元素大小必须介于指定边界（包括）之间


2
























			<div>👍 2</div>				
			<div>↗</div>				
			<div>💬</div>				
			<div>🔖</div>				
			<div>📱</div>				
			<div>&lt;</div>				
			<div>&gt;</div>				
返回数据及异常 <b>统一封装</b>			阅读数 8115				
情况下对数据返回的格式可能会有一个统一的要求，一般会包括状态码、信息及数据三部分。举个... 博文 来自： <a href="#">icarusliu的专栏</a>							
点什么							
定义业务错误信息			阅读数 50				
参数校验器，可以很方便的进行业务参数校验。引入下面连个jpom文件 &lt;!--StringUtils工具... 博文 来自： <a href="#">qq_29499993的博客</a>							
规范化 <b>封装统一</b> 返回数据格式和异常处理			阅读数 1924				
blog.csdn.net/ZHWang102107/article/details/82931584本文章是转载收藏1、统一响应数据格... 博文 来自： <a href="#">llziseweiqiu的博客</a>							
建一个后台管理（三）：全局异常处理、aop、 <b>validator</b> 校验			阅读数 211				
springboot环境搭建，引入了mybatis、generator等相关依赖。springboot搭建一个后台管理（一... 博文 来自： <a href="#">LiLrui的博客</a>							
全局异常 <b>统一</b> 处理			阅读数 785				
异常统一处理			博文 来自： <a href="#">Anwer.Ai.L</a>				
validate验证注解国际化配置文件如何自定义路径？							
的hibernate-validator还挺好用的，可是在框架中默认是在resource文件夹下，如何修改其默认路径？我... 论坛							
使用 <b>validator</b> 进行参数校验			阅读数 1165				
提供接口与用户交互(获取数据、上传数据等)，由于这个过程需要用户进行相关的操作，为了避免出... 博文 来自： <a href="#">大大肉包博客</a>							
<b>Validator</b> 学习笔记			阅读数 63				
ator笔记1背景开发过程中，后台的参数校验是必不可少的，本文关于spring-boot-starter-validat... 博文 来自： <a href="#">Little_fxc的博客</a>							
用hibernate <b>validator</b> 校验			阅读数 404				
开发中经常需要写一些字段校验的代码，比如字段非空，字段长度限制，邮箱格式验证等等，写这些... 博文 来自： <a href="#">殇莫忆的博客</a>							
<b>Validator</b> 校验			阅读数 549				
natevilidator主要使用注解的方式对bean进行校验，初步的例子如下所示：packagecom.query.im... 博文 来自： <a href="#">zhuchunyan_ajjia...</a>							
<b>alidator 完美实现+统一封装错误提示</b> - 小..._CSDN博客							
Hibernate <b>Validator</b> + <b>统一</b> 异常处理 - weixi..._CSDN博客							
全局异常处理 与 Hibernate <b>Validator</b> 校验框架整合			阅读数 1492				
ator校验框架的使用Springboot已经集成了hibernate-validator，不需要引入maven，其他框架也... 博文 来自： <a href="#">Anenan的博客</a>							
iu81		 小时光的胭脂		 llziseweiqiu		 LiLrui	
文章		19篇文章		51篇文章		5篇文章	
千里之外		排名:千里之外		排名:千里之外		排名:千里之外	
<a href="#">关注</a>		<a href="#">关注</a>		<a href="#">关注</a>		<a href="#">关注</a>	
错误,HttpPost、HttpGet关于URL重定向区别 - 小单的...							
返回数据及异常 <b>统一封装</b> - icarusliu的专..._CSDN博客							
Hibernate <b>Validator</b> + <b>统一</b> 异常处理			阅读数 187				
需要对一些参数进行校验比如非空判断,字符长度判断之类的，如果参数很多的话会给自己添加很多... 博文 来自： <a href="#">weixin_39361917...</a>							

alidation 统一返回错误信息

阅读数 889

root包compilegroup:'org.springframework.boot',name:'spring-boot-starter-validation'然后创... 博文 来自: jaryun260716001...

l):springboot全局异常处理和validator - ...\_CSDN博客

SpringBoot validator让数据更真实 - wei...\_CSDN博客

验国际化

阅读数 845

1Messages为文件名ValidationMessages.propertiesValidationMessages\_zh\_CN.properties 都... 博文 来自: maqingbin88888的...

