Validation in Spring Boot with Hibernate Validator

By Ramesh Fadatare (Java Guides)

Java Bean Validation Basics

- **We validate a Java bean with the standard framework — JSR 380, also known as Bean Validation 2.0.**
- **Validating user input is a super common requirement in most applications. And the Java Bean Validation framework has become the de facto standard for handling this kind of logic.**
- **JSR 380 is a specification of the Java API for bean validation and this ensures that the properties of a bean meet specific criteria, using annotations such as @NotNull, @Min, and @Max.**
- **Hibernate Validator is the reference implementation of the validation API.**

Important Bean Validation annotations

- **@NotNull** validates that the annotated property value is not null.
- **@Size** validates that the annotated property value has a size between the attributes min and max; can be applied to String, Collection, Map, and array properties.
- **@Min** validates that the annotated property has a value no smaller than the value attribute.
- **@Max** validates that the annotated property has a value no larger than the value attribute.
- **@Email** validates that the annotated property is a valid email address.
- **@NotEmpty** validates that the property is not null or empty; can be applied to String, Collection, Map or Array values.
- **@NotBlank** can be applied only to text values and validates that the property is not null or whitespace.

Validation in Spring Boot

- Spring boot provides good integration support with Hibernate validator
- We will use Hibernate Validator, which is one of the reference implementations of the bean validation API
- Starting with Boot 2.3, we need to explicitly add the spring-boot-starter-validation dependency:

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

Development Process

1. Add maven dependency:

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

2. Apply validation annotations to a PostDto bean. For example, **@NotNull, @NotBlank, and @Size** annotations etc.
3. Enable validation on Spring Rest Controller by adding **@Valid** annotation in addition to **@RequestBody**
4. To customize response validation we need to extend **ResponseEntityExceptionHandler** class and override ***handleMethodArgumentNotValid(MethodArgumentNotValidException ex, HttpHeaders headers, HttpStatus status, WebRequest request)*** method.