

TP3 : Gestion des formulaires

Objectifs

Se familiariser avec les API de traitement des formulaires dans un projet J2EE. Ceci passe par :

- A. La création de formulaires comprenant les éléments suivants :
 - La déclaration du formulaire
 - Les champs de saisie de texte et mots de passe, les boutons radio.
 - Les listes et menus déroulants, Les cases à cocher, les zones de saisie de texte libre.
- B. Les boutons d'envoi de formulaire. L'implémentation de Servlets capables de récupérer et de traiter le contenu d'un formulaire comprenant :
 - Des composants à retour simple (*text*, *password*, *radio*, *textarea*, *select*) et des composants à retour multiple (*checkbox*)

1. Introduction

Les formulaires permettent aux développeurs de doter une page web d'éléments interactifs permettant le dialogue avec les internautes. L'utilisateur saisit des données en remplissant des champs, en activant des boutons radio ou des cases à cocher, en sélectionnant des options dans une liste de choix ou en cliquant sur des boutons. Un formulaire peut être soumis en cliquant sur un bouton de type *submit* afin d'envoyer son contenu à un URL capable de le traiter.

2. Création d'un formulaire

2.1. Déclaration d'un formulaire

Les formulaires sont délimités par la balise `<FORM> ... </FORM>` et contiennent obligatoirement l'attribut *action* spécifiant l'URL d'envoi du formulaire et l'attribut *method* désignant la méthode d'envoi du formulaire qui pourrait être soit **GET** ou **POST**. Dans une application J2EE, l'URL d'envoi peut être soit celui d'une servlet, soit celui d'une page *jsp* (*chapitre 3*). Voici un exemple d'un formulaire :

```
<form method="GET" action="ServletTraitement">
.....
</form>
```

2.2. Composants d'un formulaire

La balise **FORM** constitue permet de regrouper des éléments qui permettent à l'utilisateur de choisir ou de saisir les informations à envoyer à l'URL indiqué dans l'attribut **ACTION** de la balise **FORM** par la méthode indiquée par l'attribut **METHOD**. Les éléments à déclarer dans un formulaire devraient avoir un identifiant (attribut *name*) permettant de récupérer leurs valeurs ensuite. Les éléments d'un formulaire peuvent être :

- Champs de saisie de texte : `<input type="text" name="prenom">`

- Zone de texte libre : `<TEXTAREA rows="3" name="commentaire">` votre commentaire `</TEXTAREA>`
- Liste : une liste permet à l'utilisateur de sélectionner un choix parmi ceux qui sont disponibles. La liste (balise *select*) a un identifiant (attribut *name*) et regroupe un ensemble de choix (balise *option*) ayant chacun une valeur prédéfinie (attribut *value*).

```
<SELECT name="niveau">
<OPTION VALUE="3IDSCC"> Troisième année IDSCC </OPTION>
<OPTION VALUE="4IDSCC" selected="selected"> Quatrième année IDSCC </OPTION>
<OPTION VALUE="5IDSCC"> Cinquième année IDSCC </OPTION>
</SELECT>
```

- Boutons radios : il s'agit d'un *bouton* permettant un choix parmi plusieurs proposés (l'ensemble des boutons radios relatifs à la même information devant porter le même attribut *name* pour être mutuellement exclusifs. La paire nom/valeur du bouton radio sélectionné sera envoyée à l'URL de l'action.

```
Homme : <INPUT type="radio" name="sexe" value="M" checked>
Femme : <INPUT type="radio" name="sexe">
```

- Cases à cocher : permettent d'effectuer des choix multiples en envoyant les valeurs des cases cochées.

```
<INPUT type="checkbox" name="loisirs" value="Lecture"> Lecture
<INPUT type="checkbox" name="loisirs" value="sport"> Sport
```

- Bouton d'envoi : est un bouton permettant d'envoyer le contenu du formulaire à l'URL défini dans l'attribut *action* de la balise *form*.

```
<INPUT type="submit" value="Envoyer">
```

2.3. Codage des données

Lors de l'envoi du formulaire les données sont codées selon le type de la méthode d'envoi. Chaque élément du formulaire est codé selon le format **nom=valeur**. Par exemple, si l'utilisateur a saisi dans le champ texte (déclaré auparavant) la chaîne « *Khalid* », alors cet élément sera codé sous la forme **prenom=Khalid**.

