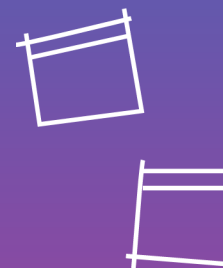




 **VEM SER**
DBC



Spring Web

Aula 4 – Validações de Dados



Sumário

- Status HTTP
- Spring validation
- ResponseEntity
- @ControllerAdvice e @ExceptionHandler

Status HTTP

- 1xxs - Respostas informativas
 - 2xxs - Sucesso!
 - 3xxs - Redirecionamento
 - 4xxs - Erros do cliente
 - 5xxs - Erros do servidor
-
- Classe java com todos: `org.springframework.http.HttpStatus`

Status HTTP

- Exemplos:
- HTTP 200 - OK
- HTTP 301 - Redirecionamento permanente
- HTTP 400 - requisição errada
- HTTP 404 - não encontrado
- HTTP 410 - desaparecido
- HTTP 500 - Erro interno do servidor
- HTTP 503 - Serviço indisponível

Spring Validation

- A dependência “**spring-boot-starter-validation**” incluirá todos os jars necessários para a validação.

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

Spring Validation

- Exemplo de validação:

```
public class PersonForm {

    @NotNull
    @Size(min=2, max=30)
    private String name;

    @NotNull
    @Min(18)
    private Integer age;

    public String getName() {
        return this.name;
    }
}
```

Spring Validation

- **@NotNull:** o valor da propriedade anotada não pode ser nulo
- **@AssertTrue:** o valor da propriedade anotada é true
- **@Size:** o valor da propriedade anotada deve estar entre os atributos min e max e pode ser aplicada à String, Collection, Map e array.
- **@Min:** valida se a propriedade anotada tem um valor que não seja menor que o valor especificado
- **@Max:** valida se a propriedade anotada tem um valor que não seja maior que o valor especificado
- **@Email:** valida se a propriedade anotada é um e-mail válido

- <https://reflectoring.io/bean-validation-with-spring-boot>

Spring Validation

- **@NotEmpty:** valida se a propriedade não é nula nem vazia e pode ser utilizada com String, Collection, Map ou array.
- **@NotBlank:** pode ser aplicada somente a a propriedades do tipo texto e valida se a propriedade não é nula ou formada por espaços vazios
- **@Positive:** aplicado a valores numéricos estritamente positivos
- **@PositiveOrZero:** aplicado a valores numéricos estritamente positivos incluindo o 0
- **@Negative:** aplicado a valores numéricos estritamente negativos
- **@NegativeOrZero:** aplicado a valores numéricos estritamente negativos incluindo o 0
- **@Past:** valida se o valor da data está no passado
- **@PastOrPresent:** valida se o valor da data está no passado ou no presente
- **@Future:** valida se o valor da data está no futuro
- **@FutureOrPresent:** valida se o valor da data está no futuro ou no presente

- <https://atitudereflexiva.wordpress.com/2020/09/06/bean-validation-customizacao>

Spring Validation

```
public class User {

    @NotNull(message = "Name cannot be null")
    private String name;

    @AssertTrue
    private boolean working;

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

    @Min(value = 18, message = "Age should not be less than 18")
    @Max(value = 150, message = "Age should not be greater than 150")
    private int age;

    @Email(message = "Email should be valid")
    private String email;

    // standard setters and getters
}
```

- <https://www.baeldung.com/javax-validation>



Vamos praticar!



Mas ainda não funcionou.



Spring Validation

- **@Validated** é para nível de classe, informamos ao Spring que a classe irá utilizar validação.
- **@Valid** é para os campos (parâmetros do método que serão validados).

```
@RestController
@RequestMapping("/pessoa")
@Validated
public class PessoaController {
```

```
@PostMapping
public ResponseEntity<Pessoa> create(@Valid @RequestBody Pessoa pessoa)
    return ResponseEntity.ok(pessoaService.create(pessoa));
}
```

Mas ainda não funcionou.




ResponseEntity

- Representa toda a resposta HTTP: status, cabeçalho e corpo.
- Para usar basta retorná-lo no método (endpoint).
- O Spring cuida do resto.
- Ele é genérico, podemos utilizar ele com qualquer objeto.

```
@GetMapping("/hello")
ResponseEntity<String> hello() {
    return new ResponseEntity<>("Hello World!", HttpStatus.OK);
}
```

@ControllerAdvice

- **@ControllerAdvice** Controller especial para lidar com exceções. (classe)
- **@ExceptionHandler** Exceção específica (método)
- **ResponseBodyExceptionHandler** Responsável por tratar as Exceções do Spring para os Controllers @RestTemplate.

```

@ControllerAdvice
public class CustomGlobalExceptionHandler extends ResponseExceptionHandler {
```

```
    @ExceptionHandler(RegraDeNegocioException.class)
    public ResponseEntity<Object> handleException(RegraDeNegocioException exception,
                                                HttpServletRequest request) {
```

Exercício #1

- Adicionar as seguintes validações na API:
 - **/pessoa**
 - **POST**
 - Nome não pode ser vazio nem nulo.
 - Data de nascimento não pode ser vazia e tem que ser no passado.
 - CPF não pode ser nulo e nem vazio, deve conter exatamente 11 caracteres.
 - **PUT**
 - Mesmas regras do POST.
 - Deve validar se a pessoa existe.
 - **DELETE**
 - Deve validar se a pessoa existe.

Task

- Adicionar as seguintes validações na API:
 - **/contato**
 - **POST**
 - Deve validar se a pessoa existe
 - Tipo de contato não pode ser nulo
 - Número não pode ser nulo e deve conter no máximo 13 caracteres
 - Descrição não pode ser vazia ou nula
 - **PUT**
 - Mesmas regras do POST
 - Deve validar se o id do contato existe
 - **DELETE**
 - Deve validar se o contato existe

Task

- **/endereco**
 - **POST**
 - Deve validar se a pessoa existe
 - Tipo de endereço não pode ser nulo
 - Logradouro não pode ser vazio e deve conter no máximo 250 caracteres
 - Número não pode ser nulo
 - Cep não pode ser vazio nem nulo e deve conter no máximo 8 caracteres
 - Cidade não pode ser vazio nem nulo e deve conter no máximo 250 caracteres
 - Estado e País não podem ser nulos
 - **PUT**
 - Mesmas regras do POST
 - Deve validar se o id do endereço existe
 - **DELETE**
 - Deve validar se o endereço existe

Task #2 Grupo (opcional)

- Utilizar Validação de dados em **todas as controllers**;
- Utilizar ResponseEntity e ResponseEntityExceptionHandler quando aplicável.

Obrigado!

DBC
DIGITAL BUSINESS COMPANY®



 /dbc.company

 /dbccompany

 /dbccompany.com.br

 /company/dbc-company