

TP 6

Développer des composants de la couche de présentation IHM avec Java et JSF2

L'objectif de ce TP est d'une part parcourir l'ensemble des tags ou composants fournis par **Oracle** et de l'implémentation de référence de JSF et, d'autre part, mettre en œuvre quelques composants fournis par Oracle.

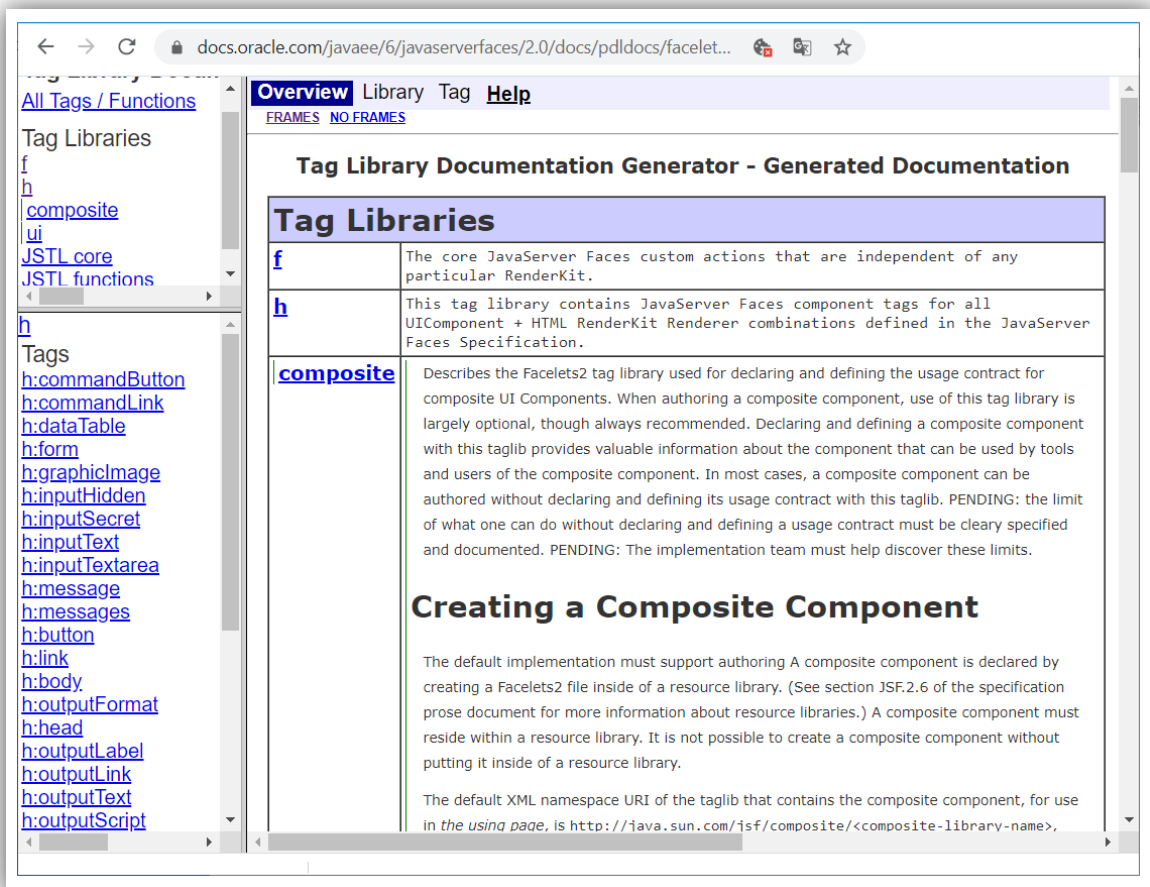
1. Contexte :

La technologie JSF a pour ambition de faciliter l'écriture d'applications WEB en offrant d'une part une architecture MVC et, d'autre part, une approche composant qui rappelle les interfaces graphiques natives.