Si le formulaire contient plusieurs éléments, leurs résultats seront concaténés et séparés par le caractère **&**

- Méthode **GET** : la méthode *get* envoie le formulaire dans l'URL comme suit :

```
http://urlDeLApplication/action?name1=value1&name2=value2
```

- Méthode **POST** : Le contenu des éléments du formulaire sont envoyées sous un format binaire, cachés dans l'entête de la requête.

3. API de traitement des formulaires

La méthode d'envoi du formulaire spécifie la méthode de la servlet qui sera invoquée. En effet, la méthode **GET** permet d'actionner la méthode **doGet** de la servlet tandis que le type **POST** invoque la méthode **doPost**. Les paramètres du formulaire sont envoyés dans un objet de type **HttpServletRequest**. Celui-ci propose deux méthodes pour la lecture des paramètres envoyés à partir d'un formulaire à savoir :

- **public String getParameter(String name)** : permet de retourner la valeur d'un champ dont on a passé le nom en argument. Cette méthode est utilisée pour récupérer la valeur d'un champ à une seule valeur de retour.
- **public String[] getParameterValues(String name)** : cette méthode est utilisée pour récupérer la ou les valeurs retournées par un champs à retour multiple (checkbox). Le tableau de retour contient les valeurs des champs sélectionnés par l'utilisateur et est égal à **null** si aucun choix n'a été activé.

Remarques :

- Si le champ dont le nom est fourni aux méthodes précédentes n'existe pas, la valeur **null** est retournée.
- Si les paramètres envoyés par un formulaire sont générés automatiquement (formulaire dynamique), la classe **HttpServletRequest** propose la méthode **Enumeration getParameterNames()** permettant de récupérer les noms des paramètres envoyés.

Travail à faire :

1. Créer un nouveau projet intitulé TP3.
2. Créer la page **index.html** contenant un formulaire permettant à l'utilisateur de saisir son nom et son prénom et de les envoyer moyennant la méthode **GET**.
3. Créer la servlet **ServletTraitement** effectuant les traitements suivants :
 - Récupération du nom ainsi que du prénom de l'utilisateur.
 - Génération d'une page dynamique affichant le message suivant : *Bonjour prénom nom*.
4. Quel est le résultat du clic sur le lien hypertexte suivant :
` lien `
5. Quels sont les changements à effectuer si l'on change la méthode d'envoi du formulaire à **POST**.
6. Rajouter deux boutons radio à la page afin de permettre à l'utilisateur de spécifier

