

Programmation sur le WEB JSP

Gestion des fiches citoyennes

1 Objectifs :

Le présent TP concerne l'utilisation des pages JSP, ce TP permet de comprendre la génération des pages du côté serveur, son objectif est de connaître le fonctionnement de JSP avec l'utilisation de balises importées comme celles gérant l'accès aux bases de données (jstl, sql), ainsi que l'écriture de ses propres balises.

Dans ce TP, nous passerons des requêtes à une base de données écrite en JAVA (JDBC), à des requêtes formulées par des balises SQL.

2 Sujet du TP :

Pour illustrer l'utilisation des pages JSP, nous allons réaliser un site pour gérer des fiches citoyennes. Le but est d'organiser la relation des citoyens avec sa mairie, en donnant aux citoyens la possibilité d'envoyer des fiches de demande de renseignements qui seront stockées dans leur compte et qui recevront une réponse par les responsables de la mairie.

Une base de données "demandecitoyen" permet de stocker les informations, au travers de cet exemple nous trouvons :

- la gestion des utilisateurs : leur inscription et leur identification.
- Il y aura deux types d'utilisateurs (fonction) : citoyen et administrateur.
- Le citoyen peut créer des fiches (avec ou sans documents joints) et les consulter.
L'administrateur peut apporter une réponse à ces fiches.
- Dans la base "demandecitoyen"
 - La table "personne" gère les utilisateurs (id, nom, prénom, identifiant, mail, fixe, mobile, fonction, rue, ville, motpasse).
 - La table "fiche" gère les fiches (id, demandeur (id personne), objet, description, datedemande, réponse).
 - La table "document" gère les documents (id, nom, propriétaire (id personne)).

Pour vous permettre d'appréhender plus facilement le contexte des pages JSP, vous modifierez certaines pages et vous construirez entièrement les suivantes.

Dans les différents fichiers, les différentes parties à modifier sont indiquées dans des commentaires de ce type :

```
<!--
****  Affichage d'un message indiquant la fin de session, l'identification est
****  obligatoire si le parametre "finsession" est présent
-->
```

3 Gestion des utilisateurs, inscription ou identification : index.jsp, identiteCitoyen.jsp.

3.1 Inscription ou identification des utilisateurs : index.jsp

La gestion de cette page "**index.jsp**" permet de lire les informations nécessaires à la **connexion** ou à l'**inscription**. Elle peut être appelée avec les paramètres "*erreur*", "*finsession*".

Elle affiche un formulaire avec quatre champs de texte dont l'action appelle "*identiteCitoyen.jsp*"

- L'identifiant (obligatoire)
- Le mot de passe (obligatoire)
- Le nom,
- Le mail

Et deux boutons pour "**se connecter**" et "**s'inscrire et se connecter**".

- Pour se connecter, si on a déjà un compte, il suffit de remplir les deux premiers champs de la page index.jsp (identifiant et mot de Passe) et cliquer sur le bouton "se connecter"
- La personne qui n'a pas de compte doit en plus remplir les deux champs suivants (nom et mail), et cliquer sur le bouton "s'inscrire et se connecter" son compte sera créé et sa connexion immédiate.
- Pour les champs nom et identifiant, le minimum de caractère est de 4, et le maximum de 10, faites ceci avec les attributs html 5. Vous pouvez aussi tester la validité du champ mail, l'expression régulière suivante
`"[a-z0-9._%+-]+@[a-z0-9.-]+\.[a-z]{2,4}$"` est souvent utilisée.

Cette page invalide toute session précédente de façon à ne pas avoir de mélange.

- Si le paramètre "*finsession*" est présent, un message est écrit sur la page comme quoi la personne a perdu sa session, elle doit s'identifier une nouvelle fois.
- Si le paramètre "*erreur*" est présent, un message est écrit sur la page comme quoi la personne fait une mauvaise identification, elle doit s'identifier une nouvelle fois.

