|  |
| --- |
| @NotNull **private** Integer **age**;  Integer 只能用@NotNull 这个注解，用notBlank会报错，传双引号或者null或者没有age这个字段就会报错  @NotBlank  这个注解可以检测参数是否为空和长度为0的空字符串，所有用这个最好。  //对于集合处理  @NotEmpty(message = "说明书不能为空")  private List<Integer> instructionsPictureIds; |
| 参考博客：  <https://blog.csdn.net/qq_19663899/article/details/79386043> |

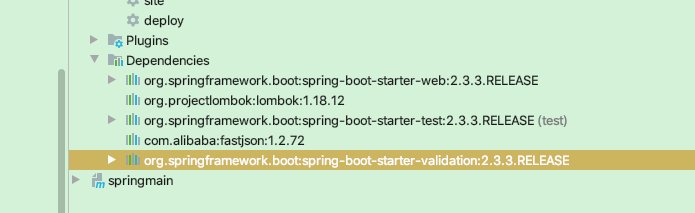
**Validated**

## springboot项目

|  |
| --- |
| <**parent**>  <**groupId**>org.springframework.boot</**groupId**>  <**artifactId**>spring-boot-starter-parent</**artifactId**>  <**version**>2.3.3.RELEASE</**version**>  <**relativePath**/> *<!-- lookup parent from repository -->* </**parent**> |

## 引入依赖

|  |
| --- |
| <**dependency**>  <**groupId**>org.springframework.boot</**groupId**>  <**artifactId**>spring-boot-starter-validation</**artifactId**> </**dependency**> |



## 实体类

|  |
| --- |
| **package** com.simpleness.entity;  **import** lombok.Data;  **import** javax.validation.constraints.Max; **import** javax.validation.constraints.Min; **import** javax.validation.constraints.NotBlank; **import** javax.validation.constraints.NotNull;  */\*\*  \** ***@author*** *iszengziqiang@163.com  \** ***@date*** *2020/9/1 16:43  \** ***@Version:*** *1.0  \** ***@desc*** *//****todo*** *\*/* @Data **public class** Man {   **private** Integer **id**;   **private** String **userName**;   @NotBlank  **private** String **passWord**;   @NotNull  **private** Integer **age**;   **private** String **emailAdd**;   **private** Boolean **deleted**;   @Max(10)  @Min(1)  **private** Integer **abc**; } |

## 接口使用

分为单个参数验证和实体类验证

|  |
| --- |
| **package** com.simpleness.controller;  **import** com.alibaba.fastjson.JSON; **import** com.simpleness.entity.Man; **import** org.springframework.validation.annotation.Validated; **import** org.springframework.web.bind.annotation.GetMapping; **import** org.springframework.web.bind.annotation.PostMapping; **import** org.springframework.web.bind.annotation.RequestBody; **import** org.springframework.web.bind.annotation.RestController;  **import** javax.validation.constraints.Max; **import** javax.validation.constraints.NotNull;  */\*\*  \** ***@author*** *iszengziqiang@163.com  \** ***@date*** *2020/9/1 16:42  \** ***@Version:*** *1.0  \** ***@desc*** *//****todo*** *\*/* @Validated @RestController **public class** validateController {   */\*\*  \* 单个验证  \*/* @GetMapping(**"/get"**)  **public** String getUserStr(@NotNull(message = **"name 不能为空"**) String name,  @Max(value = 99, message = **"不能大于99岁"**) Integer age,  @NotNull(message = **"数字"**) Integer xxx  ) {  **return "name: "** + name + **" ,age:"** + age + xxx;  }    @PostMapping(**"/post"**)  **public** String manTest(@RequestBody @Validated Man man) {  String s = JSON.*toJSONString*(man);  **return** s;  }   **public static void** main(String[] args) {  Man man = **new** Man();  }  } |

## 其他使用注解

|  |
| --- |
| 空检查  @Null 验证对象是否为null  @NotNull 验证对象是否不为null, 无法查检长度为0的字符串  @NotBlank 检查约束字符串是不是Null还有被Trim的长度是否大于0,只对字符串,且会去掉前后空格.  @NotEmpty 检查约束元素是否为NULL或者是EMPTY.  Booelan检查  @AssertTrue 验证 Boolean 对象是否为 true  @AssertFalse 验证 Boolean 对象是否为 false  长度检查  @Size(min=, max=) 验证对象（Array,Collection,Map,String）长度是否在给定的范围之内  @Length(min=, max=) Validates that the annotated string is between min and max included.  日期检查  @Past 验证 Date 和 Calendar 对象是否在当前时间之前  @Future 验证 Date 和 Calendar 对象是否在当前时间之后  @Pattern 验证 String 对象是否符合正则表达式的规则  数值检查，建议使用在Stirng,Integer类型，不建议使用在int类型上，因为表单值为“”时无法转换为int，但可以转换为Stirng为"",Integer为null  @Min 验证 Number 和 String 对象是否大等于指定的值  @Max 验证 Number 和 String 对象是否小等于指定的值  @DecimalMax 被标注的值必须不大于约束中指定的最大值. 这个约束的参数是一个通过BigDecimal定义的最大值的字符串表示.小数存在精度  @DecimalMin 被标注的值必须不小于约束中指定的最小值. 这个约束的参数是一个通过BigDecimal定义的最小值的字符串表示.小数存在精度  @Digits 验证 Number 和 String 的构成是否合法  @Digits(integer=,fraction=) 验证字符串是否是符合指定格式的数字，interger指定整数精度，fraction指定小数精度。  @Range(min=, max=) Checks whether the annotated value lies between (inclusive) the specified minimum and maximum.  @Range(min=10000,max=50000,message="range.bean.wage")  private BigDecimal wage;  @Valid 递归的对关联对象进行校验, 如果关联对象是个集合或者数组,那么对其中的元素进行递归校验,如果是一个map,则对其中的值部分进行校验.(是否进行递归验证)  @CreditCardNumber信用卡验证  @Email 验证是否是邮件地址，如果为null,不进行验证，算通过验证。  @ScriptAssert(lang= ,script=, alias=)  @URL(protocol=,host=, port=,regexp=, flags=) |

## group的使用

比如id字段，新增的时候不用传，修改的时候必须传，可以如下定义：

@Null(message = **"新增时Id必须为空"**, groups = Create.**class**)  
@NotNull(message = **"修改时id不能为空"**, groups = Update.**class**)  
**private** Integer **id**;

新增必须为空，修改必须有值

定义Create和Update接口，如下

**public interface** Default {  
}

**public interface** Create **extends** Default {  
}

**public interface** Update **extends** Default {  
}

使用时：

@ApiOperation(value = **"物资信息编辑保持"**)  
@PostMapping(**"v1/edit"**)  
**public** RespBean<?> update(@RequestBody @Validated(Update.**class**) MaterialRequest materialRequest) {}

根据更新和新增来做判断