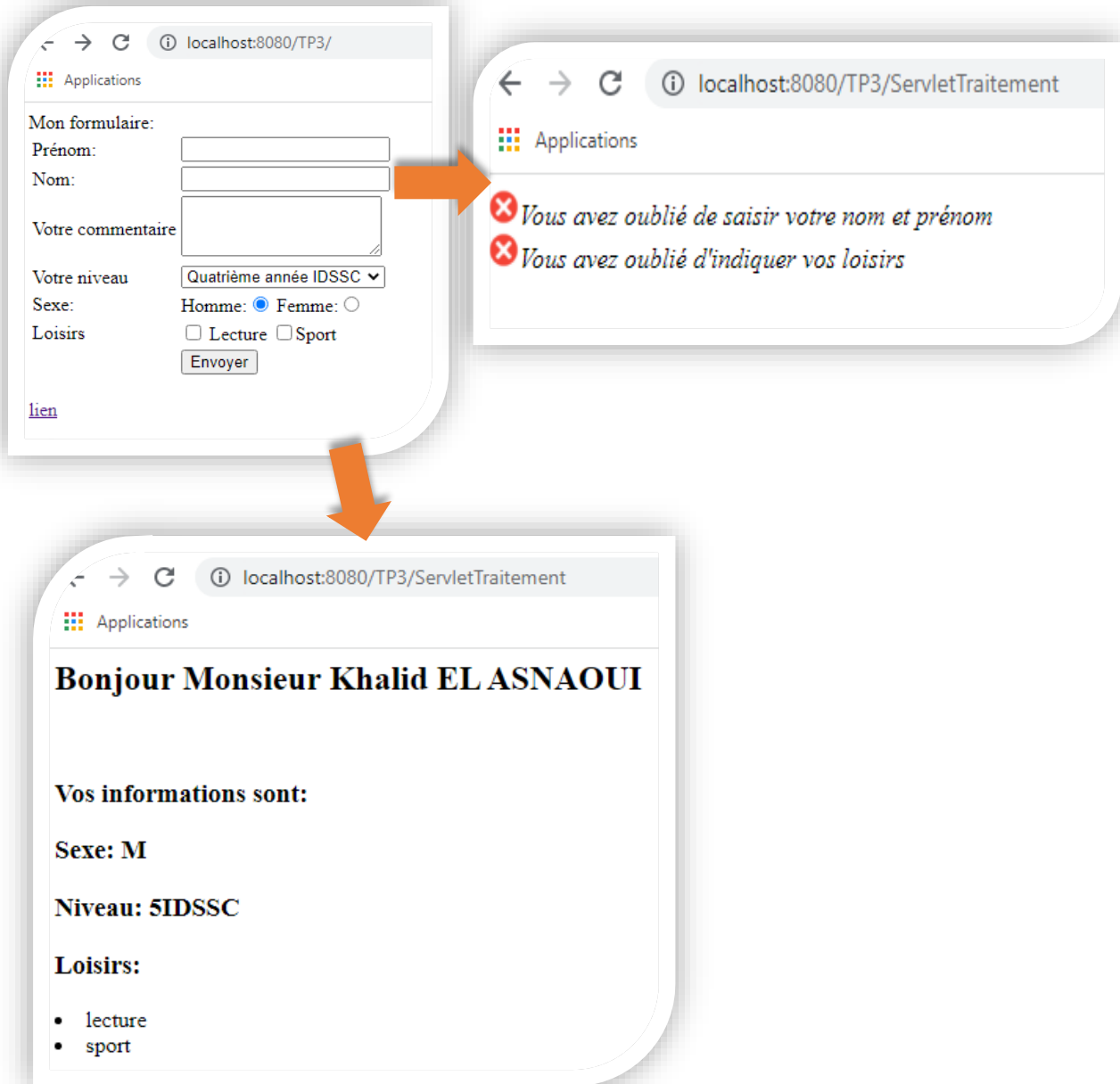
son sexe (utiliser les valeurs H et F pour homme et femme).

7. Modifier le code de la servlet *ServletTraitement* afin d'afficher le message *Bonjour {madame,monsieur} prénom nom* selon la saisie de l'utilisateur.
8. Rajouter à la page (*index.html*) une liste permettant à l'utilisateur de spécifier son niveau parmi (troisième année IDSCC, quatrième et cinquième année.) tout en spécifiant le choix quatrième année IDSCC comme un choix par défaut.
9. Rajouter au formulaire une liste de cases à cocher permettant à l'utilisateur de spécifier ses loisirs.
10. Rajouter le code nécessaire à la servlet afin d'afficher la liste des loisirs de l'utilisateur.

A vérifier / corriger

11. Tester les cas où l'utilisateur ne spécifie pas son nom ou le prénom.
12. Remédier au problème soit :
 - En spécifiant un choix par défaut
 - En traitant ce cas à part dans la servlet
13. Tester les cas où l'utilisateur ne spécifie pas ses loisirs.

Illustration



localhost:8080/TP3/

Applications

Mon formulaire:

Prénom:

Nom:

Votre commentaire:

Votre niveau: Quatrième année IDSSC ▼

Sexe: Homme: ☒ Femme: ☐

Loisirs: ☐ Lecture ☐ Sport

Envoyer

[lien](#)

localhost:8080/TP3/ServletTraitement

Applications

✖ Vous avez oublié de saisir votre nom et prénom

✖ Vous avez oublié d'indiquer vos loisirs

localhost:8080/TP3/ServletTraitement

Applications

Bonjour Monsieur Khalid EL ASNAOUI

Vos informations sont:

Sexe: M

Niveau: 5IDSSC

Loisirs:

- lecture
- sport