L'objectif est de remplacer l'approche suivante « **je traite la requête HTTP et je renvoie une page HTML** » par « **j'envoie des composants (boutons, champs de texte, etc..) et je réagis aux actions de l'utilisateur** ».

2. Parcourir et analyser l'ensemble des composants offerts :

Explorez l'ensemble des composants de la balise **h** sur la page web suivantes :



The screenshot shows the Oracle JavaServer Faces 2.0 documentation page. The left sidebar lists various tag libraries and components. The main content area displays the 'Tag Libraries' section, which includes a table with the following information:

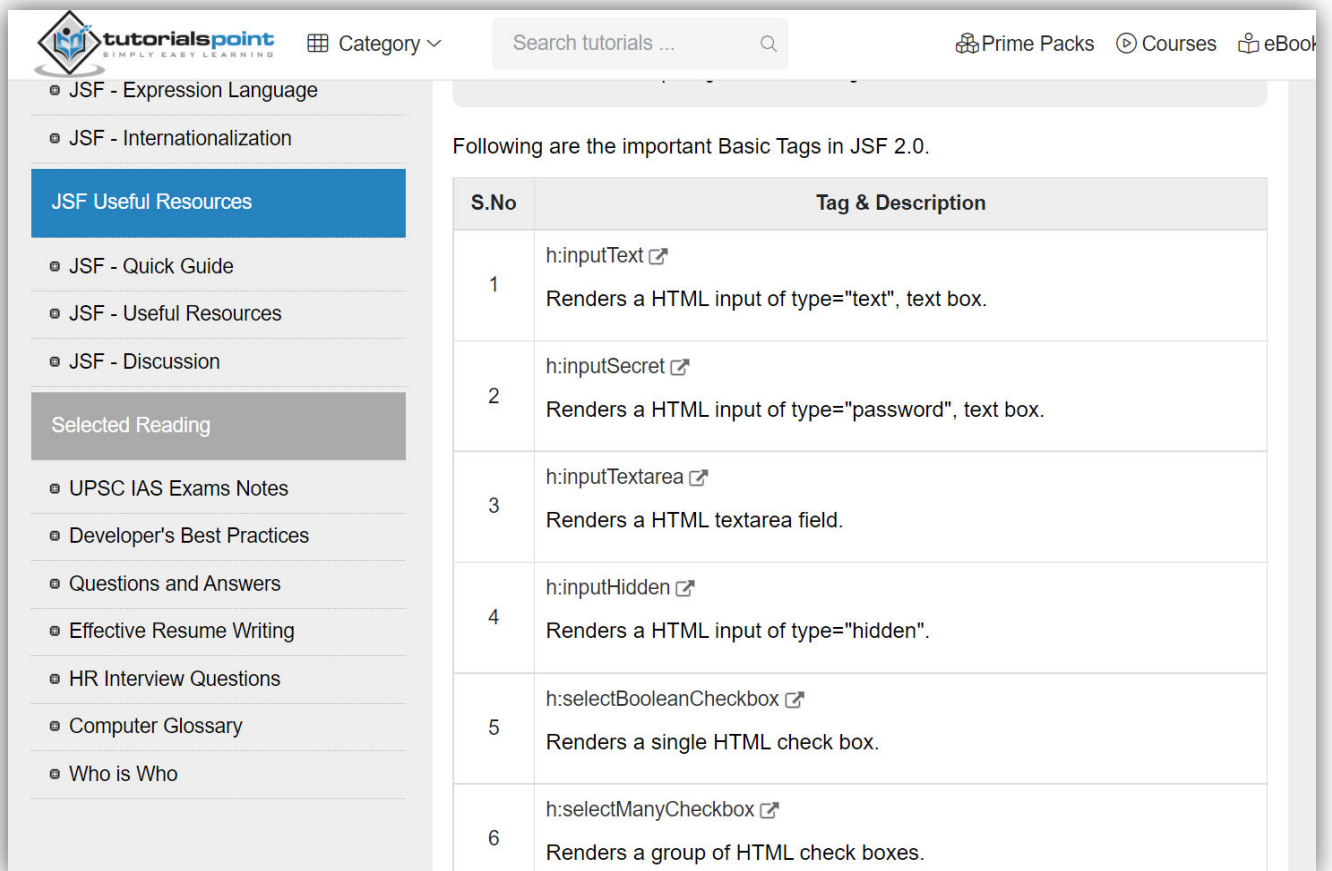
Tag Library	Description
f	The core JavaServer Faces custom actions that are independent of any particular RenderKit.
h	This tag library contains JavaServer Faces component tags for all UIComponent + HTML RenderKit Renderer combinations defined in the JavaServer Faces Specification.
composite	Describes the Facelets2 tag library used for declaring and defining the usage contract for composite UI Components. When authoring a composite component, use of this tag library is largely optional, though always recommended. Declaring and defining a composite component with this taglib provides valuable information about the component that can be used by tools and users of the composite component. In most cases, a composite component can be authored without declaring and defining its usage contract with this taglib. PENDING: the limit of what one can do without declaring and defining a usage contract must be clearly specified and documented. PENDING: The implementation team must help discover these limits.

