# **65**\_代码实战:基于自定义注解方式,实现时间参数的统一校验

**儒猿架构官网上线**,内有石杉老师架构课最新大纲,儒猿云平台详细介绍,敬请浏 览

官网: <u>www.ruyuan2020.com</u> (建议 PC 端访问)

## 1、开篇

上节课思考了在创建订单的时候需要对教学时间进行验证,为了提高代码的可读性会使用 Spring 的注释验证方式。不过针对我们的验证要求会使用到 Spring 的自定义验证,通过一个手机号验证的方式让大家快速地了解该方式的运作原理。本节课进入码实战,带大家通过 Spring 自定义注解方式将时间参数进行统一校验。今天课程的内容包括以下几个部分:

- 定义 IsDate 注释接口
- 定义 DateValidator 类
- 应用 IsDate 注释到 Order 类

### 2、定义 IsDate 注释接口

有了上节课 Spring 自定义验证注释的基础,本节课就按照上节课的构建思路。首先,创建一个注释,它是一个接口,定义了对什么对象进行注释、什么时候运行注释以及使用什么验证规则。然后再定义具体的验证规则,最后应用到对应实体类上完成验证。

那么先来定义一个注释接口 IsDate,如图 1 所示,在 spring/annotion 目录下面创建 IsDate 接口。该接口定义了三个注释。其中@Target 注解表明自定义注解能够用来标记字段的,也就是说 IsDate 这个注释可以用来修饰字段。@Retention(RUNTIME)代表注解可以保留到运行时。

@Constraint 表明注解是用来作为校验约束的。其中的 validatedBy 属性,指向了一个类,就是这个类中提供了约束检查相关的逻辑,这个类就是后面会介绍的

DateValidator。接口中还定义了默认的 message 作为验证错误之后的信息:"时间格式不正确"。同时还定义了 formatter,这个会作为验证是传入的日期格式参数,后面 DateValidator 会根据这个 formatter 对日期的格式进行验证,如果不满足该格式就会返回 message 中定义的提示信息。

```
package com.ruyuan.little.project.spring.annotion;
                             1
                             2
                             3
Jan
                             4
                                   import ...
ittle
                             12
                             13
project
                                    /**
 spring 🖿
                             14
                                    * @author little 校验时间是否为合法
    annotion
                            15
                                    */
         IsDate
                             16
                                   @Target({ElementType.FIELD})
                                   @Retention(RetentionPolicy.RUNTIME)
                             17
    aspect
  cache
                             18
                                   @Constraint(validatedBy = DateValidator.class)
                             19
                                    public @interface IsDate {
  constants
  controller
                            20
  dao
                            21
                                        String message() default "时间格式不正确";
  ▶ □ dto
                             22
  enums
                             23
                                        String formatter() default "";
                            24
  event
                            25
                                        Class<?>[] groups() default {};
  exception
                            26
  handler
   interceptor
                             27
                                        Class<? extends Payload>[] payload() default {};
  mapper mapper
                             28
```

图 1 创建 IsDate 注释接口

#### 3.定义 DateValidator 类

定义完 IsDate 注释以后,就需要定义对应的验证类了,在里面会完成主要的验证功能。

如图 2 所示,在 spring/validator 目录下面创建 DateValidator 类,该类需要实现 ConstraintValidator 并且 Override isValid 方法,该方法首先会通过注释传入的 formatter 参数的解析获得日期的格式信息,例如: "yyyy-MM-dd"。然后根据这个格式信息对传入的 value,也就是具体的日期值进行转换,并且将转换的结果中 "dayOfMonth"信息与传入的日期信息进行匹配,保证转换以后的日期和原始的日期信息保持一致。最后,返回验证结果,如果返回为"True"表示验证通过,继续后面的业务逻辑,如果返回为"False"则返回 IsDate 注释接口定义的 message 信息,从而提示调用者调整日期参数的输入。

图 2 创建 DateValidator 类

# 4、应用 IsDate 注释到 Order 类

定义好 IsDate 注释以及对应的验证规则 DateValidator 类之后,就要将其应用到具体的订单上面。我们要达到的目的验证订单开始时间和结束时间不能为空,且按照按照"yyyy-mm-dd"的格式输入日期内容。由此我们利用 IsDate 来对 Order 类中的字段进行注释。

如图 3 所示,在 spring/dto 目录中创建 Order 类用来保存订单的实体内容,其中包括订单基本信息,例如: 消费者 Id、教师 Id、课程、状态等等。我们需要关注的是 startDate 和 endDate 两个字段,也就是教学的开始时间和结束时间。首先在上面使用@NotNull 这个是 Spring 自带的一个 validator,用来验证字段内容不能为空,如果为空会提示"启示/结束时间不能为空"。同时也定义了 IsDate 注释,也就是我们上面自定义的验证,在 message 中定义了验证不通过的消息,在 formatter中定义了验证的日期格式: "yyyy- MM -dd"。

```
▶ □ dao
                                private BigDecimal priceOfDay;
                     85

▼ Image: dto

                     86
    Comment
                                @NotNull(message = "起始时间不能为空")
@IsDate(message = "起始时间格式不正确",formatter = "yyyy-MM-dd")
                    87
    Consumer
                    88
    ConsumerCou
                     89
                                private String startDate;
    c 🔓 Coupon
                     90
   😊 🖆 Order
                                @NotNull(message = "结束时间不能为空")
                    91
    C Teacher
                                @IsDate(message = "结束时间格式不正确",formatter = "yyyy-MM-dd")
                    92
enums
                    93
                                private String endDate;
▶ ■ event
                    94
exception
                    95
handler
                                 * 订单创建时间
                    96
▶ interceptor
                                 */
```

图 3 在 Order 的 startDate 和 endDate 字段上加入注释

至此就完成了 Order 实体的日期验证方式的自定义,当对 Order 实体类进行赋值的时候,也就是在写入 startDate 和 endDate 字段时就会对输入的内容进行验证,如果验证规则会遵循 DateValidator 类中的定义,如果验证不通过会返回对应的提示信息。

# 5、总结

本节课通过 Spring 自定义验证的方法,将 Order 中 statrtDate 和 endDate 的验证方式进行类改造。分别进行了 IsDate 注释接口定义,DateValidator 类定义,并将 IsDate 注释到 Order 类的 statrtDate 和 endDate 字段上。下节课会思考,如何解 决订单、优惠券、积分的数据一致性问题。这里将代码给大家,下期见,拜拜。 友情提示:本章讲述代码只是部分核心代码,完整代码请查阅文末链接中代码,谢谢。