Applying Built-in Validation Rules

INTRODUCTION

Sekhar Srinivasan

@sekharonline4u www.sekhartheguru.net

Road map

Overview of Spring MVC
Validations

Set Up Development
Environment

Check for a required Field

Validate Number range, Regular Expressions & more...

Create Business Validation Rules

Create Custom Java Annotation.

Spring MVC Form Validations Overview

Java Standard Bean Validation API

API defined in JSR 303 and 309

- Defines Meta model
- API for Entity Validation

Supported by both Server-side and Client-side

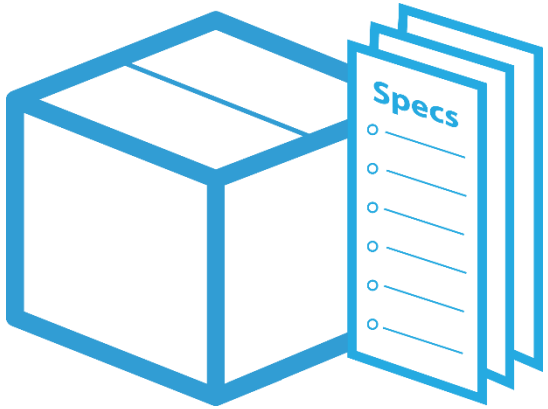
JSR: Java Specification Request

Bean Validation Annotations

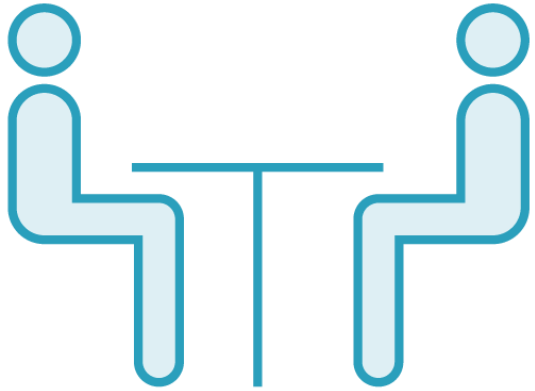
- @NotNull
- @Min
- @Max
- @Size
- @Pattern
- @Past
- @Future

Setting up Development Environment for Form Validations

Java Standard Bean Validation API



Specification



Vendor Independent

Portable


We need something that implements the Specification
We have
Fortunately
Hibernate for

Hibernate

Started as an ORM Project

Hibernate Search

Hibernate OGM

Hibernate Validator

Hibernate Validator

- Fully compliant JSR-303 / 309
- Not tied to the ORM or Database models
- Separate project used for performing Validations
- Fully compliant with Java Beans Validation API

Setup Process

Download Hibernate Validator Jar Files

Add the downloaded Jar files to the Project

Demo: Applying Validation Rules

Roadmap

1. Add Validation rule to the Bean field
2. Add Support to Controller class for supporting Validations
3. Update the View for displaying error messages

Demo: Applying Validation Rules Size, Min, Max, Pattern and Past

Pre-processing Code Validation Using @InitBinder

InitBinder annotation works as a pre-processor

Steps: Pre-processing Code Validation Using @InitBinder

Create a Method which provides the definition for the validation at Controller class

Annotate the method using @InitBinder annotation

Recap

Creating Custom Validation Rules

How to Create Custom Validation Rules

Create our own Annotation

To Create our own custom validation rules for our application we need to follow 4 steps

Step 1: Create a New Annotation

```
@Constraint(validatedBy = ValidatorClassName.class)
@Target( { ElementType.METHOD,                    )
           ElementType.FIELD}
@Retention(RetentionPolicy.RUNTIME)
public @interface AnnotationName {
    //define default methods
    //define default groups
    //define default payloads
}
```

Step 2: Define a Validator

```
public class ValidatorClass implements
ConstraintValidator<AnnotationName,
    DataType> {
        …
       @Override
    public Boolean isValid( … ) {
        //write your custom validation rules
    }
}
```

```
@CustomAnnotationName
private String fieldname;
```

Step 3

Use Custom Annotation over property or method

```
<form:errors path="fieldname" cssClass="error" />
```

Step 4

Update the view to display the error message

Demo: Creating Custom Validation Rules

Summary

How to apply Built-in Validation rules

- Set up Development Environment
- How to use Built-in Validations
- How to use Pre-processor code
- How to create our Custom Validation rules