Mode d'emploi du PDA ou comment ajouter une nouvelle application au PDA

1- L'application que vous voulez ajouter

Votre application sera composée d'au moins 3 classes, qui correspondent au découpage du pattern MVC :

- package data (HelloDatas) : le modèle c'est-à-dire le coeur de votre application.
- package view (HelloView) : implémente l'interface graphique
- package control (HelloCtrl) : son constructeur crée une instance du modèle et une instance de la vue. Cette classe doit implémenter l'interface IApplication, et éventuellement d'autres écouteurs de Swing.

1.1 le package

L'application doit être définie sera répartie dans les packages du « pda ».Vous pouvez aussi créer des sous-packages pour regrouper les classes de votre projet.

1.2 Icône de votre application

Il faut choisir (ou fabriquer) une petite image qui dénotera votre application sur le PDA et la plcar dans le dossier approprié : les icônes se trouvent dans data/img/

2- La classe LaunchDatas.java

Quand on lance la version de base du PDA, deux applications de base apparaissent :

- launcher : qui lance les applications
- hello : une petite application pour montrer comment s'ajoute une application.

Les informations de base :

Les informations nécessaires pour connecter une application dans le PDA :

- un nom qui s'affichera dans le PDA
- le nom de la classe compilée qui sera le lanceur de votre application réalisé en implémentant l'interface `IApplication` (voir plus bas)
- une icône qui sera associée au nom

Ajout des informations de base :

Les applications sont ajoutées au PDA via la classe data`/LaunchDatas.java`

Cette classe déclare un tableau initialisé avec les noms des applications :

```
      private static final String[] appArray = { "hello" };
Il faut ajouter le nom de son application dans le tableau appArray.

et deux Hashtable (applications et icons):
/** The list of application classes that can be launched */
    private Hashtable<String, Class> applications ;

    /** The list of icons associated to applications */
```

```
    private Hashtable<String, String> icons ;
```

Dans le constructeur de la classe `LaunchData.java` les deux Hashtable sont créées, puis on ajoute une application en donnant :

```
   // son nom
      appArray[0] = "hello";  // 0 c'est la 1ère application
  // son code compilé avec comme clé le nom
      applications.put("hello", HelloCtrl.class);
  // un icône avec comme comme clé le nom
      icons.put(appArray[0], "hello.png");
```

Donc si on ajoute une nouvelle application on doit refaire ses trois lignes de code avec le nom de l'application, le nom de la classe compilée de départ et l'icône.

`LaunchData` doit importer toutes les classes lanceurs des applications donc elles sont importées avec l'import en tête du fichier:

```
import pda.control.*;
```

3- L'implémentation de IApplication (cas de la classe HelloCtrl)

La partie contrôleur de l'application **`HelloCtrl`** doit implémenter l'interface `pda/control/IApplication.`
C'est cette classe qui sera lié au Panel de l'interface graphique.

1- Le code de HelloCtrl commence par importer les classes nécessaires

```
import pda.datas.*;
import pda.view.*;

import javax.swing.*;
import java.awt.event.*;

public class HelloCrtl implements Iapplication, ActionListener {
```

2- Les variables d'instance

```
Une variable qui contiendra le nom de l'application
/** the name of the application */
    private String name;

On peut avoir besoin aussi d'une variable pour l'instance du modèle et d'une
variable pour l'instance de la vue.

  /** the view of the application */
    private HelloView view;

    /** the engine of the application */
    private HelloDatas engine;
```

3 – Les méthodes

Le constructeur crée les instances de modèle et vue :

```
public HelloWorld() {
     engine = new HelloDatas();
     view = new HelloView(engine);

/*
 *  Start the application in the specified PDA
 */
 public void start(PdaCtrl pda) {
     System.out.println ( "Start of Hello application") ;
 }

 /**
  *  Return the application name as String
  */
 public String getAppliName() {
      return name;
 }

 /**
  *  Return the panel that contains all the application display
  */
 public JPanel getAppliPanel() {
   return view.getPanel();
 }

 /**
  *  Close the application and its display panel
  * @param false if the application denied to commit a suicide
  */
 public boolean close() {
   return true;
 }

/**
 * Give a unique name to the application
 * @param theName the name of the application
 */
public void setAppliName ( String theName ) {
   this.name=theName ;
}
```

4- les fichiers de données :

Si vous avez des fichiers de données, les localiser dans le répertoire .data/ (il y a un lien dans ww)

4 - Lancement des applications

On utilisera l'utilitaire ant et le fichier build.xml associé qui définit les tâches (compilation,

exécution, création d'archive etc..)
Par exemple pour lancer l'archive standard :
Se placer dans ww.
Commencer par compiler : ant compile (si vous avez ajouté des choses)
Ensuite on lance le pda avec : ant runjar