2ème Année Informatique
Groupe C

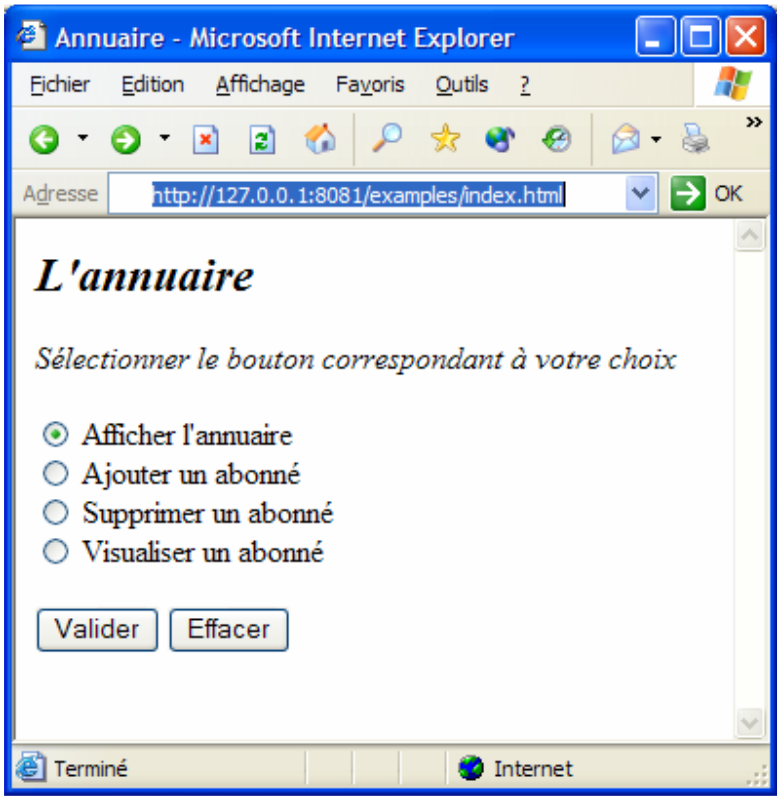
PROJET RESEAUX :
ANNUAIRE

❶ Introduction au projet

- Le but du projet est de créer un **annuaire** en **servlet** et qui sera donc accessible à partir d'un navigateur Web.

- Cet annuaire contiendra le **nom** de l'abonné, son **adresse** et son **numéro de téléphone**.

- L'utilisateur y ayant accès pourra **afficher** l'annuaire, lui **ajouter** ou **supprimer** un abonné et **rechercher** un abonné en spécifiant les critères de son choix.



Le choix des actions à la disposition de l'utilisateur de l'annuaire

❷ Détails de programmation

- L'annuaire est stocké dans un fichier texte. Chaque ligne y représente un abonné et se compose ainsi
  `Nom_de_l'abonné:Adresse:Numéro de téléphone`

  L'accès aux différents champs d'une ligne se fera donc à l'aide d'un `StringTokenizer` et prendra donc comme paramètre le séparateur « **:** » ainsi que la chaîne à analyser.

- L'accès en lecture à l'annuaire se fera ainsi :

```
BufferedReader leFluxFic=new BufferedReader(new FileReader(chemin));
```

  `chemin` étant une chaîne de caractères contenant le chemin complet du fichier annuaire)
  On utilisera ensuite le méthode `readLine()` pour lire chaque ligne du fichier

- L'accès en écriture se fera de la manière suivante :

```
BufferedWriter leFluxFicW=new BufferedWriter(new FileWriter(chemin,bool));
```

  Si `bool` est à `true` alors l'écriture se fera à partir de la fin du fichier. Si `bool` est à `false`, le contenu existant du fichier sera alors effacé.

  On utilisera la méthode `write(String chaine)` pour écrire `chaine` dans le fichier.

❸ Organisation – Description – Gestion des erreurs

La servlet se compose d'une classe (`UnChoix`) héritant de `HttpServlet` dont la méthode `doGet` a été surchargée.

