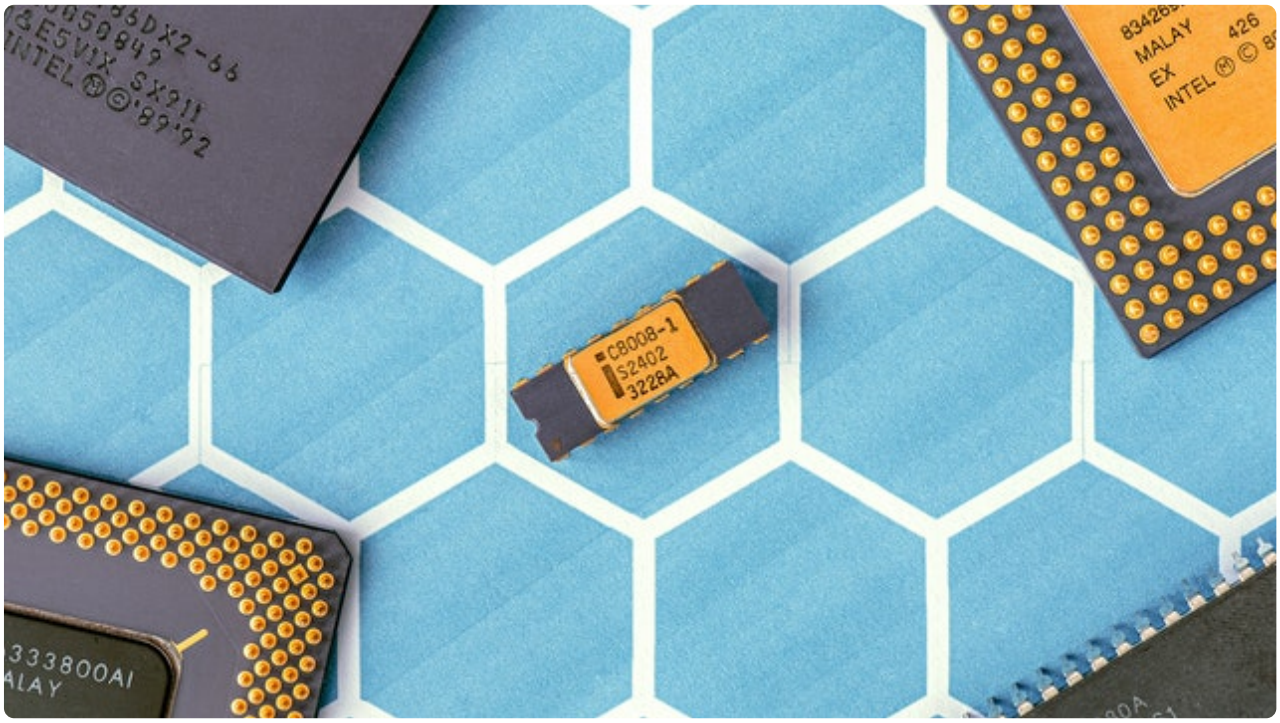
32 数据转换总结及常见面试题目解析

更新时间：2020-07-29 16:56:42



> 受苦的人，没有悲观的权利。——尼采

常见面试题目

0. Java Bean 是什么？

最初的定义：

> “A Java Bean is a reusable software component that can be manipulated visually in a builder tool.”

它的规范是 The JavaBeans 1.01 specification，可以到官方：https://www.oracle.com/java/technologies/javase/javabeans-spec.html 下载。

1. 什么时候使用 Validator？什么时候使用 Util 工具类的验证方法呢？

Util 工具类的验证一般是通用度较高的验证方法，这些往往可以包装成自定义注解的方式。

Validator 是只针对业务的特性验证需求，即普通验证逻辑不一致的验证，自定义一个特殊的验证方式，通用性上来说更低。

2. @Valid vs. @Validated

Spring Validation 验证框架对参数的验证机制提供了 @Validated（Spring's JSR-303 规范，是标准 JSR-303 的一个变种），javax 提供了 @Valid（标准 JSR-303 规范），配合 BindingResult 可以直接提供参数验证结果。

在检验 Controller 的入参是否符合规范时，使用 @Validated 或者 @Valid 在基本验证功能上没有太多区别。但是在分组、注解地方、嵌套验证等功能上两个有所不同。