Below the table, there is a section titled 'Creating a Composite Component' which provides detailed instructions on how to create and use a composite component within a resource library.

Explorez l'ensemble des composants des balises JSF des pages web suivantes :

(Lien principal = <https://www.tutorialspoint.com/jsf/>)

- https://www.tutorialspoint.com/jsf/jsf_basic_tags.htm
- https://www.tutorialspoint.com/jsf/jsf_facelets_tags.htm
- https://www.tutorialspoint.com/jsf/jsf_data_tables.htm
- https://www.tutorialspoint.com/jsf/jsf_convertor_tags.htm



The screenshot shows the tutorialspoint.com website. The left sidebar contains a menu with categories like 'JSF - Expression Language', 'JSF - Internationalization', 'JSF Useful Resources' (highlighted), 'JSF - Quick Guide', 'JSF - Useful Resources', 'JSF - Discussion', 'Selected Reading', 'UPSC IAS Exams Notes', 'Developer's Best Practices', 'Questions and Answers', 'Effective Resume Writing', 'HR Interview Questions', 'Computer Glossary', and 'Who is Who'. The main content area is titled 'Following are the important Basic Tags in JSF 2.0.' and contains a table with 6 rows of JSF tags and their descriptions.

S.No	Tag & Description
1	<code>h:inputText</code> Renders a HTML input of type="text", text box.
2	<code>h:inputSecret</code> Renders a HTML input of type="password", text box.
3	<code>h:inputTextarea</code> Renders a HTML textarea field.
4	<code>h:inputHidden</code> Renders a HTML input of type="hidden".
5	<code>h:selectBooleanCheckbox</code> Renders a single HTML check box.
6	<code>h:selectManyCheckbox</code> Renders a group of HTML check boxes.

3. Mise en œuvre de quelques composants

Créez un nouveau package **fr.doranco.tpjsf.entity** puis ajoutez une nouvelle classe (entité) **User** dans ce nouveau package. Cette classe **User** (qui est un *Entity Bean*) contiendra :

- les mêmes paramètres que pour **UserBean**,
- un constructeur vide,
- un constructeur avec tous les paramètres,
- les getters et setters.

Ajouter Maintenant dans **UserBean** la variable **userList** ci-dessous ainsi que son getter.

```
private static final List<User> userList = new ArrayList<User>(Arrays.asList(  
  
    new User("Benoît", "Leclerc", "homme", "27/10/1977", "benoit@doranco.fr", "medium"),  
    new User("Paul", "Andrieux", "homme", "12/06/1965", "paul.augustin@doranco.fr", "medium"),  
    new User("Laura", "Treich", "femme", "07/10/1987", "laura@doranco.fr", "medium"),  
    new User("Nathalie", "Tango", "femme", "17/07/1980", "nathalie@doranco.fr", "premium"))  
);
```

