

JSF 2

16/04/2017

Walid YAICH

walid.yaich@esprit.tn

Bureau E204



Plan

- Managed Bean (ou backing bean)
- Les portées d'un bean géré
- Gérer les employés - ajouter
- Gérer les employés - Enum
- Gérer les employés - Lister
- Gérer les employés - Supprimer
- Gérer les employés - Modifier - créer les boutons
- Gérer les employés - Modifier - remplir le form
- Gérer les employés - Modifier - Mettre à jour dans la base
- employeldToBeUpdated is null, pourquoi ?
- Navigation
- Ajouter contrat à un employé existant
- Supprimer contrat

Managed Bean (ou backing bean)

Les beans gérés (Managed Bean) sont des classes Java **prises en charge par la FacesServlet**.

Les composants de l'interface utilisateur(IHM) sont **liés** aux propriétés du bean (backing bean) et peuvent invoquer des méthodes d'action, pour établir cette **liaison**, on utilise **EL**, le langage d'expressions.

Les portées d'un bean géré

@ApplicationScoped. Il s'agit de l'annotation la moins restrictive, avec la plus longue durée de vie. Les objets créés sont disponibles dans tous les cycles requête/ réponse de tous les clients utilisant l'application tant que celle-ci est active. Ces objets peuvent être appelés de façon concurrente et doivent donc être threadsafe (c'est-à-dire utiliser le mot-clé synchronized). Les objets ayant cette portée peuvent utiliser d'autres objets sans portée ou avec une portée d'application.

@SessionScoped. Ces objets sont disponibles pour tous les cycles requête/réponse de la session du client. Leur état persiste entre les requêtes et dure jusqu'à la fin de la session du client. Ces beans peuvent utiliser d'autres objets sans portée, avec une portée de session ou d'application.

@ViewScoped. Ces objets sont disponibles dans une vue donnée jusqu'à sa modification. Leur état persiste jusqu'à ce que l'utilisateur navigue vers une autre vue auquel cas il est supprimé. Ces beans peuvent utiliser d'autres objets sans portée, avec une portée de vue, de session ou d'application.

@RequestScoped. Il s'agit de la portée par défaut. Ces objets sont disponibles du début d'une requête jusqu'au moment où la réponse est envoyée au client.

@NoneScoped. Les beans gérés ayant cette portée ne sont visibles dans aucune page JSF ; ils définissent des objets utilisés par d'autres beans gérés de l'application. Ils peuvent utiliser d'autres objets avec la même portée.

Les portées d'un bean géré

Supprimer `@sessionScoped` du `loginBean` et voir ce qui se passe

welcome

logout

Gérer les employés - ajouter

Welcome.xhtml

```
<h:panelGrid columns="2">
```

```
<h:outputText value="login (email)" />
```

```
<h:inputText value="#{employeeBean.email}" />
```

```
<h:outputText value="password" />
```

```
<h:inputSecret value="#{employeeBean.password}" />
```

```
<h:outputText value="nom" />
```

```
<h:inputText value="#{employeeBean.nom}" />
```

```
<h:outputText value="Prenom" />
```

```
<h:inputText value="#{employeeBean.prenom}" />
```

```
<h:outputText value="Actif / Inactif" />
```

```
<h:selectBooleanCheckbox value="#{employeeBean.actif}" />
```

```
<h:commandButton action="#{employeeBean.addEmployee()}" value="Ajouter employe" />
```

```
</h:panelGrid>
```

login (email)

password

nom

Prenom

Actif / Inactif

Ajouter employe

Gérer les employés - ajouter

```
@ManagedBean
public class EmployeBean {

    private String prenom;
    private String nom;
    private String password;
    private String email;
    private boolean isActif;
    private Role role;

    @EJB
    EmployeeService employeeService;

    public void addEmploye(){
        employeeService.ajouterEmploye(new Employe(nom, prenom, email, password, isActif, role));
    }
}
```

Tester l'ajout d'un employé

Gérer les employés - Enum

```
<h:outputText value="Role" />
<h:selectOneMenu value="#{employeeBean.role}">
    <f:selectItem itemLabel="Selectionner le role" itemValue="" />
    <f:selectItems value="#{data.roles}" />
</h:selectOneMenu>
```

```
@ManagedBean
@ApplicationScoped
public class Data {

    public Role[] getRoles(){
        return Role.values();
    }
}
```

login (email)	<input type="text"/>
password	<input type="password"/>
nom	<input type="text"/>
Prenom	<input type="text"/>
Actif / Inactif	<input type="checkbox"/>
Role	<div>Selectionner le role ▼ Selectionner le role CHEF_DEPARTEMENT ADMIN INGENIEUR TECHNICIEN</div>
<input type="button" value="Ajouter employe"/>	

