

19/05/2023

Version 4



SPRING

JÉRÉMY PERROUULT

A decorative graphic on the left side of the slide consisting of three parallel, wavy vertical lines. The outermost line is white, the middle line is a light blue color, and the innermost line is white. They are set against a dark blue background.

VALIDATION

VALIDATION

VALIDATION

- Le principe de la validation
 - Vérifier si les champs obligatoires sont remplis
 - Vérifier si une valeur est comprise entre x et y
 - Ces validations sont à faire, une à une, dans la méthode POST
- **Spring MVC** et l'API de validation vont nous éviter tout ça !
- Utilisation de l'annotation `@Valid` ou `@Validated`

VALIDATION

- Ajouter @Valid (ou @Validated)

Active la validation api-validator
(hibernate-validator)

```
@PostMapping("/produit/nouveau")
public String ajouterProduit(@Valid Produit produit, BindingResult result, Model model) {
    if (result.hasErrors()) {
        System.out.println("Le produit n'a pas été validé ...");
        return "form-produit";
    }

    return "redirect:/produits";
}
```

Permet de connaître les erreurs lors du Bind,
si erreur il y a

- Si utilisation de plusieurs @Valid
 - Il faut placer un BindingResult juste après un @Valid

VALIDATION

- 2 options sont possibles pour la validation
 - Utiliser une classe qui implémente l'interface "Validator"
 - Utiliser **Hibernate-Validator** et l'API Validation



API VALIDATOR

HIBERNATE VALIDATOR

HIBERNATE VALIDATOR

- Il suffit d'annoter les propriétés de la classe

```
public class Produit {  
    @NotBlank(message = "Le nom est obligatoire")  
    private String nom;  
}
```



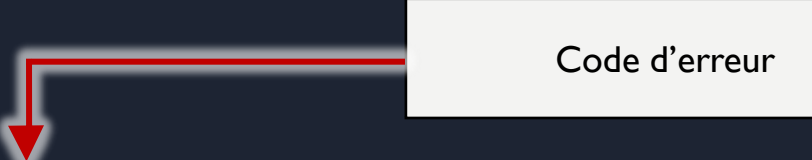
VALIDATOR

SPRING VALIDATOR

VALIDATOR

- Implémenter une classe qui implémente Validator

```
public class ProduitValidator implements Validator {  
    @Override public boolean supports(Class<?> cls) {  
        return Produit.class.equals(cls);  
    }  
  
    @Override public void validate(Object obj, Errors e) {  
        ValidationUtils.rejectIfEmptyOrWhitespace(e, "nom", "nom.empty", "Le nom doit être saisi");  
    }  
}
```



Code d'erreur

VALIDATOR

- La méthode `@RequestMapping` nécessite un `Validator`
 - Soit en première instruction de la méthode (dans ce cas, `@Valid` n'est plus nécessaire)

```
@PostMapping("/produit/nouveau")
public String ajouterProduit(@Valid Produit produit, BindingResult result, Model model) {
    new ProduitValidator().validate(produit, result);
    //...
}
```

- Soit via une méthode annotée de `@InitBinder` (directement dans le contrôleur)

```
@InitBinder
protected void initBinder(WebDataBinder binder) {
    binder.addValidators(new ProduitValidator());
}
```

EXERCICE

- Modifier le CRUD « produit »
 - Utiliser la validation
 - Messages d'erreur si le nom / prix n'est pas saisi ou mal saisi