Créez une nouvelle classe de style nommée **table-style.css** dans le dossier css qui contient le code suivant :

```
.order-table {  
    border-collapse: collapse;  
}  
  
.order-table-header {  
    text-align: center;  
    background: none repeat scroll 0 0 #E5E5E5;  
    border-bottom: 1px solid #BBBBBB;  
    padding: 16px;  
}  
  
.order-table-odd-row {  
    text-align: center;  
    background: none repeat scroll 0 0 #FFFFFF;  
    border-top: 1px solid #BBBBBB;  
}  
  
.order-table-even-row {  
    text-align: center;  
    background: none repeat scroll 0 0 #F9F9F9;  
    border-top: 1px solid #BBBBBB;  
}  
  
.order-table-column {  
    background: none repeat scroll 0 0 #F9F9F9;  
}
```

Créez le fichier **register4.xhtml** à partir du fichier **register3.xhtml** puis modifiez-le comme suit :

```
<h:head>
  <!--Intégration du fichier CSS dans la page-->
  <h:outputStylesheet library="css" name="table-style.css" />
  <h:outputStylesheet library="css" name="common-style.css" />
</h:head>

<body>
  <h:form>
    <h4>
      Formulaire JSF d'inscription à l'<a
        href="https://www.doranco.fr/candidature/parcours/41?">Ecole DORANCO</a>
    </h4>
    <table>
      <tr>
        <td>Prénom:</td>
        <td><h:inputText value="#{userbean.prenom}" required="true"
          id="prenom" /> <h:message for="prenom" /></td>
      </tr>
      <tr>
        <td>Nom:</td>
        <td><h:inputText value="#{userbean.nom}" required="true"
          id="nom" /> <h:message for="nom" /></td>
      </tr>
      <tr>
        <td>Sexe:</td>
        <td><h:selectOneRadio value="#{userbean.genre}" required="true"
          id="genre">
          <f:selectItem itemLabel="Homme" itemValue="homme" />
          <f:selectItem itemLabel="Femme" itemValue="femme" />
        </h:selectOneRadio> <h:message for="genre" /></td>
