INFO-H-200 : Programmation orientée objet
TP 5 - Interfaces graphiques (Swing) et Threads

Professeur : Hugues Bersini
http://cs.ulb.ac.be/public/teaching/infoh200

Année académique 2014-2015

Exercice 5.1 : Hello World

Ecrire l'application *HelloWorld* présentée dans le rappel théorique en utilisant des instances des classes `JFrame` et `JLabel`.

Exercice 5.2 : Gestion d'événements

Concevoir une interface graphique comprenant un `JTextArea` et deux `JButton` : un bouton *test* et un bouton *clean*. Cliquer sur le bouton *test* met le texte de la `JTextArea` à *test réussi* et cliquer sur le bouton *clean* réinitialise la `JTextArea`. Utiliser des classes séparées pour gérer les événements des `JButton`.

Exercice 5.3 : Timers

Implémentez la classe `MyTimer` présentée dans le rappel et testez là avec plusieurs valeurs.

Exercice 5.4 : Dessin

Le but de cet exercice est de concevoir une application de dessin minimaliste.

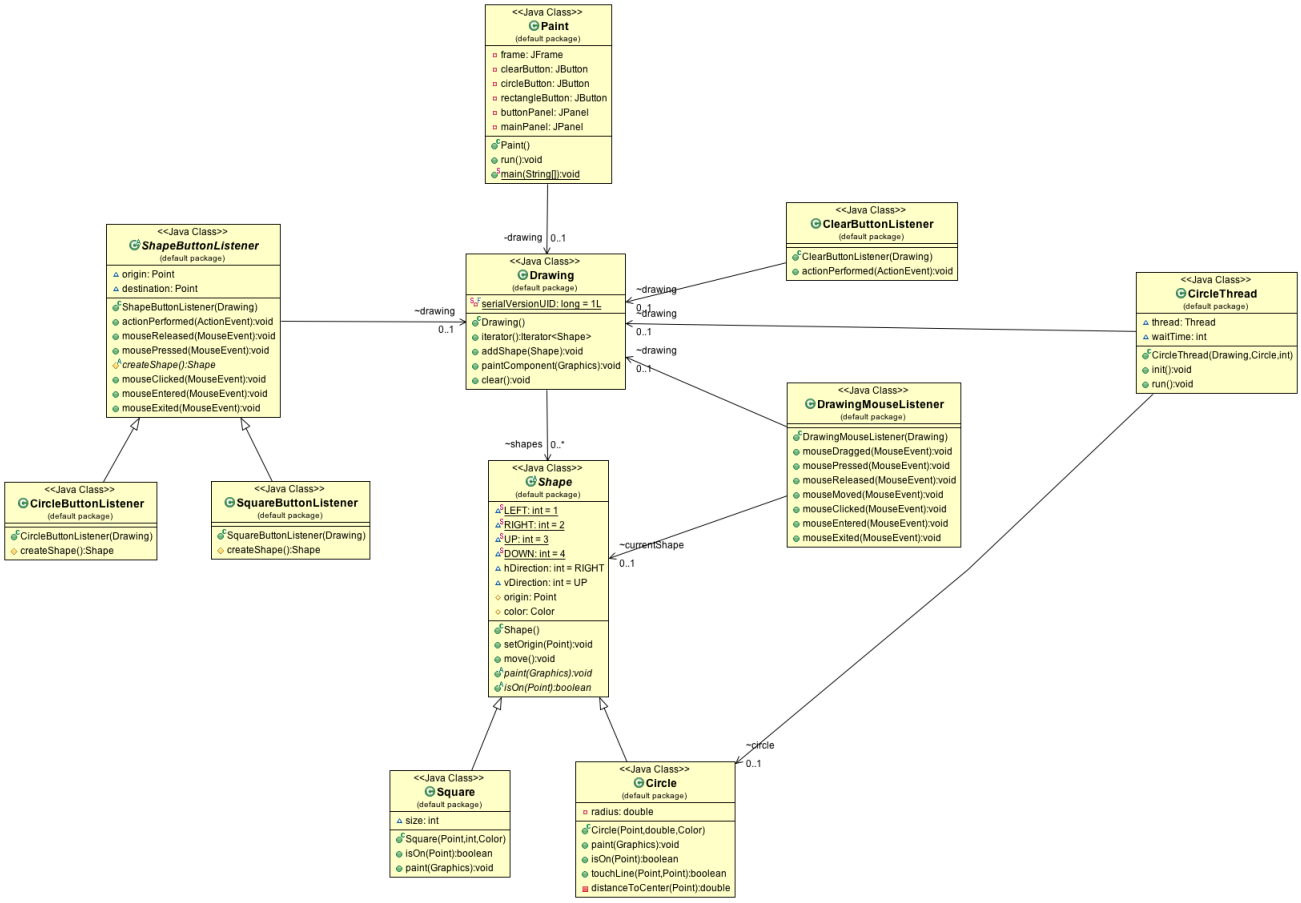
1. Analyser le code fourni sur la page web des TPs et essayez le.
2. Plutôt que de dessiner des petits cercles, dessiner des courbes. Une courbe peut être vue comme un ensemble de segments reliant des points. Utiliser la documentation de `Graphics` pour apprendre comment dessiner.
3. Dessiner uniquement quand le bouton de la souris est enfoncée. Aide : utiliser un `MouseListener` pour récupérer l'état de la souris.
4. Effacer le dessin lorsque le bouton est cliqué.
5. **Bonus** Redimensionnez la fenêtre. Vous remarquerez que votre dessin disparaît. Stockez les courbes comme une liste de points. Créer une sous-classe de JPanel (à utiliser pour dessiner) et redéfinir sa méthode `paintComponent(Graphics g)` qui redessine tout l'espace lors d'un redimensionnement ou lors d'un appel à la méthode `repaint()`.