用validator进行参数校验

阅读数 511

前后端分离的，为了保证前端传输数据的合法性，对参数进行校验就很有必要。hibernate-validator... 博文 来自: 小哥哥的博客

建一个后台管理(三):全局异常处理、aop、v...\_CSDN博客

用validator进行参数校验 - 小哥哥的博客 - CSDN博客

一对表单做数据校验

阅读数 37

用了Hibernate-validate校验框架校验规则@NotBlank:判断字符串是否为null或者是空串(去掉首尾... 博文 来自: 时光漫步的博客

alidation校验参数

阅读数 75

博文 来自: Mars小布

SpringBoot validator让数据更真实 - wei...\_CSDN博客

alidator国际化随笔 - jin861625788的专栏 - CSDN博客

用Hibernate Validator验证参数

阅读数 43

og.csdn.net/yufeianliu/article/details/81150169HibernateValidator是BeanValidation的参考... 博文 来自: WXXA

轻松搞定数据验证 (一)

阅读数 9420

'www.spring4all.com/article/1224对于任何一个应用而言，客户端做的数据有效性验证都不是安全... 博文 来自: 有天你会让我妒忌的.

码与它们的用途

阅读数 398

HTTP1.1客户端发送信息的servlet的每个可用的状态码，以及与每个状态码关联的标准消息。对这些... 博文 来自: T.Y 专栏

l): springboot全局异常处理和validator

阅读数 104

dependencies>>&lt;!-validation--&gt;&lt;dependency>&lt;groupId>org.springframework... 博文 来自: 流的博客

用hibernate-validator利用AOP实现统一参数校验

阅读数 2112

g.springframework.boot.spring-boot-starter-web1.5.9.RELEASEor 博文 来自: 千寻啊千寻

习之旅(五)---Hibernate Validator(完美的参数校验)

阅读数 168

别前言科技的进步是靠懒人推动的，我觉得没有任何毛病，但是这里说的懒人不是说纯懒的那种！开... 博文 来自: 阿飞的博客

删改查，接口统一封装！

阅读数 116

有的时候要用的后台接口，还是一件比较麻烦的事情。最近看了下SpringBoot试了下，以后再也不... 博文 来自: 男儿不展风云志, ...

使用Hibernate Validator验证参数


阅读数 1821


博客: https://ethendev.github.io/2018/05/17/spring-hibernate-validator/开发WEB应用时参... 博文 来自: yufeianliu的专栏


-api来验证spring-boot的参数


阅读数 48


验证前端传入的参数的合法性是一个必不可少的步骤，但是验证参数是一个基本上是一个体力活， ... 博文 来自: Lord\_of\_Code的博...


2






















- 用Hibernate Validator校验参数

阅读量 1838

tor后端开发中，常常需要对入参进行非空、非法格式校验，以确保数据的安全性和准确性。一般会... 博文 来自: Phycholee
- 1入门系列二 hibernate validator

阅读量 419

ot2.1系列的第二篇，代码可以从github下载 https://github.com/yinww/demo-springboot2.git... 博文 来自: 跟着互联网
- 全局异常抓取以及统一接口返回结果

阅读量 344

全局异常抓取以及日志log使用的文章，今天我们来单独优化一下这个全局异常抓取，并结合统一的... 博文 来自: 小目标青年的博客
- 非除当前进程（kill当前Tomcat进程）

阅读量 32

经常使用“ps-ef|grepomcat”查看进程，如下图是我操作的服务器上使用该命令后查看的结果：... 博文 来自: weixin\_34167043...
- validator 动态返回国际化提示

阅读量 0

去实现了读取指定国际化文件的校验器。1.MyMessages是自定义的国际化文件，放置在src的根目录... 博文 来自: weixin\_34261415...
- rest接口返回统一格式数据

阅读量 7836

rest接口返回统一格式数据为什么要统一格式？我们使用SpringBoot编写接口的时候，最好是返回一个... 博文 来自: alisonyu的博客
- 统一restful返回值及统一异常处理

阅读量 6271

数据：首先确定需要返回的json数据的格式，定义一个统一返回类packagecom.peixun.Response;i... 博文 来自: mtb的博客
- Spring Validation数据校验

阅读量 44

Validation是对hibernatevalidation的二次封装，添加自动校验的功能，并将校验结果封装到特定的... 博文 来自: 记录, 记录, 记录
- 校验工具-validator（重点）需复习

阅读量 70

xml添加依赖<!--validator--><!--groupId>javax.validation</gr... 博文 来自: Richard\_666的博客
- Spring Boot Validator校验

阅读量 7198

子；(2)国际化；(3)在代码中添加错误信息；(1)入门例子；Validator主要是校验用户提交的数... 博文 来自: linxingliang的专栏
- Validator在SpringMVC中的使用

阅读量 618

校验框架Hibernate-Validator使用说明文档简介SpringBoot集成hibernate-validation快速使用说... 博文 来自: Janche的博客
- Validator框架

阅读量 1万+

ootvalidator实际集成了Hibernatevalidator。主要是校验用户提交的数据的合理性的，比如是否为... 博文 来自: wingkoo1986的专栏
- 封装api统一格式返回值

阅读量 1136

使用前后分离的模式进行开发。统一返回方便前段进行开发和提醒用户操作出现错误原因。约定返回... 博文 来自: qq\_29499993的博客
- 系列(11): validator的使用