      </tr>
      <tr>
        <td>Date de naissance:</td>
        <td><h:inputText value="#{userbean.datedenaissance}"
          id="datenaissance" required="true">
          </h:inputText> (mm-dd-yy) <h:message for="datenaissance" /></td>
      </tr>
      <tr>
        <td>Email:</td>
        <td><h:inputText value="#{userbean.email}" required="true"
          id="email" /> <h:message for="email" /></td>
      </tr>
      <tr>
        <td>Qualité de service:</td>
        <td><h:selectOneMenu value="#{userbean.niveaudeservice}">
          <f:selectItem itemLabel="Basic" itemValue="basic" />
          <f:selectItem itemLabel="Medium" itemValue="medium" />
          <f:selectItem itemLabel="Premium" itemValue="premium" />
        </h:selectOneMenu></td>
      </tr>
    </table>
```

```
<p>
  <h:commandButton value="Inscription" action="#" />
</p>
<h:form>
  <h4>Listes des participants inscrits</h4>

  <h:dataTable value="#{userbean.userList}" var="o"
    styleClass="order-table" headerClass="order-table-header"
    rowClasses="order-table-odd-row,order-table-even-row">

    <h:column>
      <!-- column header -->
      <f:facet name="header">Prénom</f:facet>
      <!-- row record -->
      <h:outputText value="#{o.prenom}" />
    </h:column>

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

    <h:column>
      <f:facet name="header">Email</f:facet>
      <h:outputText value="#{o.email}" />
    </h:column>

    <h:column>
      <f:facet name="header">Date de Naissance</f:facet>
      <h:outputText value="#{o.datedenaissance}" />
    </h:column>

    <h:column>
      <f:facet name="header">Niveau de Service</f:facet>
      <h:outputText value="#{o.niveaude-service}" />
    </h:column>

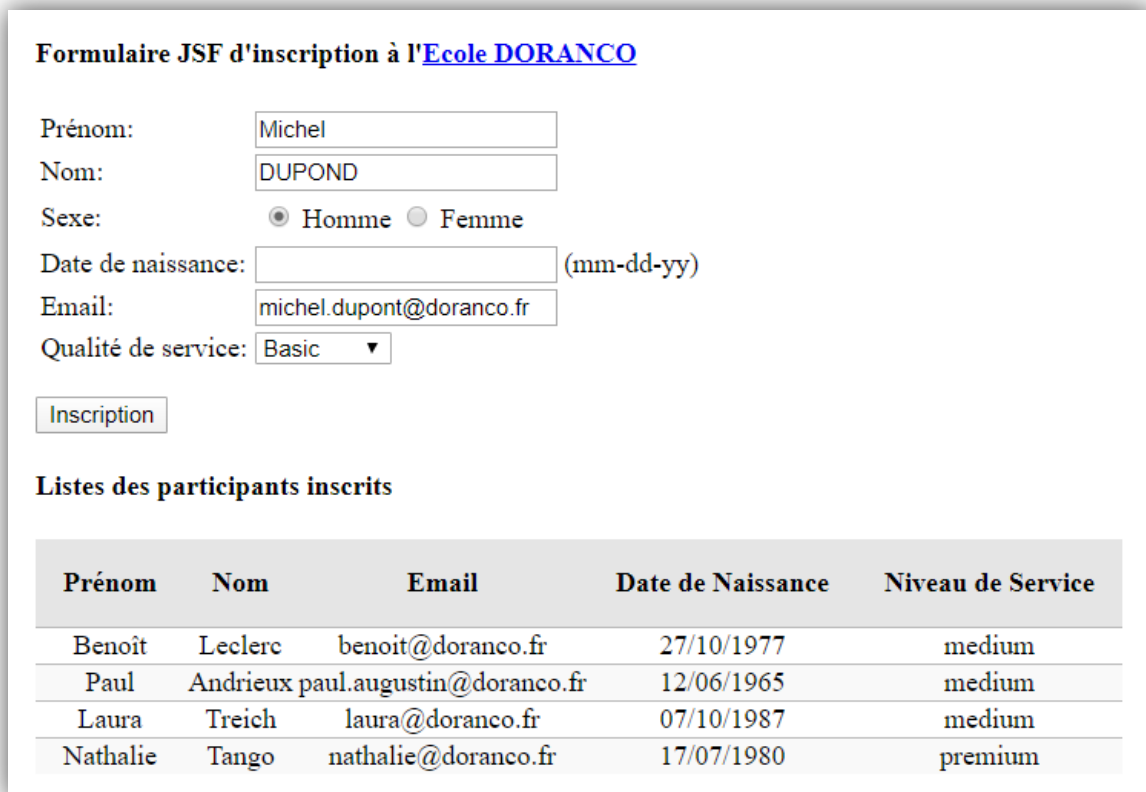
  </h:dataTable>
</h:form>
</h:form>
</body>
</html>
```