Rôle des différentes méthodes de la classe :

`public void doGet(HttpServletRequest telleRequete,HttpServletResponse telleReponse)`

C'est la méthode qui est exécutée lorsque l'utilisateur a effectué un choix sur la page d'accueil. Cette méthode a pour rôle de lancer la fonction correspondant au choix de l'utilisateur.

Gestion des erreurs :
La méthode vérifie si le choix de l'utilisateur est correct et si le fichier annuaire existe.

`public void affiche(HttpServletRequest telleRequete,PrintWriter leFluxSortie)`

Cette méthode affiche le contenu de l'annuaire.

Gestion des erreurs :
La méthode vérifie si le fichier annuaire existe.

`public void ajouter(HttpServletRequest telleRequete,PrintWriter leFluxSortie)`

Si aucun paramètre supplémentaire n'est passé dans l'URL, la méthode affiche le formulaire permettant l'ajout d'un nouvel abonné.

Si les paramètres `nom`, `adresse` et `num` sont renseignés, la méthode effectue alors l'ajout de l'abonné dans l'annuaire.

Gestion des erreurs :
La méthode vérifie si le fichier annuaire existe, si tous les champs du formulaire d'ajout sont renseignés, si l'utilisateur n'a pas entré de caractères interdits et si l'abonné n'est pas déjà existant dans l'annuaire.

`public void suppression(HttpServletRequest telleRequete,PrintWriter leFluxSortie)`

Si aucun paramètre supplémentaire n'est passé dans l'URL, la méthode affiche l'annuaire avec un lien « supprimer » pour chaque abonné permettant de le supprimer.

Si le paramètre `ligneasupprimer` ou `all` sont renseignés, la méthode s'occupe alors de supprimer l'abonné correspondant de l'annuaire, et de ré-afficher celui-ci.

Gestion des erreurs :
La méthode vérifie si le fichier annuaire existe.

```
public void visualise(HttpServletRequest telleRequete,PrintWriter leFluxSortie)
```

Si aucun paramètre supplémentaire n'est passé dans l'URL, la méthode affiche un formulaire permettant la recherche d'abonnés.

Si au moins un des paramètres `nom`, `adresse` ou `num` est renseigné, alors la méthode effectue la recherche et affiche le(s) abonné(s) trouvé(s).

Gestion des erreurs :
La méthode vérifie si le fichier annuaire existe, et si le formulaire de recherche a bien au moins un champs de rempli.

```
public void html_header(PrintWriter leFluxSortie,String title,boolean grosTitre)
```

Cette méthode écrit une entête HTML standard.

```
public void html_footer(PrintWriter leFluxSortie)
```

Cette méthode écrit un pied de page HTML standard.

```
public boolean ligne_existe(String telleLigneSearched)
```

Cette méthode vérifie si la ligne passée en paramètre existe dans le fichier.

❹ Remarques

Problèmes :

On a vu que l'annuaire est stocké dans un fichier texte, que l'on appelle aussi fichier plat. Cette méthode de stockage pose différents problèmes :

- La manipulation d'un tel fichier est d'autant ralentie que sa taille augmente.
- La recherche dans un tel fichier peut se révéler difficile. En effet, pour rechercher quelque chose, il faut vérifier individuellement chaque enregistrement.
- Dans une implémentation réelle (c'est-à-dire avec un nombre de visiteurs assez important), les accès simultanés au fichier peuvent être à l'origine de ralentissement, car un seul utilisateur pourra modifier le fichier à un moment précis.
- Certaines manipulations peuvent se révéler lourde et coûteuse en mémoire, comme l'insertion d'un enregistrement au milieu du fichier du fait de l'accès séquentiel. En effet, il faut d'abord stocker tout le contenu du fichier en mémoire, effectuer le traitement, et réécrire l'intégralité du fichier (Voir la méthode de suppression de l'annuaire)

Solution :

Dans une implémentation réelle, l'utilisation de fichiers plats à accès séquentiel est donc à proscrire du fait de ses inconvénients vu au dessus. Les SGBD (bases de données) apportent des solutions à ces problèmes :

- L'accès aux données est rapide
- Les recherches par critère sont faciles
- Le problème des accès simultanés est entièrement géré par la base de données et n'implique pas de modification de la conception du programme

❺ Code Source

Voir feuilles suivantes …