阅读量 933

的Web组件内部集成了hibernate-validator，我们可以直接使用。常用注解这些注解使用在pojo上... 博文 来自: 精确而优雅
- 使用异常自定义错误码

阅读量 3051

目前的前后端交互普遍通过Restful的形式，而错误处理仅仅通过HTTP状态码往往不满足需求；基... 博文 来自: shida's blog
- @Validation @Valid

阅读量 833

bernate validation的时候使用@Validated 是只用spring Validator 校验机制使用基于方法参数的... 博文 来自: 盲目的拾荒者的博客
- 封装结果集异常处理

阅读量 201

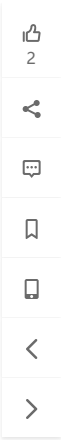
需要有一个结果集来封装controller返回的结果，让其构成一个统一的格式，封装的类我就不写了，网... 博文 来自: 宇锅锅的博客
- validator参数验证restful自定义错误码响应

阅读量 2338

应用中关于如何使用 Bean Validation API和hibernate-validator的文章已经很多，本文就不再重复... 博文 来自: shalousun的专栏
- validator国际化随笔

阅读量 7891

pringBoot提供一个web服务，需要对request的中的请求的数据进行校验一、使用dto中有一个职... 博文 来自: jin861625788的专栏



pring-boot应用hibernate-validator NoClassDefFoundError

阅读量 4936

mo代码：demo.zip最近排查一个springboot应用抛出hibernate.validatorNoClassDefFoundErro... 博文 来自： 横云断岭的专栏

ot Validator校验【从零开始学Spring Boot】

阅读量 6076

子；(2) 国际化；(3) 在代码中添加错误信息；(1) 入门例子； Validator主要是校验用户提交的... 博文 来自： AndyLizh的专栏

：基于 SpringBoot 的 Hibernate Validator 后台校验

阅读量 2670

ot的基础上使用 HibernateValidator进行参数校验？1.在拥有SpringBoot的情况下引入依赖org.hib... 博文 来自： 玩具熊猫的博客

封装优化，格式的统一（Spring Boot）

阅读量 2934

与必填项，无论业务成功与否，都将返回内容规范化为：1，http请求返回到最外层层对象2，封装... 博文 来自： selfimpr626

之战之全局异常捕获 实现参数异常检查返回统一错误信息

阅读量 3万+

之全局异常处理实现参数非法性检查在一个项目中的异常我们我们都会统一进行处理的，本文实现对... 博文 来自： sun\_t89的专栏

：#怎么读取html文件 c#如何跳出整个循环 c# throw的用法 c# 判断域名端口 c#前景怎么样 c#遍历datelist c#如何改变控件  
程 vs c# 文件读取image

没有更多推荐了， 返回首页



catoop

博客专家

私信

关注

TA的个人主页 >

原创

345

粉丝

4075

喜欢

1355

评论

1286

等级：

博客 1

访问：

608万+

积分：

2万+

排名：

360

勋章：



最新文章

Jenkins+Gitlab+Generic Webhook Trigger插件

Jenkins 插件之 SSH Pipeline Steps

shell 脚本单行 if 语句

Jenkins 添加 Slave 节点

微服务架构图

分类专栏



Spring Boot 学习

32篇



Android

73篇



应用部署

31篇



其他中间件


28篇





Linux


53篇


展开


2






















归档

2019年9月

2019年8月

2019年7月

2019年6月

2019年5月

2019年4月

2019年3月

2019年2月

6篇

16篇

16篇

14篇

12篇

5篇

5篇

2篇

展开

热门文章

Spring Boot 部署与服务配置

阅读数 173583

Spring Boot 静态资源处理

阅读数 169437

Spring Boot 事务的使用

阅读数 157536

httppost 302 错误, HttpPost、HttpGet 关于URL重定向区别

阅读数 135203

Spring Boot Shiro 权限管理

阅读数 135162

最新评论

Jenkins 插件之 SSH P...

catoop: [reply]LRXmrlirixing[/reply] wx: xz\_xia oshan

Jenkins 插件之 SSH P...

LRXmrlirixing: 博主, 能加个微信或者qq请教 ...

Spring Boot 动态数据源...

SeptDays: 那个 implements ImportBeanDefini tionRegistrar, EnvironmentAware 是不是可! ...

Spring Boot 使用Jav...

qq\_30052199: [code=java] [/code]

群晖NAS的docker中安装fr...

happyboymph: 感谢, 已经实现了, 网上大部分 都是介绍怎么设置客户端, 终于看到使用的介 ...



CSDN学院



CSDN企业招聘

QQ客服

客服论坛

kefu@csdn.net

400-660-0108

工作时间 8:30-22:00

关于我们 招聘 广告服务 网站地图

百度提供站内搜索 京ICP备19004658号

©1999-2019 北京创新乐知网络技术有限 公司

网络110报警服务 经营性网站备案信息

北京互联网违法和不良信息举报中心

中国互联网举报中心 家长监护 版权申诉



2





