3.2 Connexion ou inscription : *identiteCitoyen.jsp*

Le bean "*utile.GereCitoyen*" possède quatre propriétés qui sont gérées par cette page : "*nom*", "*identifiant*", "*motPasse*", "*mail*". L'instance de ce bean est créée avec une portée "*session*". **Vous ne devez pas les initialiser vous même, vous devez signaler à la page de s'en charger.**

Ce bean offre comme méthodes utilisées dans cette page :

- *recherchePersonne()* retourne un booléen qui permet de savoir si la personne a déjà un compte.
- *getResult()* qui retourne un "*ResultSet*" sur les caractéristiques de la personne (ensemble des attributs de la table "*personne*").
- *inscrireUtilisateur()* qui inscrit la personne dans la table "*personne*".
- *ouverture (String base)* qui ouvre la connexion à la base, le bean mémorise cette connexion.
- *getConnection ()* qui retourne la connexion à la base mémorisée dans le bean.

- la page "**identiteCitoyen.jsp**" est appelée par "**index.jsp**",
- Si la personne n'est pas enregistrée, (boutons pour "**s'inscrire et se connecter**") c'est une inscription, la page vérifie la présence des champs "*nom*" et "*mail*". S'ils ne sont pas présents:
 - un rappel à la page "*index.jsp*" avec le paramètre "*erreur*" qui informe sur le manque de mail ou de nom corrects.
- Sinon, la personne est inscrite dans la base en tant que citoyen (*inscrireUtilisateur()*).
 - On est alors dans le même cas qu'une personne déjà inscrite qui désire se connecter boutons "**se connecter**".
 - Vérification avec *recherchePersonne()* que l'identifiant et le mot passe correspond bien à ceux enregistrés dans la base de données.
 - Si c'est le cas une session est créée, elle comprend les éléments suivants retirés de la base de données : id, nom, prénom, identifiant, fixe, mobile, mail, rue, ville et fonction (citoyen ou administrateur) de la personne.
 - Si c'est un citoyen un appel à la page "*mesInformationsPersonnelles.jsp*" est fait.

- Sinon c'est un administrateur, un appel à la page "gereDemandeCitoyen.jsp" est fait
 - Si la personne n'est pas trouvée, rappel de la page "index.jsp" avec le paramètre "erreur" qui informe sur l'erreur d'authentification.
- **Protection des mots de passe** : Une fonction de hachage cryptographique avec salage (type Bcrypt) assure la protection des mots de passe. Rappelons que le hachage est irréversible, c'est-à-dire qu'il est impossible de retrouver le texte original à partir du cryptage. Pour vérifier qu'un mot de passe proposé est correct il faut comparer son hachage avec celui mémorisé. Avec le salage, un traitement supplémentaire fait en sorte que le hachage d'un même texte sera différent à chaque fois, dans ce cas la vérification se fait par en utilisant le même salage pour un utilisateur donné. Vous pouvez voir le résultat du hachage dans la base de données. *inscrireUtilisateur()* fait appel à la fonction de cryptage (*BCrypt.hashpw()*) et *recherchePersonne()* utilise une fonction (*BCrypt.checkpw()*) pour comparer le mot passe proposé avec celui crypté.
 - **Pages : mesInformationsPersonnelles.jsp, mesInformationsPersonnellesJSTL.jsp et gestionBaseCitoyen.jsp, gestionBaseCitoyenJSTL.jsp**
 mesInformationsPersonnelles.jsp affiche les caractéristiques d'un utilisateur et permet de les changer, gestionBaseCitoyen.jsp enregistre ces caractéristiques dans la base de données.
 Pour les deux il y a une écriture avec les scriptlets et une écriture avec JSTL, vous pourrez vous en inspirer par la suite.

4 Déposer une fiche citoyenne.

4.1 Sans joindre de document :

Page de création de la fiche : **deposerUneDemande.jsp**

La page doit être faite dans le même modèle que "mesInformationPersonnelles.jsp" avec le même en tête qui comprend :

- Le titre du service : Mon Compte citoyen
- Les 5 boutons :
 - Déposer une demande, Suivre mes demandes, Mon porte documents, Mes informations personnelles, Se déconnecter
- Et la ligne avec le nom du demandeur, son identifiant et la date.

