链滴

Springboot 之分组验证以及自定义参数验证

作者：hjljy

原文链接：https://ld246.com/article/1589195464617

来源网站：链滴

springboot之前端参数验证

学习完简单的验证之后发现基本能满足百分之80的验证需求，接下来深入学习下验证。

分组验证

分组验证需要使用到@Validated 这个注解，是spring基于@Valid注解新增的功能。所以基本上在controller层接收参数的时候，可以用@Validated注解替换@Valid。不过@Validated注解不能放在类的字段上面，而@Valid注解可以放在类中的字段上面。所以如果类中的某个字段不是基础类型，但里面的参数需要校验的话，需要在字段上面加上@Valid 代码如下：

```
/**
 * @author 海加尔金鹰
 */
@Data
public class TestVo {

    @NotBlank(message = "name 不能为空字符串")
    private String name;

    @Valid
    private List<TestVo> testVos;  //验证集合里面的TestVo ，不加上无法进行验证
```

分组验证：例如同一个参数，在新增的时候，id不传 但是在修改的时候必传。这个时候可以利用grou来指定验证的规则组

创建两个不同的验证组:　关于是否继承默认验证组，建议都继承，如果不继承，在验证的时候只会证指定的字段

```
/**
 * @author 海加尔金鹰
 * 新增验证组
```

```
 */
public interface Insert {
}

/**
 * @author 海加尔金鹰
 *更新验证组
 */
public interface Update extends Default {
}
```

设置实体的验证规则

```
@Data
public class TestVo {

    @NotNull(message = "id 不能为空", groups = Update.class)
    @Null(message = "id必须为空", groups = Insert.class)
    private Integer id;

    @NotBlank(message = "name 不能为空字符串")
    private String name;
}
```

在controller进行验证规则设置

```
   @GetMapping("/id")
    public TestVo getTestVo(@RequestBody  @Validated({Update.class})   TestVo vo, BindingR
sult bindingResult) {
    //@Validated({Update.class}) 表示只验证Update这个组。由于这个组继承了默认default组 nam
也可以被验证，如果是Insert 就无法验证。
    //如果不配置{Update.class} 表示验证默认组的数据
        return vo;
    }
```

最后发送请求进行验证即可。

自定义验证

当自己的验证规则比较奇特的时候，可以自定义验证

第一步： 创建自定义验证注解

```
/**
 * @author 海加尔金鹰
 * 注意@Constraint(validatedBy = PhoneValidator.class) 这个注解 表明具体验证规则在PhoneVal
dator类里面
 */
@Constraint(validatedBy = PhoneValidator.class)
@Target({ElementType.METHOD, ElementType.FIELD})
@Retention(RetentionPolicy.RUNTIME)
@Documented
public @interface Phone {
    String message() default "手机号格式不合法";
```

```
    Class<?>[] groups() default {};

    Class<? extends Payload>[] payload() default {};
}
```

第二步创建具体验证PhoneValidator类

```
/**
 * @author 海加尔金鹰
 * @date 2020/5/11 17:52
 * @apiNote 手机号码验证
 */
public class PhoneValidator implements ConstraintValidator<Phone,String> {
    private Pattern pattern = Pattern.compile("1(([38]\\d)|(5[^4&&\\d])|(4[579])|(7[0135678]))\\
{8}");
    @Override
    public boolean isValid(String s, ConstraintValidatorContext constraintValidatorContext) {
        if(s!=null){
            return pattern.matcher(s).matches();
        }
        return true;
    }
}
```

第三步：和其他的验证注解一样使用即可

```
 @Phone
    private String phone;
```

总结：基本上到这里就能满足百分之九十五的验证需求。