Validation des données

# Démonstration 5 du module 4

|  |
| --- |
| Les objectifs de cette démonstration :   * Mettre en place la validation sur les BO * Activation de la validation sur les contrôleurs |

Contexte

* Nous continuons notre application représentant une partie du cœur de métier de l’ENI Ecole.

Déroulement

# Configuration de Spring Boot

* Ajout du starter spring-boot-starter-validation
* Ouvrir build.gradle, et ajouter :

//Validation des BO

**implementation** 'org.springframework.boot:spring-boot-starter-validation'

* + Faire un « Refresh Gradle Project »

## Annotations de validation sur Employe

* Utilisation des annotations @NotNull, @NotBlank, @Size pour valider les données transmises du client

**package** fr.eni.demo.bo;

**import** jakarta.persistence.\*;

**import** jakarta.validation.constraints.\*;

**import** lombok.\*;

**import** lombok.experimental.SuperBuilder;

@NoArgsConstructor

@AllArgsConstructor

@Getter

@Setter

@EqualsAndHashCode(of = { "immatriculation" })

@ToString

@SuperBuilder

@Entity

@Table(name = "EMPLOYEE")

@Inheritance(strategy = InheritanceType.***JOINED***)

**public** **class** Employe {

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

@Id

@Column(name = "EMPLOYEE\_ID")

**private** Integer id;

@ManyToOne

@JoinColumn(name = "CIVILITY\_ID")

**@NotNull**

**private** Civilite civilite;

@Column(name = "LAST\_NAME", nullable = **false**, length = 90)

**@NotBlank**

**@Size(max = 90)**

**private** String nom;

@Column(name = "FIRST\_NAME", nullable = **false**, length = 150)

**@NotBlank**

**@Size(max = 150)**

**private** String prenom;

@Column(nullable = **false**, unique = **true**, length = 255)

**@NotBlank**

**@Size(max = 255)**

**private** String email;

@Column(name = "EMPLOYEE\_REGISTRATION", nullable = **false**, unique = **true**, length = 100)

**@NotBlank**

**@Size(max = 100)**

**private** String immatriculation;

@Column(name = "HOME\_PHONE\_NUMBER", length = 12)

**private** String numDom;

@Column(name = "CELL\_NUMBER", length = 12)

**private** String numPortable;

@OneToOne(cascade = CascadeType.***ALL***, orphanRemoval = **true**, fetch = FetchType.***EAGER***)

@JoinColumn(name = "ADDRESS\_ID")

**@NotNull**

**private** Adresse adresse;

}

## Activation de validation dans EmployeController

* Ajout de l’annotation @Valid sur le paramètre de la méthode POST

…

**import** jakarta.validation.Valid;

…

@PostMapping

**public** ResponseEntity<?> ajoutEmploye(**@Valid** @RequestBody Employe employe) {

**try** {

employeService.ajouter(employe);

**return** ResponseEntity.*ok*(employe);

} **catch** (RuntimeException e) {

// Erreur BLL ou DAL

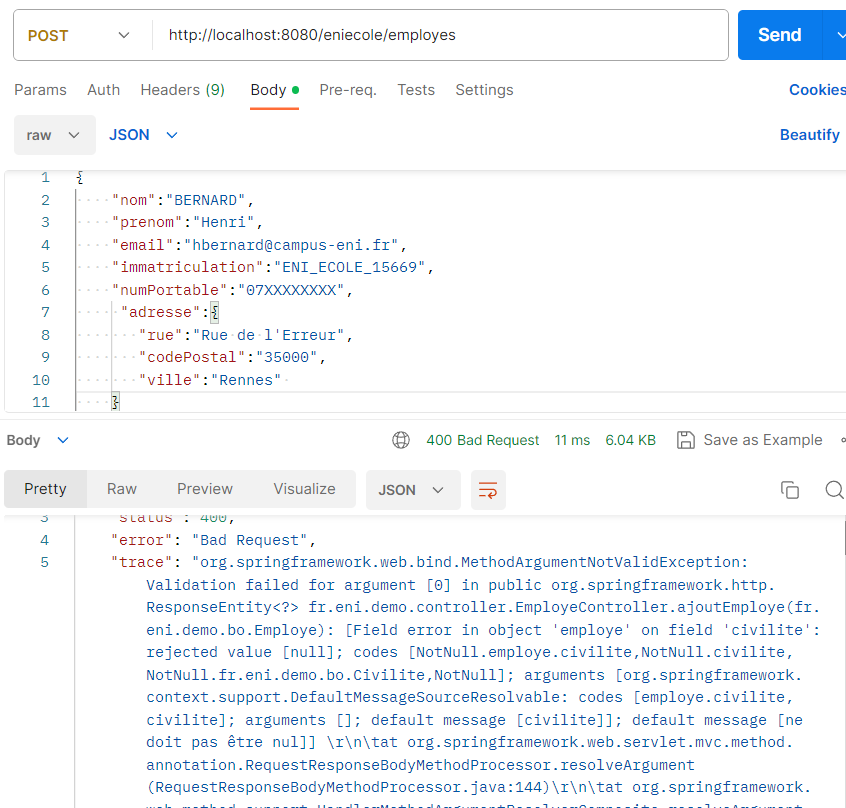
**return** ResponseEntity.*status*(HttpStatus.***NOT\_ACCEPTABLE***).body(e.getMessage());

}

}

# Exécution avec Postman

* Testons sans civilité :



* Le statut de la réponse (400)
* Nous avons une trace d’exception de Spring sur une erreur de validation
  + En regardant dans la trace ; il est possible de trouver le message d’origine
  + Il indique que le message par défaut est : ne doit pas être nul
  + Ce message provient de la validation de la librairie qui est par défaut internationalisée.

Cependant, l’information se perd dans une trace d’erreur Java.

Il est possible de gérer proprement