Puis vous y ajoutez une table comprenant un formulaire avec

- un champ input texte "Objet" (max 100 caractères).
- Un champ textArea "Description" (100 colonnes, 15 lignes)
- Et deux boutons "valider" et "abandonner"

Le bouton "abandonner" vous ramène à une page d'accueil, par exemple "mesInformationsPersonnelles.jsp"

Le bouton "valider" appelle la page **gestionBasedemande.jsp** avec le contenu du formulaire, cette page enregistre cette fiche dans la table "fiche", avec une référence au citoyen qui l'écrit, puis appelle "mesInformationsPersonnelles.jsp"

Dans un premier temps, écrivez cette page "**gestionBasedemande.jsp**" en utilisant des scriptlets pour la gestion de la base de données.

4.2 Avec la possibilité de joindre un document

Rajoutez dans le formulaire de la page **deposezUneDemande.jsp** un bouton supplémentaire " validez en ajoutant un document", de nom "ajoutdoc" qui permet de joindre un document (pdf, word, excel, jpg, etc..) à la fiche.

Modifier la page **gestionBasedemande.jsp** pour qu'après avoir enregistré la fiche, elle appelle la page "**uploadPage.jsp**" si le paramètre "ajoutdoc" est présent

- **uploadPage** : permet de connaître le document situé sur l'ordinateur qui doit être chargé, et l'envoyer vers le serveur. Il fait appel ensuite à uploadGestion.

- **uploadGestion** : enregistre le document sur le serveur et appelle **gestionBaseDocument.jsp**. Attention, dans cette page l'adresse du répertoire local où sont stockés les documents doit être précisée:

String savefileDir = "D:\\JSP 2018\\travail 2018\\corrigeCitoyen\\WebContent\\images\\" + saveFichier;

Écrire la page **gestionBaseDocument.jsp** qui rajoute le nom du document dans la table "document" avec une référence du citoyen.

4.3 Avec l'envoi d'un mail à la création d'une fiche.

4.3.1.1 Cas sans document joint

Modifier la page **gestionBasedemande.jsp** pour qu'elle appelle la page "envoiMail.jsp" au lieu de la page "mesInformationsPersonnelles.jsp" après l'enregistrement de la fiche.

La page "**envoiMail.jsp**" envoie un mail à la personne qui crée la fiche. Ce mail précise le numéro de la fiche créée (id), l'objet et la description de la fiche et un texte précisant que sa fiche sera traitée dans un court délai.

Pour cela vous devez avoir l'archive **javax.mail.jar**, à mettre sous WEB-INF/lib. Vous avez à votre disposition au travers du bean GèreCitoyen, une des deux méthodes suivantes:

- La première si vous n'êtes pas obligés de vous identifier (de plus en plus rare), (livebox, freebox, etc..)

```
public static void envoieMail( String objet, String deLaPart, String pour , String contenu)
```

- La deuxième si vous êtes obligés de vous identifier, (campus avec z.imt.fr)

```
public static void envoieMailSecure( String objet, String deLaPart, String pour , String contenu, String motpasse, String host, String port)
```

Avec cette dernière méthode, vous utilisez votre adresse mail et le mot de passe utilisé pour vos mails à l'école, pour avoir le droit d'émettre un mail. Le paramètre *deLaPart* représente votre adresse mail et *motpasse* le mot de passe associé à ce mail. L'envoi d'un mail requiert deux adresses mails, celui de l'expéditeur et celui qui le reçoit. Pour pouvoir envoyer un mail par le biais d'un opérateur (gmail.com, imt.fr), celui-ci demande le mot de passe.

Pour cela, si la page ne connaît pas les valeurs (adresse mail, mot de passe, host et port) de l'expéditeur, elle envoie un formulaire pour vous les demander, vous conservez ces valeurs dans des variables de session pour ne pas avoir à les redemander par la suite.