A ce stade, le déploiement du projet sur le serveur cible Tomcat donne le rendu ci-dessous :



The screenshot shows a web browser window displaying the DORANCO registration page. At the top left is the DORANCO logo with the text 'Espace Multimédia' and 'www.doranco.fr'. The main heading is 'Formulaire d'enregistrement Pour un stagiaire DORANCO'. Below it is a blue link 'Cliquez ici pour vous enregistrez...'. At the bottom, there is a section titled 'Ajouter votre footer ou garder celui par défaut'.

Cliquez sur le lien « **Cliquez ici pour vous enregistrez...** »
Vous devez obtenir le rendu suivant :



The screenshot shows a web browser window displaying the DORANCO JSF registration form. The title is 'Formulaire JSF d'inscription à l'Ecole DORANCO'. The form contains the following fields:

- Prénom: Michel
- Nom: DUPOND
- Sexe: ☒ Homme ☐ Femme
- Date de naissance: (mm-dd-yy)
- Email: michel.dupont@doranco.fr
- Qualité de service: Basic

Below the form is an 'Inscription' button. Underneath is a section titled 'Listes des participants inscrits' containing a table with the following data:

Prénom	Nom	Email	Date de Naissance	Niveau de Service
Benoît	Leclerc	benoit@doranco.fr	27/10/1977	medium
Paul	Andrieux	paul.augustin@doranco.fr	12/06/1965	medium
Laura	Treich	laura@doranco.fr	07/10/1987	medium
Nathalie	Tango	nathalie@doranco.fr	17/07/1980	premium

Identifiez les composants mis en place pour ce rendu et expliquez l'utilité des tags ci-dessous:

- **h:inputText**
- **h:selectOneRadio**
- **f:selectItem**
- **h:selectOneMenu**
- **h:commandButton**
- **h:dataTable**
- **h:column**
- **f:facet**
- **h:form**

Expliquez la différence faite entre les balises h et f ?

Maintenant, pour faire communiquer le formulaire d'inscription et le tableau des participants, modifiez **register4.xhtml** comme suit :

```
<p>  
    <h:commandButton value="Inscription" action="#{userbean.addAction()}" />  
</p>
```

Ici, la modification consiste à ajouter une action (donc une méthode) qui permet d'ajouter un utilisateur au tableau des participants.

Modifiez alors la classe **UserBean** en ajoutant la méthode suivante :

```
public void addAction() {  
    User user = new User(this.prenom, this.nom, this.genre, this.datedenaissance,  
                        this.email, this.niveaudeSERVICE);  
    userList.add(user);  
}
```


Redémarrez le serveur Tomcat puis affichez de nouveau le rendu :

Formulaire JSF d'inscription à l'[Ecole DORANCO](#)

Prénom:

Nom:

Sexe: ☒ Homme ☐ Femme

Date de naissance: (mm-dd-yy)

Email:

Qualité de service:

Listes des participants inscrits

Prénom	Nom	Email	Date de Naissance	Niveau de Service
Benoît	Leclerc	benoit@doranco.fr	27/10/1977	medium
Paul	Andrieux	paul.augustin@doranco.fr	12/06/1965	medium
Laura	Treich	laura@doranco.fr	07/10/1987	medium
Nathalie	Tango	nathalie@doranco.fr	17/07/1980	premium

Ajouter un participant en cliquant sur le bouton "Inscription".
Vous obtenez de nouveau le rendu suivant :

Formulaire JSF d'inscription à l'[Ecole DORANCO](#)

Prénom:

Nom:

Sexe: ☒ Homme ☐ Femme

Date de naissance: (mm-dd-yy)

Email:

Qualité de service:

Listes des participants inscrits

Prénom	Nom	Email	Date de Naissance	Niveau de Service
Benoît	Leclerc	benoit@doranco.fr	27/10/1977	medium
Paul	Andrieux	paul.augustin@doranco.fr	12/06/1965	medium
Laura	Treich	laura@doranco.fr	07/10/1987	medium
Nathalie	Tango	nathalie@doranco.fr	17/07/1980	premium
Michel	DUPOND	michel.dupont@doranco.fr	01/01/2020	medium

Nous allons maintenant ajouter une autre fonctionnalité au tableau : supprimer un participant.

Pour cela, ajouter la colonne suppression dans **register4.xhtml** :

```
<h:column>
    <f:facet name="header">Action</f:facet>
    <h:commandLink value="Supprimer" action="#{userbean.deleteAction(o)}" />
</h:column>

<h:column>
    <f:facet name="header">Action</f:facet>
    <h:commandButton value="Supprimer" action="#{userbean.deleteAction(o)}" />
</h:column>
```

Ici, la modification consiste à ajouter une action (donc une méthode) qui permet de supprimer un utilisateur du tableau des participants en cliquant soit sur un lien soit sur un bouton.

