TP 1 - Prise en main de KToolbar et du Mobility Pack

Youssef RIDENE

Exercice 1 : KToolbar

- Executer KToolbar à partir de son répertoire d'installation
- Créer un projet simple
- Explorer les paramètres du projet
- Reprendre la MIDlet vue en cours et l'ajouter à votre projet
- Compiler et executer le projet
- Essayer les différents émulateurs intégrés
- Afficher un texte dans la console lors du lancement de l'application

Exercice 2 : Netbeans

0.1 Créer un projet Java ME

- File -> New Project -> Java ME -> Mobile Application
  1. Donnez un nom à votre projet
  2. Décochez «Create Hello MIDlet»
  3. Choisissez l'émulateur à utiliser, le profil et la configuration de votre projet
  4. Valider
- Click droit sur le projet dans la fenêtre gauche de Netbeans
- Choisir « new MIDlet »
- Entrez le nom de votre MIDlet et Validez
- Lancer la compilation de votre projet
- Explorer les répertoires de votre projet
- Explorer le contenu du Jar et du Jad générés
- Lancer le simulateur
- Afficher sur la console un texte lors du démarrage de l'application

0.2 Manipulation du Jad

- Ajouter une icône de votre choix à votre application
- Dans les proprietés du projet, modifier des attributs dans le jad, rajouter les votres. Génrer les binaires et explorer à nouveau le fichier Jad
- Récupérer dynamiquement et afficher sur la console les valeurs de trois attributs

0.3 Créer des configurations dans le projet

Les configurations sous Netbeans sont une réponse simple et basique à la problématique de fragmentation abordée en cours. Le but de l'exercice est de l'appliquer sur un exemple pratique. Supposons que vous avez une application qui doit utiliser la fonction vibreur du téléphone à un instant donné. Voici les contraintes liée à cette fonctionnalité :

- En MIDP10, il n y a pas de méthode pour utiliser le vibreur
- En MIDP20, il existe une fonction pour utiliser le vibreur

Donc, pour gérer ça dans notre application, il suffit, une fois qu'on a bien définie les 2 configurations : MIDP10 et MIDP20, d'utiliser le préprocesseur intégré de Netbeans. Cela donne :

Listing 1 – Preprocess.java

```
//#if MIDP10
    Je ne peux pas utiliser le vibreur
//#elif MIDP20
    J'utilise la fonction vibrate(...) de la classe MIDlet
//#endif
```

Les //# avant les directives de pré compilation indiquent à Netbeans que nous souhaitons utiliser le préprocesseur. Lors du choix de la configuration à compiler, Netbeans commentera et décommentera le code suivant vos choix.

- Afficher un texte différent pour chaque configuration créée
- Compiler et executer les différentes configurations
- Est-il possible de faire autrement qu'en utilisant le préprocesseur ?

0.4 Obfuscation

Netbeans intègre un obfuscateur assez puissant qui vous permet de choisir le niveau d'obfuscation de vos classes. Sur cette page : http ://proguard.sourceforge.net/ vous trouverez des informations sur l'obfuscateur les plus utilisés.

- Ajouter des classes et des méthodes inutiles dans votre projet
- Changer les niveaux d'obfuscation et explorer le contenu et la taille du fichier jar généré
- Créer une classe Util.java, cette classe contient :
  - Un tableau de 100 int
  - Un constructeur qui affiche les valeurs du tableau. Afin de provoquer une exception, on affichera les valeurs de 0 à 100 inclus.

1. Désactiver l'obfuscation et essayer votre programme
2. Activer l'obfuscation et reessayer. Que remarqez-vous ?