Tester l'ajout d'un employé avec Role

Gérer les employés - Lister

```
<h:dataTable value="#{employeBean.employees}" var="employe" border="1">
  <h:column>
    <f:facet name="header">
      <h:outputText value="login" />
    </f:facet>
    <h:outputText value="#{employe.email}" />
  </h:column>

  <h:column>
    <f:facet name="header">
      <h:outputText value="nom" />
    </f:facet>
    <h:outputText value="#{employe.nom}" />
  </h:column>

  <h:column>
    <f:facet name="header">
      <h:outputText value="prenom" />
    </f:facet>
    <h:outputText value="#{employe.prenom}" />
  </h:column>

  <h:column>
    <f:facet name="header">
      <h:outputText value="role" />
    </f:facet>
    <h:outputText value="#{employe.role}" />
  </h:column>

  <h:column>
    <f:facet name="header">
      <h:outputText value="Actif" />
    </f:facet>
    <h:outputText value="#{employe.actif == true ? 'Oui' : 'Non'}" />
  </h:column>
</h:dataTable>
```

```
private List<Employe> employees;
```

```
public List<Employe> getEmployes() {
    employees = employeeService.getAllEmployes();
    return employees;
}
```

login	nom	prenom	role	Actif
walid@esprit.tn	YAICH	Walid	ADMIN	Oui
mohamed.zitouni@ssiiconsulting.tn	zitouni	mohamed	CHEF_DEPARTEMENT	Non
aymen.ouali@ssiiconsulting.tn	ouali	aymen	INGENIEUR	Oui
bochra.bouزيد@ssiiconsulting.tn	bouزيد	bochra	CHEF_DEPARTEMENT	Oui
yosra.arbi@ssiiconsulting.tn	arbi	yosra	CHEF_DEPARTEMENT	Oui

login (email)	<input type="text"/>
password	<input type="password"/>
nom	<input type="text"/>
Prenom	<input type="text"/>
Actif / Inactif	<input type="checkbox"/>
Role	<input type="text" value="Selectionner le role"/>
Ajouter employe	<input type="button" value="Ajouter employe"/>

Gérer les employés - Supprimer

```
<h:column>
    <f:facet name="header">
        <h:outputText value="Action" />
    </f:facet>

    <h:commandButton action="#{employeBean.supprimer(employe.id)}" value="supprimer" />
</h:column>
```

login	nom	prenom	role	Actif	Action
walid@esprit.tn	YAICH	Walid	ADMIN	Oui	supprimer
mohamed.zitouni@ssiiconsulting.tn	zitouni	mohamed	CHEF_DEPARTEMENT	Non	supprimer
aymen.ouali@ssiiconsulting.tn	ouali	aymen	INGENIEUR	Oui	supprimer
bochra.bouزيد@ssiiconsulting.tn	bouزيد	bochra	CHEF_DEPARTEMENT	Oui	supprimer
yosra.arbi@ssiiconsulting.tn	arbi	yosra	CHEF_DEPARTEMENT	Oui	supprimer

login (email)

password

nom

Prenom

Actif / Inactif ☐

Role

Ajouter employe

```
public void supprimer(Integer employeeID){
    employeService.deleteEmployeeById(employeeID);
}
```

Gérer les employés - Modifier - créer les boutons

```
<h:commandButton action="#{employeBean.modifier(employe)}" value="Modifier"/>
```

login	nom	prenom	role	Actif	Action	
walid@esprit.tn	YAICH	Walid	ADMIN	Oui	supprimer	Modifier
mohamed.zitouni@ssiiconsulting.tn	zitouni	mohamed	CHEF_DEPARTEMENT	Non	supprimer	Modifier
aymen.ouali@ssiiconsulting.tn	ouali	aymen	INGENIEUR	Oui	supprimer	Modifier
bochra.bouزيد@ssiiconsulting.tn	bouزيد	bochra	CHEF_DEPARTEMENT	Oui	supprimer	Modifier
yosra.arbi@ssiiconsulting.tn	arbi	yosra	CHEF_DEPARTEMENT	Oui	supprimer	Modifier

login (email)

password

nom

Prenom

Actif / Inactif ☒

Role

Ajouter employe

Mettre a jour employe

```
<h:commandButton action="#{employeBean.mettreAJourEmploye()}" value="Mettre a jour employe" />
```

Rq : Chaque appui sur un bouton génère une nouvelle requête.