Écrivez l'envoi du formulaire dans "envoiMail".

4.3.1.2 Cas avec document joint

C'est la page "**gestionBaseDocument.jsp**" qui va appeler la page "envoiMail" au lieu de la page "**gestionBaseDemande.jsp**".

La différence avec le cas précédent, c'est que le contenu du mail contient en plus le nom du document qui a été transmis.

Modifier "**gestionBaseDocument.jsp**" qu'elle fasse cet appel avec les informations nécessaires, et modifier "envoiMail" pour qu'il mette dans le mail, le nom du document.

4.3.1.3 Envoi d'une signature dans le mail

Rajoutez un paramètre signature à "*envoiMailSecure*", et réalisez l'appel avec signature dans "envoiMail"

4.4 Utilisation des balises sql pour le traitement des requêtes à la base de données

Sauvegarder cette page dans "**gestionBasedemandeScriptlet.jsp**" et réécrivez "**gestionBasedemande.jsp**" en utilisant uniquement des balises.

Vous devez déclarer ces balises par

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
```

Par la suite, vous pouvez écrire les pages suivantes uniquement par ces balises.

4.5 Afficher la liste des fiches envoyées par un Citoyen

Page d'affichage de la liste de fiches : **suivreMesDemandes.jsp**

- La page reprend le même modèle que "**mesInformationPersonnelles.jsp**" et "**deposerUneDemande.jsp**" avec le même en tête.

- Puis vous y ajoutez une ligne par fiche, cette ligne comprend : le numéro de la fiche (id), l'objet de la fiche, la date de création de la fiche. Pour que ce soit plus lisible, alternez entre deux couleurs sur les lignes.
- Avec le numéro de la fiche vous mettez un lien vers la page d'affichage du contenu de la fiche : **ficheCitoyenne.jsp**, vous pouvez le faire sous forme de bouton.

4.6 Afficher une fiche.

Page d'affichage de la fiche : **ficheCitoyenne.jsp**

Avec toujours le même en-tête.

Vous y mettez le contenu d'une fiche : date de création, objet, description et réponse


4.7 Afficher la liste des documents que vous avez téléchargés.

Page de la liste de document : **monPorteDocuments.jsp**

Avec toujours le même en-tête.

Donner la liste, des documents que la personne a téléchargés, un document par ligne.

Sur cette ligne, vous mettez d'abord un icône donnant le type de document, par exemple pour le

pdf : (pdf.jpg) , pour une image une vignette de l'image (petite image), inutile de le faire pour tous les documents possibles.

Puis le nom du fichier de ce document, associé à un lien html sur le fichier, de façon qu'en cliquant dessus on puisse lire le document.

4.8 Gestion des fiches par l'administration.

4.8.1 Liste des fiches

Pour accéder à ces pages il faut avoir la fonction " administrateur " au lieu de " citoyen " par défaut. On ne vous demande pas de gérer l'inscription d'un administrateur, mais dans les personnes présentes dans la base, il y en a un son identifiant est " admin " et son mot de passe est " admin ". Vous pouvez en créer d'autres avec " phpmyadmin " par exemple, mais ce n'est pas nécessaire.

Vous vous connecter avec ces valeurs et vous devez voir la page **gereDemandeCitoyen.jsp**, si elle existe et si vous avez bien écrit **"IdentiteCitoyen.jsp"**.

Page d'affichage de la page d'administration : **gereDemandeCitoyen.jsp**

L'en tête n'est plus la même car il s'agit de celle de l'administration.

Vous pouvez mettre une ligne avec le titre "Gestion des fiches Citoyennes", un bouton pour se déconnecter.

Une autre ligne avec le nom de la personne qui traite, et la date courante.

Vous pouvez mettre cette en tête dans "accesmenuFicheAdministration.jspf", car il resservira par la suite.

Puis

- une liste des fiches qui n'ont pas été traitées (champ réponse inexistant),
- une liste des fiches traitées.

Dans ces deux listes vous mettez :

- une ligne par fiche, cette ligne comprend : le numéro de la fiche (id), l'objet de la fiche, la date de création de la fiche. Pour que ce soit plus lisible, alternez entre deux couleurs sur les lignes.
- Avec le numéro de la fiche vous mettez un lien vers la page **reponseFicheCitoyenne.jsp**, avec le numéro de fiche en paramètre, vous pouvez le faire sous forme de bouton.

4.8.2 Contenu de la fiche et gestion de la réponse

Page d'affichage de la page d'administration : **reponseFicheCitoyenne.jsp**

Cette page reprend le même en-tête que **gereDemandeCitoyen.jsp**,

Sur une ligne vous mettez le numéro de la fiche et son objet.

Puis sa description

Puis un " TEXTAREA " pour écrire la réponse.

Enfin deux boutons, un pour "validez" et l'autre pour "abandonner".

Le bouton "validez" appelle la page "gereBaseReponse" qui écrit la réponse dans la table fiche de la base de données.

Le bouton "abandonner" appelle la page `gereDemandeCitoyen.jsp`

4.8.3 Écriture de la réponse dans la base de données

Page d'affichage de la page d'administration : `gereBaseReponse.jsp`

Cette page écrit, si le paramètre "valider" est présent réponse dans la table fiche de la base de données.

Puis elle envoie un mail à la personne qui a envoyé cette fiche pour la prévenir que la réponse a été faite, et aussi lui donner cette réponse.

Puis elle appelle la page "`gereDemandeCitoyen.jsp`".

4.9 Contrôle d'accès aux pages

Nous n'avons pas défini de principe d'accès global aux différentes pages. Si l'on exclut les pages d'accès au site `index.jsp`, et `identiteCitoyen.jsp`, il est important d'interdire tout accès aux autres pages sans s'être identifié. Nous allons utiliser pour cela la notion de filtres comme pour les servlets.

Dans le fichier `web.xml` de l'application, nous définissons :