Ajouter maintenant la méthode `deleteAction()` dans la classe **UserBean** :

```
public void deleteAction(User user) {
    userList.remove(user);
}
```

Redémarrez Tomcat puis observez le rendu de la page :

Formulaire JSF d'inscription à l'[Ecole DORANCO](#)

Prénom:
Nom:
Sexe: ☒ Homme ☐ Femme
Date de naissance: (mm-dd-yy)
Email:
Qualité de service:

Listes des participants inscrits

Prénom	Nom	Email	Date de Naissance	Niveau de Service	Action	Action
Benoît	Leclerc	benoit@doranco.fr	27/10/1977	medium	Supprimer	<input type="button" value="Supprimer"/>
Paul	Andrieux	paul.augustin@doranco.fr	12/06/1965	medium	Supprimer	<input type="button" value="Supprimer"/>
Laura	Treich	laura@doranco.fr	07/10/1987	medium	Supprimer	<input type="button" value="Supprimer"/>
Nathalie	Tango	nathalie@doranco.fr	17/07/1980	premium	Supprimer	<input type="button" value="Supprimer"/>

Observez qu'il n'y a pour l'instant que 4 participants dans la liste affichée.

Saisissez dans le champ "*date naissance*" une date (exemple : 01/01/2020) puis cliquez sur Inscription.

Le rendu de la page est dynamique et on observe que l'ajout du nouvel utilisateur a bien été effectué (donc 5 participants dans la liste) :

Formulaire JSF d'inscription à l'Ecole DORANCO

Prénom:

Nom:

Sexe: ☒ Homme ☐ Femme

Date de naissance: (mm-dd-yy)

Email:

Qualité de service:

Listes des participants inscrits

Prénom	Nom	Email	Date de Naissance	Niveau de Service	Action	Action
Benoît	Leclerc	benoit@doranco.fr	27/10/1977	medium	Supprimer	Supprimer
Paul	Andrieux	paul.augustin@doranco.fr	12/06/1965	medium	Supprimer	Supprimer
Laura	Treich	laura@doranco.fr	07/10/1987	medium	Supprimer	Supprimer
Nathalie	Tango	nathalie@doranco.fr	17/07/1980	premium	Supprimer	Supprimer
Michel	DUPOND	michel.dupont@doranco.fr	01/01/2020	medium	Supprimer	Supprimer

Supprimer maintenant le participant ajouté précédemment en cliquant sur le lien "Supprimer". Testez le bouton "Supprimer" également. Qu'observez-vous ? Pourquoi ?

4. Exercice :

4.1. Ajoutez les champs suivants dans le formulaire :

- **Adresse** (sur une seule ligne : numéro + rue + code postal + ville)
- **Téléphone**
- **Fonction actuelle** : Une liste de fonctions ("Agent de service", "Technicien", "Caissier", "En chômage", "Autre")
- **Disponible** : Propose deux radioBoutons sélection ("Oui" ou "Non") comme pour le champ **sexe**.
- **Langages souhaités** : Propose trois cases à cocher ("Java", "PHP", "C#").

Modifiez la classe **UserBean** pour tenir compte de ces paramètres avec les actions associées.

Adresse étant une entité Java et l'entité **User** contient un paramètre de type **Adresse**, testez la bonne fonctionnalité de l'ajout d'un User à la liste des participants.

- 4.2. Maintenant, on souhaite réaliser sur l'IHM la possibilité d'ajouter plusieurs adresses de type **Adresse** pour un même **User**, grâce au bouton « Ajouter adresse » et qu'à la fin on ajoute le user avec ses adresses grâce toujours au même bouton « Inscription ».

Modifiez le code pour tenir compte de ses changements, soit en déclarant une liste d'adresses (liste d'objets Adresse) comme paramètre dans User, soit en créant un bean managé de Adresse nommé **AdresseBean**.

5. Conclusion :

Dans ce TP, nous avons exploré la bibliothèque de composants fournis par **JSF**, mis en œuvre un certain nombre d'entre eux comme **h:dataTable** pour construire une IHM, nous avons compris la différence entre les composants visuels et non visuels de tag **h** et **f**.

