# Spring - validate String field in bean which is in fact a Date

Asked 5 years, 3 months ago Modified 5 years, 3 months ago Viewed 5k times

0

I'm writing tests for a bean that is a parameter in `@RestController` 's method. Bean POJO:

```
public class AddTownRequestBean
{
    @NotEmpty(message = "INVALID_REQUEST")
    @Length(min = 1, max = 30, message = "PARAMETER_OUT_OF_BOUNDS")
    private String name;
    @NotEmpty(message = "INVALID_REQUEST")
    @Length(min = 3, max = 4, message = "PARAMETER_OUT_OF_BOUNDS")
    private String typeCreated;
    @DateTimeFormat(pattern = "yyyy-MM-dd") //style = "S-", iso =
DateTimeFormat.ISO.DATE,
    private String foundationDate;

    getters and setters...
}
```

My question is related to `@DateTimeFormat` annotation. In documentation it is stated that this annotation:

> Can be applied to java.util.Date, java.util.Calendar, Long (for millisecond timestamps) as well as JSR-310 java.time and Joda-Time value types.

As one can see, there's no support of simple `String` type, but my POJO's date field is `String` . I already tested using `@DateTimeFormat` as outlined above, also with commented parameters, mutually excluded every time. And obviously it didn't work out.

So the question itself - is there any annotation or similar workaround to add a (let's call it) "validator" for specific date format in a `String` type variable that's meant to be a date?

java spring date validation spring-annotations

Share Edit Follow

asked Nov 6, 2019 at 15:14

Dmitriy Fialkovskiy
3,225 8 34 48

## 2 Answers

Sorted by: Highest score (default)

1

This question or similar one previously asked and answered. Below is the link to previous question. Please see if that answer helps you.

[Java String Date Validation Using Hibernate API]()

Share Edit Follow

answered Nov 6, 2019 at 15:52

Ramu
681 6 9

1 @tunapq That is exact answer given by the other contributor that I've linked. If you use someone else's answer, please give credit to them. – Ramu Nov 6, 2019 at 16:19

To be honest, your answer is not really an answer, rather a comment, but I upvote to give you ability to post comments=) – Dmitriy Fialkovskiy Nov 6, 2019 at 16:37

Thanks, I posted it in answer since I didn't have the ability to post it as comment. Thanks for the upvote. – Ramu Nov 6, 2019 at 17:39

0

You can create custom validator annotation for this case. Example

*DateTimeValid.class*

```
@Constraint(validatedBy = DateTimeValidator.class)
@Target({ElementType.METHOD, ElementType.FIELD})
@Retention(RetentionPolicy.RUNTIME)
public @interface DateTimeValid{
    public String message() default "Invalid datetime!";
    public String fomart() default "MM/dd/yyyy";
    public Class<?>[] groups() default {};
    public Class<? extends Payload>[] payload() default {};
}
```

*DateTimeValidator.class*

```
public class DateTimeValidator implements ConstraintValidator<DateTimeValid,
String> {

    private String dateFormat;

    @Override
    public void initialize(DateTimeValid constraintAnnotation) {
        dateFormat = constraintAnnotation.fomart();
    }

    @Override
    public boolean isValid(String strDate, ConstraintValidatorContext context) {
        try {
            DateFormat sdf = new SimpleDateFormat(this.dateFormat);
            sdf.setLenient(false);
            sdf.parse(strDate);
        } catch (Exception e) {
            return false;
        }
        return true;
    }
}
```

*Usage*

```
@DateTimeValid(fomart="...", message="...")
private String foundationDate;
```

Edit: @Ramu: This code from my project I have done before. But yeah, I read the link and it is the same idea with above code

Share Edit Follow

edited Nov 6, 2019 at 16:37

answered Nov 6, 2019 at 16:05

tunapq

**362** 3 14

there are too many coincidences in your (as you state it) project, and in the answer in linked question. Very suspicious... – Dmitriy Fialkovskiy Nov 6, 2019 at 16:49

It is something like standard way to create annotation, but no problem just trying to help – tunapq Nov 6, 2019 at 17:00

I was wondering if there wasn't already such annotation available, instead of having to write my own for such a common case – Enrico Giurin May 26, 2020 at 12:53