```
<filter>
  <filter-name>portier</filter-name>
  <filter-class>utile.FiltreEntree</filter-class>
</filter>

<filter-mapping>
  <filter-name>portier</filter-name>
  <url-pattern>/citoyen/*</url-pattern>
</filter-mapping>
```

qui détermine que le filtre s'exécutera sur toutes les pages `jsp (*)` qui se trouveront dans le répertoire "citoyen" de notre application.

Vous devez donc sortir `index.jsp`, et `identiteCitoyen.jsp` de ce répertoire, vous pouvez les mettre directement sous `WebContent`, ou dans un autre répertoire. Faites attention au changement de contexte que cela génère.

Nous allons créer un filtre qui bloquera l'accès à ces pages en cas de non identification.

Un exemple de filtre :

```
package utile;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class FiltreEntree implements Filter {
    private FilterConfig filterConfig = null;

    public void init(FilterConfig filterConfig) throws ServletException {
        this.filterConfig = filterConfig;
    }

    public void doFilter(ServletRequest request, ServletResponse response,
        FilterChain chain) throws IOException, ServletException {
        HttpServletRequest hrequest = (HttpServletRequest) request;

        // si les attributs de session nom et identifiant sont absents
        // on dirige l'appel vers index.jsp avec si nécessaire un paramètre

        chain.doFilter(request, response);
    }

    public void destroy() {
        this.filterConfig = null;
    }
}
```

Redéfinir votre application avec ce filtre.

4.10 Enregistrement unique d'une fiche ou d'une réponse

Dans cette application, si vous validez une fiche puis que vous revenez sur la page précédente du navigateur et que vous la validez à nouveau, la fiche est enregistrée deux fois.

Régalez ce problème

4.11 Écriture d'une balise formulaire

L'écriture d'un formulaire est chaque fois similaire, vous allez écrire une balise "formulaire" en utilisant la technique des tags files.

Elle pourrait par exemple être la suivante