Gérer les employés - Modifier - remplir le form

```
<h:commandButton action="#{employeBean.modifier(employe)}" value="Modifier"/>
```

```
private String prenom;  
private String nom;  
private String password;  
private String email;  
private boolean isActif;  
private Role role;  
private List<Employe> employes;  
private Integer employeIdToBeUpdated;
```

```
public void modifier(Employe employe){  
    this.setEmail(employe.getEmail());  
    this.setPassword(employe.getPassword());  
    this.setActif(employe.isActif());  
    this.setPrenom(employe.getPrenom());  
    this.setNom(employe.getNom());  
    this.setRole(employe.getRole());  
    this.setEmployeIdToBeUpdated(employe.getId());  
}
```


Modifier

login (email)	<input type="text" value="walid@esprit.tn"/>
password	<input type="password"/>
nom	<input type="text" value="YAICH"/>
Prenom	<input type="text" value="Valid"/>
Actif / Inactif	<input checked="" type="checkbox"/>
Role	<input type="text" value="ADMIN"/>

Rq : Chaque appui sur un bouton génère une nouvelle requête.

Gérer les employés - Modifier - Mettre à jour dans la base

```
<h:commandButton action="#{employeBean.mettreAJourEmploye()}" value="Mettre a jour employe" />
```



```
public void mettreAJourEmploye(){  
    employeeService.updateEmploye(new Employe(nom, prenom,  
        email, password, isActif, role, employeIdToBeUpdated));  
}  
  
public void updateEmploye(Employe employe){  
    em.merge(employe);  
}
```

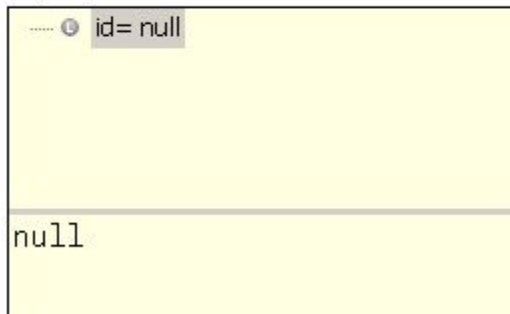
login (email)	<input type="text" value="walid@esprit.tn"/>
password	<input type="password" value="....."/>
nom	<input type="text" value="YAICH"/>
Prenom	<input type="text" value="Walid"/>
Actif / Inactif	<input checked="" type="checkbox"/>
Role	<input type="text" value="ADMIN"/>
Ajouter employe	<input type="button" value="Mettre a jour employe"/>

Rq : Chaque appui sur un bouton génère une nouvelle requête.

employeIdToBeUpdated is null, pourquoi ?

Caused by: [java.lang.NullPointerException](#)
at tn.esprit.timesheet.entities.Employe.<init>(Employe.java:84)
at tn.esprit.managedBeans.EmployeBean.mettreAJourEmploye(EmployeBean.java:51)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:497)

```
75  
76 public Employe(String nom, String prenom, String email, String password,  
77                boolean isActif, Role role, Integer id) {  
78     this.nom = nom;  
79     this.prenom = prenom;  
80     this.email = email;  
81     this.isActif = isActif;  
82     this.role = role;  
83     this.password = password;  
84     this.id = id;  
85 }
```



Vérifier le Scope

Navigation

Application Timesheet
 Bienvenue YAICH Walid [Logout](#)

[Page d'accueil](#)

Choisir une fonctionnalité :

[Gérer les employés](#)

[Ajouter un contrat pour un employé existant](#)

Copyright Esprit 2017

gererEmploye.xhtml

ajouterContrat.xhtml

Gérer les employés

login	nom	prenom	role	Actif	Action	
walid@esprit.tn	YAICH	Walid	ADMIN	Oui	supprimer	Modifier
mohamed.zitouni@ssiiconsulting.tn	zitouni	mohamed	CHEF_DEPARTEMENT	Non	supprimer	Modifier
aymen.ouali@ssiiconsulting.tn	ouali	aymen	INGENIEUR	Oui	supprimer	Modifier
yosra.arbi@ssiiconsulting.tn	arbi	yosraaa	CHEF_DEPARTEMENT	Oui	supprimer	Modifier

login (email)

password

nom

Prenom

Actif / Inactif ☐

Role

[Ajouter employe](#) [Mettre a jour employe](#)

