UFR IM²AG

UNIVERSITÉ Grenoble Alpes

INF362 - Seconde interrogation
Année 2015-2016

Durée : 1h.
Tout document interdit. Calculatrices interdites.
Pour chaque question, une partie des points peut tenir compte de la présentation.
Le barême est indicatif.

1 Questions de cours (4 points)

1. Expliquez comment fonctionne une application Swing en distinguant les parties contenues dans Swing de celles écrites par l'utilisateur.
2. Pourquoi n'est-il en général pas possible de réaliser une animation en ne modifiant que la méthode `paintComponent` ?

2 Machine à sous (16 points)

Dans cette partie, vous devez écrire une application qui simule le comportement d'un machine à sous simple. Voici la spécification de l'application en question :

- l'application devra comporter 3 zones de texte. Chacune de ces zones contiendra un chiffre (initialement déterminé aléatoirement) et devra être accompagnée d'un bouton `Stop`
- l'application devra contenir un bouton `Demarrer`. Lors d'une pression sur le bouton démarrer, l'application passera en mode `Défilement` : les chiffres des zones de texte doivent alors changer aléatoirement toutes les 100 millisecondes.
- en mode `Défilement`, une pression sur le bouton `Stop` accompagnant un chiffre provoquera l'arrêt du changement aléatoire pour ce chiffre. Une fois le dernier chiffre stoppé, l'application devra afficher un message indiquant si le joueur a gagné ou non (le joueur gagne si les trois chiffres stoppés sont identiques).

L'image ci-contre donne une idée de l'aspect de l'interface qu'il vous est demandé de réaliser. Pour la génération de nombres aléatoires, la classe `Random` s'utilise en deux temps : le constructeur permet de fixer la graine du générateur, soit choisie par le système (constructeur sans argument), soit donnée explicitement en argument. Ensuite, la méthode `nextInt` renvoie une valeur pseudo aléatoire comprise entre 0 et l'argument entier (non compris) qui lui est donné. Vous pourrez vous inspirer du code suivant pour démarrer :

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

class MachineASous implements Runnable {
    public void run() {
        JFrame frame = new JFrame();

        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.pack();
        frame.setVisible(true);
    }

    public static void main(String [] args) {
        SwingUtilities.invokeLater(new MachineASous());
    }
}
```

Choses Utiles

```
Interface ActionListener (dans java.awt.event.*) :
    void actionPerformed(ActionEvent e)

Classe JPanel (dans javax.swing.*) :
    Component add(Component comp);

Classe JTextField (dans javax.swing.*) :
    static int RIGHT_ALIGNMENT;
    void setHorizontalAlignment(int alignment);
    void setEditable(boolean b);
    void setText(String t);

Classe JFrame (dans javax.swing.*) :
    static int EXIT_ON_CLOSE;
    JFrame(String title);
    void add(Component comp, Object constraints);
    void pack();
    void setVisible(boolean b);
    void setDefaultCloseOperation(int operation);

Classe JButton (dans javax.swing.*) :
    JButton(String text)
    void addActionListener(ActionListener l)

Classe Random (dans java.util.*) :
    int nextInt(int n);

Classe SwingUtilities (dans javax.swing.*) :
    static void invokeLater(Runnable doRun);

Classe Timer (dans javax.swing.*) :
    Timer(int delay, ActionListener listener);
    void start();
    void stop();

Classe BorderLayout (dans java.awt.*) : attributs statiques NORTH, SOUTH, EAST, WEST
```