TP java
Graphisme et timer

Timer

La classe javax.swing.Timer remplace avantageusement l'utilisation directe de thread . Elle génère un évènement à intervalles réguliers (à la milli-seconde). C'est le "event-dispatching thread" qui exécute l'évènement, donc on peut directement modifier des objets graphiques.

Méthodes importantes :
le constructeur de javax.swing.Timer :

```
public Timer (int delay, ActionListener listener);
/** delay = l'intervalle de temps entre chaque évènement.
 * listener = l'objetécouteur de ces evènements.
 */
```

Il faut donc implémenter l'interface ActionListener dans la classe d'écoute et définier la méthode actionPerformed().

méthodes permettant de lancer le processus :

```
public void start();
```

méthode permettant de stopper le processus :

```
public void stop();
```

Note :
Cette méthode est à utiliser uniquement si le traitement à faire est court en temps d'exécution sinon on aura une interface non réactive pendant le traitement ....

Manip : Ecrire un composant graphique qui réalise un affichage de l'heure rafraichi toutes les secondes.

Remarque : débrouillez vous pour trouver une classe permettant de récupérer l'heure du système.

Graphisme

Il y a principalement deux cas où un programmeur est amené à dessiner : la réalisation d'un composant graphique non-standard tel qu'un diagramme, une minuterie, …, ou la création automatique d'une image à partir de données (synoptique de surveillance entre autres).

Dans les 2 cas, le dessin sera assuré par la classe java.awt.Graphics2D. Vous ne devez en aucun cas essayer de construire une instance de cette classe. Pour dessiner dans un composant graphique, l'instance vous est donnée en paramètre de la méthode paint(Graphics g) pour AWT ou paintComponent() pour SWING. Pour le dessin dans une image, on crée un objet java.awt.Image, puis on utilise la méthode getGraphics() pour récupérer l'objet Graphics et dessiner dans l'image.

```
Image img = new BufferedImage(150,52,BufferedImage.TYPE_INT_RGB);
Graphics2D g2d = (Graphics2D)img.getGraphics();
```

Puis les méthode de Graphics2D feront le reste (drawLine(), …
Pour manipuler des fichier image on passe par la classe toolkit :
//Voici un exemple

```
public void paint(Graphics g)
{
theImage = Toolkit.getDefaultToolkit().createImage( url ou fichier);
if (theImage != null)
{
g.drawImage(theImage, 0, 0, 0);
}
}
```

Souvent on modifie la méthode paint() ou la réécrit (pas d'appel de la super méthode).

A propos des transformation :
La méthode rotate() attend un angle en radian, le Graphics2D garde mémoire de ses transformations pour pouvoir les combiner.

Manip :

Manip Ecrire un objet Chrono graphique avec un cadran d'horloge et une aiguille des secondes qui tourne.

```
public void paint(Graphics G){


            g2d.clearRect(0,0,200,200);      // on efface
            g2d.rotate(angle,100.0f,100.0f);
            g2d.drawString("HelloWord",110,90);
            g2d.drawLine(100,100,150, 100);

            G.drawImage(img, 50,50, this);



            G.drawLine(10,10,20, 30);

      }
```