Exercice 5.5 (Bonus) : Editeur multi-fenêtre

Concevoir une petite application permettant d'éditer du texte. L'application devra permettre :
— d'éditer un texte,
— d'ouvrir et éditer un fichier texte,
— de sauver le texte dans un fichier (utiliser un `JFileChooser`),

Exercice 5.6 : Animation (GUI et Threads)

Analyser le code fourni sur la page web des TPs et testez le. Le code étant un peu plus conséquent que les précédents, aidez vous du diagramme UML suivant pour parcourir le code source. Modifiez ensuite le programme de manière à ce que les rectangles se déplacent aussi mais uniquement verticalement.



Exercice supplémentaire : Dessin avancé

Reprendre l'exercice 5.4 et ajouter des JButton pour choisir la couleur de peinture, mais également la grosseur du pinceau, ajouter des formes (carrés, cercles, ...), etc.

INFO-H-200 - Programmation orientée objet
TP 5

Corrections

Exercice 5.1

Voir dans les slides

Exercice 5.2

Fichier GUI.java

```
import javax.swing.*;

public class GUI {
  private JFrame frame;
  private JPanel panel;
  private JButton clearButton;
  private JButton testButton;
  private JTextArea textArea;

  public GUI(){
    this.initializeControls();
  }

  private void initializeControls(){
    frame = new JFrame("Exercice2");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    panel = new JPanel();
    frame.getContentPane().add(panel);
    textArea = new JTextArea(3,20); //3 lignes, 20 colonnes
    testButton = new JButton("Test");
    testButton.addActionListener(new TestActionListener(textArea));
    clearButton = new JButton("Effacer");
    clearButton.addActionListener(new ClearActionListener(textArea));
    panel.add(textArea);
    panel.add(testButton);
    panel.add(clearButton);
    frame.pack();
    frame.setVisible(true);
  }

  public static void main(String[] args){
    GUI g = new GUI();
  }
}
```

Fichier ClearActionListener.java

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

public class ClearActionListener implements ActionListener {

  JTextArea textArea;

  public ClearActionListener(JTextArea textArea){
    this.textArea = textArea;
  }

  @Override
  public void actionPerformed(ActionEvent arg0) {
    textArea.setText("");
  }
}
```

Fichier TestActionListener.java

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

public class TestActionListener implements ActionListener {

  JTextArea textArea;

  public TestActionListener(JTextArea textArea){
    this.textArea = textArea;
  }

  @Override
  public void actionPerformed(ActionEvent arg0) {
    textArea.setText("test");
  }
}
```

Exercice 5.3

```
// Fichier MyTimer.java
public class MyTimer implements Runnable {

  String string;
  int waitTime;
  Thread thread;
```

```
  public MyTimer(String string, int waitTime){
    this.string = string;
    this.waitTime = waitTime;
    this.thread = new Thread(this);
    thread.start();
  }

  public void run() {
    // TODO Auto-generated method stub
    try{
      while(true){
        System.out.println(string);
        Thread.sleep(waitTime);
      }
    } catch (Exception e) {};
  }

  public static void main(String[] args) {
    MyTimer first = new MyTimer("first",2000);
    MyTimer second = new MyTimer("second",3000);
  }
}
```

Exercice 5.4 et Exercices supplémentaires

Voir fichiers fournis !