Jacques VINCENSINI
Conseil Formation Développement
Linux, Java, XML

Editeur, analyseur de fichiers JAVA

Le but de ce TP est de réaliser un éditeur en JAVA capable d'éditer des fichiers sources (.java) et d'en analyser la structure pour la restituer graphiquement.

Vous devez rendre le tp sous forme d'un fichier ZIP dont le nom est votre nom de famille.
Ex ; Vincensini.zip.
Ce fichier est à rendre pour le 3 mars 2006 à Minuit dernier délai. Aucun délai ne sera accordé.

Ce fichier doit contenir

- ✓ Toutes les sources java de votre projet (Attention aux paquetages…) Les paquetages doivent figurer dans le zip en chemins relatifs. Chaque source porte votre nom dans l'entête.
- ✓ La documentation au format doc. Faire figurer votre nom dans l'entête de chaque page. Elle comprend les schémas UML de conception (diagrammes de classes pour le modèle métier), les choix d'implémentation, l'état de votre développement et toutes autres informations que vous jugez utiles.
- ✓ La documentation au format JAVADOC (Attention aux chemins relatifs).
- ✓ Je n'ai pas besoin des .class.

Attention l'installation, la recompilation et l'exécution de l'application compteront dans l'évaluation. Il faut que tout se déroule le plus facilement possible. En résumé la mise en œuvre de votre livraison (zip) doit être immédiate. Je ne dois pas chercher, déplacer des fichiers pour lancer votre application.

Conseils :
Pensez à faire une interface la plus ergonomique possible. Des points seront donnés à la présentation, la robustesse du code, la clarté des messages d'erreur. L'application ne doit pas se « planter » et je ne devrai pas voir une seule « NullPointerException ».
Pensez à faire un paquetage pour les classes d'IHM et un autre pour vos classe métier.

L'INTERFACE GRAPHIQUE :

Elle doit comporter au moins une barre de menus, une barre d'état et une zone d'édition avec la gestion complète du MDI.

LES PLUS :

- Une barre d'outils
- Des onglets

L'EDITION :

- le logiciel doit gérer l'ouverture (et fermeture) simultanée de plusieurs fichiers JAVA.
- L'analyse ne se fait que sur les fichiers ouverts. Il n'y a pas de notion de projet.
- Le mode lecture est obligatoire.

Types de fichiers java reconnus au minimum;

- Un fichier de classe standard avec des attributs d'instance (quelque soit son modificateur de visibilité) et des méthodes d'instance. On souhaite voir apparaître au minimum, pour chaque classe : son nom, l'ensemble de ses attributs (nom, type et modificateurs de visibilité), l'ensemble de ses méthodes (nom, type du code retour, nombre de paramètres) et ses constructeurs (nombre de paramètres).
- Vous ne devez pas prendre en compte les classes mal écrites, 2 classes dans le même fichier. Par contre un message signalant que vous avez vu le problème est nécessaire.
- Les paquetages ne sont pas pris en compte (attention aux messages)
- Les choix d'affichage pour les classes sont à votre convenance.
- En cas d'erreur de compilation d'une classe vous ne devez pas modifier l'arborescence déjà existante.

LES PLUS

- Gérer aussi dans les méthodes les noms et types des attributs
- Gérer les classes imbriquées
- Gestion des interfaces
- Gestion de l'écriture
- Gestion des paquetages

LA SAUVEGARDE :

- Elle doit être automatique à intervalles réguliers pour TOUS les fichiers ouverts **même si vous ne faites que la lecture**.
- Elle doit pouvoir être commandée manuellement et individuellement (Fichier qui a le focus)

LES PLUS :

- Offrir à l'utilisateur la possibilité de configurer les intervalles de sauvegarde et de compilation.
- Implémenter un mécanisme qui ne sauvegarde que les fichiers qui ont été modifiés.

L'IMPRESSION :

- Elle concerne les fichiers JAVA ouverts. Elle doit être conviviale et offrir toutes les possibilités modernes d'impression (choix des pages à imprimer, nombre de pages…). Sa commande est manuelle.

LA COMPILATION :

- Elle doit être automatique à intervalles réguliers pour TOUS les fichiers ouverts même si vous ne faites que la lecture.
- Elle doit pouvoir être commandée manuellement et individuellement (Fichier qui a le focus)
- Pour les classes qui offrent des erreurs de compilation, les messages d'erreur de compilation doivent apparaître dans la barre d'états de l'application.

LES PLUS :

- Offrir à l'utilisateur la possibilité de configurer les intervalles de sauvegarde et de compilation.
- Implémenter un mécanisme qui ne compile que les fichiers qui ont été modifiés.
- Quand l'utilisateur clique sur une erreur dans la barre d'état, le code correspondant est sélectionné dans l'éditeur.

LA GESTION DES ERREURS :

- Elle doit être la plus complète et explicite possible.

NOTE TRES IMPORTANTE:

Comme il s'agit d'un cahier des charges, il ne peut être complet. Il vous est donc demandé de fournir par écrit dans un fichier au format DOC toutes les justifications à vos choix d'implémentation.
Toutefois, pour toute question importante vous **pouvez et devez** m'envoyer un email à :
contact@jvconsultant.fr pour me poser toutes vos questions aux qu'elles je répondrai volontiers.