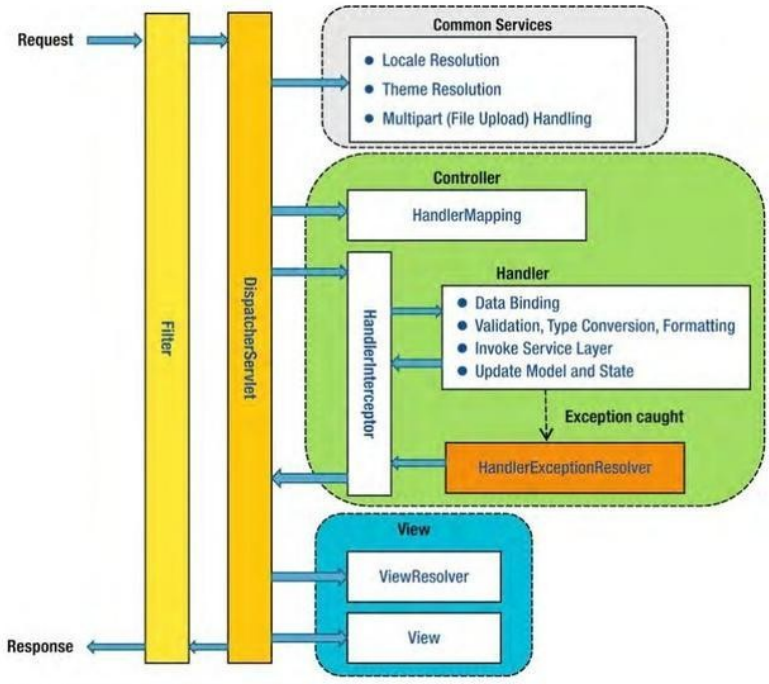


Figure 17-3. Spring MVC request life cycle

分组

@Validated：提供了一个分组功能，可以在入参验证时，根据不同的分组采用不同的验证机制，可参考高效使用 hibernate-validator 校验框架。

@Valid：作为标准 JSR-303 规范，不支持分组的功能。

注解作用位置

@Validated：可以用在类型、方法和方法参数上。但是不能用在成员属性（字段）上。

@Valid：可以用在方法、构造函数、方法参数和成员属性（字段）上。

3. JSR303 JSR-349 JSR 380 分别是什么？应用场景是什么？

Bean Validation 为 JavaBean 验证定义了相应的元数据模型和 API，Bean Validation 是一个运行时的数据验证框架，在验证之后验证的错误信息会被马上返回。

各个版本的规范对应关系如下：

- JSR 380 (Bean Validation 2.0)
- JSR 349 (Bean Validation 1.1)
- JSR 303 (Bean Validation 1.0)

4. Bean Validation 有哪些常用的验证注解？

Bean Validation 中内置的 constraint，常用的实现为：Hibernate Validator。

- **@Null**：被注释的元素必须为 null；
- **@NotNull**：被注释的元素必须不为 null；
- **@AssertTrue**：被注释的元素必须为 true；
- **@AssertFalse**：被注释的元素必须为 false；
- **@Min(value)**：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；
- **@Max(value)**：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；
- **@DecimalMin(value)**：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；
- **@DecimalMax(value)**：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；
- **@Size(max=, min=)**：被注释的元素的大小必须在指定的范围内；
- **@Digits (integer, fraction)**：被注释的元素必须是一个数字，其值必须在可接受的范围内；
- **@Past**：被注释的元素必须是一个过去的日期；
- **@Future**：被注释的元素必须是一个将来的日期；
- **@Pattern(regex=,flag=)**：被注释的元素必须符合指定的正则表达式；

Hibernate Validator 附加的 constraint 。

@NotBlank(message =)：验证字符串非 null，且长度必须大于 0；

@Email： 被注释的元素必须是电子邮箱地址；

@Length(min=,max=)：被注释的字符串的大小必须在指定的范围内；

@NotEmpty：被注释的字符串的必须非空；

@Range(min=,max=,message=)：被注释的元素必须在合适的范围内。

5. Spring Bean 的属性是如何绑定到 Bean 上的？

XML 配置 bean 方式如下：

```
<bean id="pojoBean" class="com.davidwang456.test.POJOBean">
    <property name="name" value="davidwang456"></property>
    <property name="age" value="10"></property>
    <property name="name" value="2019-12-24"></property>
</bean>
```