```
tags:formulaire couleur="#c0ebe7" action="identiteCitoyen.jsp" methode="get" taille="30" nom="text" prenom="text"
adresse="text" profession="text" telephone="text" courriel="text" soumettre="submit" />
```

Les 5 premiers attributs sont obligatoires, les autres en fonction du formulaire que vous écrivez.

Cette balise ci-dessus donne la présentation suivante



Écrivez ce tag, et réécrivez la page index.jsp avec cette balise, qui donnera :



Vous avez, joint avec ce texte, un support de cours qui vous donne une piste, avec des attributs dynamiques (non définis dans la balise). Attention, vous devez trouver une astuce pour garder l'ordre des attributs dans le même que celui de l'utilisation de la balise.

4.12 Possibilité de plusieurs descriptions ou réponses

Dans l'état actuel, le Citoyen ne peut pas réagir pour sur sa fiche en fonction de la réponse, l'administration peut modifier ou compléter sa réponses.

Modifier l'application (table de la base et page JSP) pour permettre d'avoir une espèce de forum entre le citoyen et l'administration.

5 La base de données "demandecitoyen"

Pour pouvoir utiliser une base de données vous devez :

- mettre le driver "Mysql" dans le répertoire "**lib**" de **WEB-INF** de l'application,
- mettre la définition des "**taglib**" utilisés en JSP pour accéder aux balises "sql" pour la base de données et les balise jstl (*dbtags.jar, standard.jar, jstl.jar*), dans ce même répertoire "**lib**"

Ces éléments sont dans les données propres au TP qui vous sont accessibles.

Ce TP utilise une base de données, ce sera une base **Mysql**, le code serait le même pour une autre base à part le choix du driver.

La base s'appelle "demandecitoyen", elle contient trois tables : personne, fiche et document.

Vous trouverez le fichier "demandecitoyen.sql" dans les données de votre TP, il permet d'importer ces tables.

Pour la connexion à la base, suivant que vous utiliser des scriplets ou des balises vous avez deux connexions possibles :

Scriptlet : ouvrebase2.jsp avec le taglib :

```
<%@ taglib uri="http://jakarta.apache.org/taglibs/dbtags" prefix="sql"%>
```

```
<sql:connection id="conn1">
  <sql:url>jdbc:mysql://localhost:3306/demandecitoyen?user=root&useEncoding=true&characterEncoding=UTF-8 </sql:url>
  <sql:driver>com.mysql.jdbc.Driver</sql:driver>
</sql:connection>
```

Balise pour la gestion de la base avec les taglib

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
```

Le fichier context.xml dans META-INF

```
<Context>
  <Resource name="DonneesBase"
    auth="Container"
    type="javax.sql.DataSource"
    username="root"
    password=""
    driverClassName="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost:3306/demandecitoyen"
    validationQuery="select 1"
    maxActive="10"
    maxIdle="2"/>
</Context>
```

6 Compte rendu

Vous devez rendre l'ensemble des classes qui constituent ce TP (le répertoire application WEB), dans une archive

nom1ETnom2JSP.zip

nom1 et nom2 sont les deux noms d'un binôme.

Le placer sous l'espace habituel réservé à la réception des TP

et obligatoirement l'envoyer par mail à l'adresse ogor.robert@orange.fr

Vous pouvez commenter votre code !

Date limite **lundi 18 février 2019**

7 Pages de référence.

<http://docs.oracle.com/javaee/6/tutorial/doc/bnafd.html>

http://fr.wikipedia.org/wiki/JavaServer_Pages

http://lfe.developpez.com/Java/TomCat/?page=page_2

<http://docs.oracle.com/javaee/1.4/api/>

<http://docs.oracle.com/javase/8/docs/api/>

JSTL

<http://www.developer.com/java/ejb/article.php/1447551/An-Introduction-to-JSP-Standard-Template-Library-JSTL.htm>

<http://adiguba.developpez.com/tutoriels/j2ee/jsp/taglib/>

<http://adiguba.developpez.com/tutoriels/j2ee/jsp/jstl/>

http://www.tutorialspoint.com/jsp/jsp_standard_tag_library.htm

<https://docs.oracle.com/javaee/5/tutorial/doc/bnama.html>