Ajouter un contrat a un em

Employé	Type contrat	Salaire	Date Début
YAICH Walid	CDI	1600.0	01-02-2015
arbi yosraaa	CDI	2600.0	01-03-2010
zitouni mohamed	CDI	14.0	20-04-2017

Employe

Date debut

Salaire

Type contrat

[Ajouter contrat](#)

Welcome.xhtml

```
<ui:define name="title">Choisir une fonctionnalité : </ui:define><br/>
<ui:define name="content">
```

```
<h:link outcome="/pages/admin/gererEmploye" value="Gérer les employés"/>
<br />
```

```
<h:link outcome="/pages/admin/ajouterContrat" value="Ajouter un contrat pour un employé existant"/>
```

```
</ui:define>
```



Les chemins doivent être
relatif à webapp

ajouterContrat.xhtml

Ajouter un contrat a un employe

```

<tr>
  <td>
    <h:outputText value="Employee" />
  </td>
  <td>
    <h:selectOneMenu value="#{contratBean.selectedEmployeeId}" >
      <f:selectItems value="#{contratBean.employees}"
        var="employee" itemValue="#{employee.id}"
        itemLabel="#{employee}" />
    </h:selectOneMenu>
    <!-- toString defini dans Employee -->
  </td>
</tr>
</tr>
<tr>
  <td>
    <h:outputText value="Date debut" />
  </td>
  <td>
    <h:inputText id="date" value="#{contratBean.date}">
      <f:convertDateTime pattern="dd-MM-yyyy"/>
    </h:inputText>
  </td>
</tr>
<tr>
  <td>
    <h:message for="date" style="color:red" />
  </td>
  <td>
  </td>
</tr>
</tr>

```

Converter

Employé	Type contrat	Salaire	Date Début
YAICH Walid	CDI	1600.0	01-02-2015
arbi yosraaa	CDI	2600.0	01-03-2010
zitouni mohamed	CDI	14.0	20-04-2017

Employee

Date debut

Salaire

Type contrat

ContratBean.java

```
public class ContratBean {  
    private Date date;  
    private String typeContrat;  
    private Float salaire;  
    private List<Employe> employes;  
    private Integer selectedEmployeeId;  
    private List<Contrat> contrats;
```

```
@EJB  
EmployeeService employeeService;
```

```
@PostConstruct  
public void init(){  
    date = new Date(); //default value  
    typeContrat = "CDI"; //default value  
    employes = employeeService.getAllEmployes();  
}
```

```
public void ajouterContrat() throws ParseException{  
    SimpleDateFormat simpleFormat = new SimpleDateFormat("dd-MM-yyyy");  
    System.out.println("dateDebut : " + simpleFormat.format(date));  
    System.out.println("selectedEmployeeId : " + selectedEmployeeId);  
  
    Contrat contrat = new Contrat(date, typeContrat, salaire);  
    Employee selectedEmployee = new Employee();  
    selectedEmployee.setId(selectedEmployeeId);  
    contrat.setEmployee(selectedEmployee);  
    employeeService.ajouterContrat(contrat);  
}
```

```
public List<Contrat> getContrats() {  
    contrats = employeeService.getAllContrats();  
    return contrats;  
}
```

Date

Pour obliger JSF a utiliser le timezone du system, il faut rajouter dans web.xml :

```
<context-param>  
    <param-name>javax.faces.DATETIMECONVERTER_DEFAULT_TIMEZONE_IS_SYSTEM_TIMEZONE</param-name>  
    <param-value>true</param-value>  
</context-param>
```

Supprimer un employé qui a un contrat

- Vider la base
- Créer un employé
- Créer un contrat et l'affecter à l'employé créé
- Aller à la page gererEmploye
- Essayer de supprimer l'employé

Caused by:

```
com.mysql.jdbc.exceptions.jdbc4.MySQLIntegrityConstraintViolationException:  
Cannot delete or update a parent row: a foreign key constraint fails  
(`imputation`.`contrat`, CONSTRAINT `FK_kg96tqusokgdlypu18qqglxvs` FOREIGN KEY  
(`employe_id`) REFERENCES `employe` (`id`))  
    at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)  
    at
```