Spring MVC Validations

Spring has provided inbuilt validations framework. Since we are using Spring framework validation implementation, we will have to use Spring Form tags to get the errors and set the form bean and variable names.

Validations will be applied by using annotations to fields of pojo class. As a developer we need not to write logic for everything basic validations they already provided so it's easy to use.

Here are some of the most commonly used validation annotations.

1. @Min(value=) – Checks whether the annotated value is greater than or equal to the specified minimum value.
2. @Max(value=) – Checks whether the annotated value is smaller than or equal to the specified maximum value. ‘
@Min(value = 18, message = "Age must be greater than 18")

@Max(value = 25, message = "Age must be smaller than 25")

private int age;

3. @NotNull – Checks that the annotated value is not null.
4. @NotBlank – Checks that the annotated character sequence/string is not null and the trimmed length is greater than 0.
5. @NotEmpty – Checks that the annotated element is not null and not empty.
// @NotNull: The CharSequence, Collection, Map or Array object is not null, but can be empty.

// @NotEmpty: The CharSequence, Collection, Map or Array object is not null and size > 0.

// @NotBlank: The string is not null and the trimmed length is greater than zero.

@NotEmpty(message = "First name cannot be null and must have size greater than 0")

private String firstName;

@NotNull(message = "Second name must not be null, empty value/space can be considered")

private String lastName;

@NotBlank(message = "Username must not be null and must contain 1 or more characters")

private String userName;

6. @Email – Checks whether the specified character sequence/string is a valid email address.
@Email(message = "Email should be valid")

private String email;

7. @Pattern(regex=, flags=) – Checks that the annotated string matches the regular expression considering the given flag matches.
// The regex specifies that the password can contain characters from a to z, A to Z and 0-9 only,

// also it must be in between 6 to 10 characters long.

@Pattern(regexp = "^[a-zA-Z0-9]{6,10}$")

private String password;

8. @AssertFalse – Checks that the annotated element is false.
9. @AssertTrue – Checks that the annotated element is true.
@AssertTrue

private boolean isWorking;

@AssertFalse

private boolean isStudent;

10. @NegativeOrZero – Checks if the given element is smaller than or equal to 0.
11. @Null – Checks that the annotated value is null.
12. @Negative – Checks if the element is strictly smaller than 0.

13. @Positive – Checks if the element is strictly greater than 0.
14. @PositiveOrZero – Checks if the given element is greater than or equal to 0.

```
@Positive
private int operand1;

@Negative
private int operand2;

@PositiveOrZero
private int operand3

@NegativeOrZero
private int operand4;

@Null
private int nullVal;
```

15. @Size – Checks if the annotated element's size is between min value and max value provided (inclusive).

```
@Size(min = 10, max = 200, message = "About Me must be between 10 and 200 characters")
private String aboutMe;
```