SWING :
principes de base

Introduction

Swing est la 2nd bibliothèque de classes, après AWT, qui permet de créer une Interface Utilisateur et de la gérer. La procédure de construction d'une interface Swing est similaire à celle d'AWT : créer un cadre, des composants dans ce cadre, une mise en page pour ces composants, des méthodes en réponse aux actions de l'utilisateur. Il n'est pas possible de faire le tour de Swing en un seul chapitre, c'est pourquoi ce chapitre s'intéresse seulement aux principes de base et à la construction d'une UI simple.

Différences entre Swing et AWT

Du côté de l'utilisateur, la différence est majeurs : l'apparence des composants est totalement différente (bien que celle-ci soit paramétrable). Du côté du concepteur, Swing présente plus de composants qu'AWT, ces composants commencent par un J (ex : JButton, JLabel…), la gestion des évènements est complètement différente. Les *layouts*, eux, restent les mêmes.

Composants

Comme précédemment dit, les composants sont presque les mêmes que ceux d'AWT. Pour une description des composants et plus d'informations, se reporter au chapitre sur AWT.

- JButton
- JLabel
- JTextArea
- JTextField
- JRadioButton
- JCheckBox
- JPasswordField
- JComboBox
- JList
- JScrollBar

D'autres composants avancés seront traités dans les chapitres suivants concernant Swing.

Conteneurs

Les conteneurs sont les objets dans lesquels sont incorporés les composants de l'UI. Un cadre (JFrame) est un conteneur. On peut citer : JFrame, JDialog (c'est un peu différent…), JPanel, JScrollPane, JTabbedPane, JSplitPane…

- JFrame, JPanel : classiques, une JFrame est un JPanel dans un cadre spécifique à l'environnement (le cadre contenant par exemple la X pour fermer la fenêtre)
- JDialog : boîte de dialogue qui surgit en plein milieu de l'écran (les utilisateurs de Windows connaissent sûrement celle de type Warning qui contient les habituelles Fatal Errors)
- JScrollPane : barres de défilement. Ajouter un JScrollPane à votre JFrame, ajoutez ensuite un JTextArea (ou autre composant nécessitant des barres de défilement) l'intérieur de ce JScrollPane.
- JtabbedPane : cadre à onglets

Gestion d'évènements

Les gestionnaires d'événement permettent d'intercepter les actions des utilisateurs et d'assigner au programme un comportement adapté en réponse. Il existe de nombreuses manières de gérer les évènements. Ce chapitre propose une manière classique, c'est celle qui est utilisée par défaut par JBuilder. Elle se met en œuvre en 3 temps : création d'un « écouteur » d'actions qui attend et capte les actions de l'utilisateur, d'une méthode qui met en œuvre la réponse adaptée, de la classe qui définie cette méthode.

```
// 1°
// assigne un écouteur d'évènements basique à un composant nommé composantX
composantX.addActionListener(new java.awt.event.ActionListener() {
public void actionPerformed(ActionEvent e) {
// quand une action sera captée, on devra exécuter la méthode ci-dessous, implémentée dans
// la partie 2°
composantX_actionPerformed(e);
}
});

// 2°
void jButton1_actionPerformed(ActionEvent e) {
// code à insérer
  }
}

// 3°
class nomDeLaClasse implements java.awt.event.ActionListener {
NomDeLaClasseOuSeTrouveLeComposant adaptee;
nomDeLaClasse(NomDeLaClasseOuSeTrouveLeComposant adaptee) {
this.adaptee = adaptee;
}
public void actionPerformed(ActionEvent e) {
adaptee.jButton1_actionPerformed(e);
}
}
```

Le code présenté ci-dessus est exactement celui crée par JBuilder, d'autres manières de capter les actions de l'utilisateur et d'appeler un code en réponse seront étudiées plus tard. Si vous possédez JBuilder, il suffit de double-cliquer sur un composant en mode Conception pour que soit automatiquement généré le code ci-dessus.

Construction d'une UI

Cette partie a pour fonction de créer une application à l'aide de Swing.

1) Comment sera le programme ? / Que fera t-il ?

Le programme sera une petite fenêtre avec un bouton et un label. Au clic sur le bouton, le texte « Hello World again ! » s'affichera dans le label. Pour cela, l'application sera composée de 2 classes : une définissant l'interface, une autre contenant la méthode main() et appelant l'autre classe.

2) Interface graphique

1) Créer la première classe en la définissant comme un cadre
2) Ajouter les composants
3) Créer le gestionnaire d'événement pour le bouton

1)

```
public class Cadre1 extends JFrame { // définission du cadre
public Cadre1() { // méthode constructeur
}
}
```

2)

```
jLabel1.setText(""); // création du label
jButton1.setText("Cliquez ICI !"); // création du bouton
this.getContentPane().setLayout(flowLayout1); // mise en place d'un layout
this.getContentPane().add(jLabel1, null); // ajout du label du cadre
this.getContentPane().add(jButton1, null); // ajout du bouton au cadre
```

3)

```
// 1° partie: "quand une action est effectuée sur le bouton, appeler la méthode
// JButton1_actionPerformed()"
jButton1.addActionListener(new java.awt.event.ActionListener() {
public void actionPerformed(ActionEvent e) {
jButton1_actionPerformed(e);
}
});
```

```
// 2° partie: création de la méthode jButton1_actionPerformed()
void jButton1_actionPerformed(ActionEvent e) {
jLabel1.setText("Hello World again !"); // afficher ce texte dans le label
}
```

3) Classe principale

```
public class Application {
//Construire l'application
public Application() {
Cadre1 frame = new Cadre1();
}
//Méthode principale
public static void main(String[] args) {
new Application();
}
```