如果使用编程方式，该如何实现呢？

实现方式：

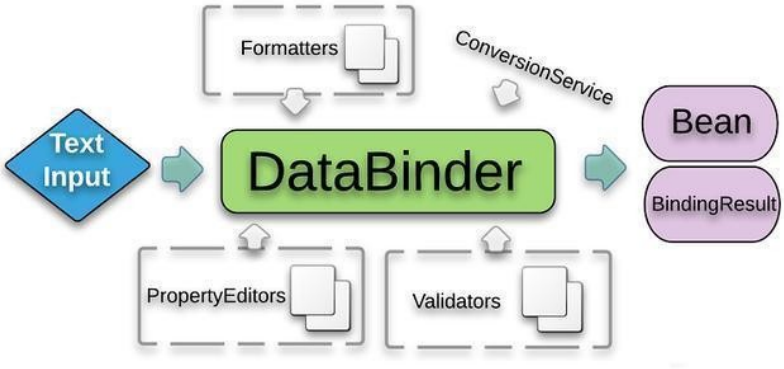
1. 使用 BeanWrapper 实现属性指定；
2. 使用 DataBinder 来实现属性指定。

6. 在 Spring MVC 中，如何将用户传来的时间格式的参数转换为 Java 的日期格式？

在使用 SpingMVC 框架的项目中，经常会遇到页面某些数据类型是 Date、Integer、Double 等的数据要绑定到控制器的实体，或者控制器需要接受这些数据，如果这类数据类型不做处理的话将无法绑定。

这里我们可以使用注解 @InitBinder 来解决这些问题，这样 SpingMVC 在绑定表单之前，都会先注册这些编辑器。一般会将这些方法些在 BaseController 中，需要进行这类转换的控制器只需继承 BaseController 即可。其实 Spring 提供了很多的实现类，如 CustomDateEditor、CustomBooleanEditor、CustomNumberEditor 等，基本上是够用的。

7. PropertyEditor Converter Formatter 的应用场景



Spring 早期使用 PropertyEditor 进行 Object 与 String 的转换。
到 Spring 3 后，Spring 提供了统一的 ConversionService API 和强类型的 Converter SPI，以实现转换逻辑。Spring 容器使用该系统来绑定 bean property values。

但是，除了格式转换，你还经常需要本地化 String values。也就是以当地格式展示，如货币、日期等。通用的 core.convert Converter SPI 不能直接完成格式化需求。基于此，Spring 3 引入了 Formatter SPI，相比 PropertyEditors 简单直接。
ConversionService 为 Converter SPI 和 Formatter SPI 提供了统一的 API。

8. JSR-354 是什么？应用场景是什么？

> Java: Money and Currency API

```
<dependency>
<groupId>org.javamoney</groupId>
<artifactId>moneta</artifactId>
<version>1.0.3</version>
</dependency>
```

JSR354 的设计目标：

1. 提供处理和计算货币金额的 API；
2. 定义货币和货币金额的类别，以及货币四舍五入；
3. 处理汇率；
4. 处理货币和货币金额的格式化和解析。

总结

校验作为业务逻辑有正面也有负面意义，而 Spring 不考虑这些，它认为验证设计（和数据绑定)不应该依赖于 Web 层，且易用，可以插入任何可用的验证器。基于上述的考虑，Spring 提供了一个 Validator 接口，这是一个基础组件，可以用于应用程序的每一层。

数据绑定机制允许将用户输入动态绑定到应用程序的 JavaBean 对象（或用来处理用户输入的任何对象）。Spring 提供了 DataBinder 类实现数据绑定机制。

Validator 和 DataBinder 组成了 validation 包，主要但不限于在 MVC Framework 中应用。BeanWrapper 是 Spring Framework 中的一个基础组件，在很多地方都用到，但是开发者一般不会直接使用。。如果开发者想使用它，最可能是在数据绑定场景下使用。

Spring 的 DataBinder 和底层的 BeanWrapper 都使用 PropertyEditor 来解析和格式化属性值。

PropertyEditor 是 JavaBean 所特有的，Spring 3 引入了“core.convert”包，既提供了一个通用类型转换工具，也提供了高级“format”包来格式化 UI 字段值。

Spring 中的这些新的程序包用起来可能比 PropertyEditor 简单。

}

31 Converter还是Formatter?

33 SpEL String应用示例及背后原理探究