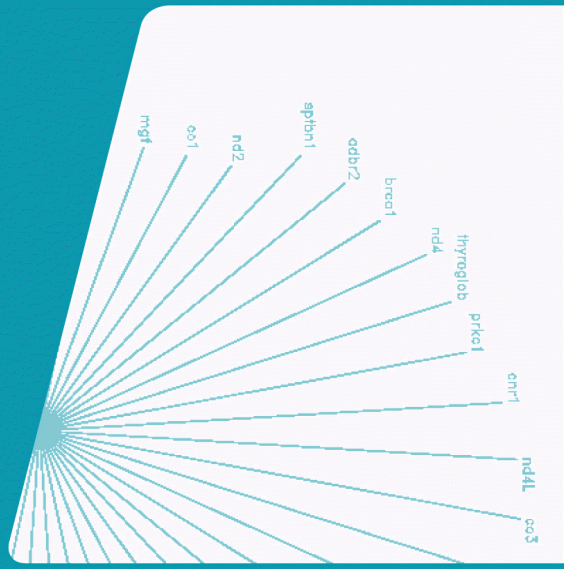
INSTITUT NATIONAL DES SCIENCES APPLIQUÉES DE LYON

FORMATION




# Cours 5 – Graphisme

membre de
UNIVERSITÉ DE LYON

INSA LYON

# PLAN

1. Graphisme
2. Coordonnées & Dimensions des écrans
3. Classes utiles : LinkedList
4. Animations et souris
5. Fluidité des animations
6. Timer
7. Gestion de la transparence
8. Graphisme 2D avancé
9. Exemples d’applications graphiques

# Taille des ecrans

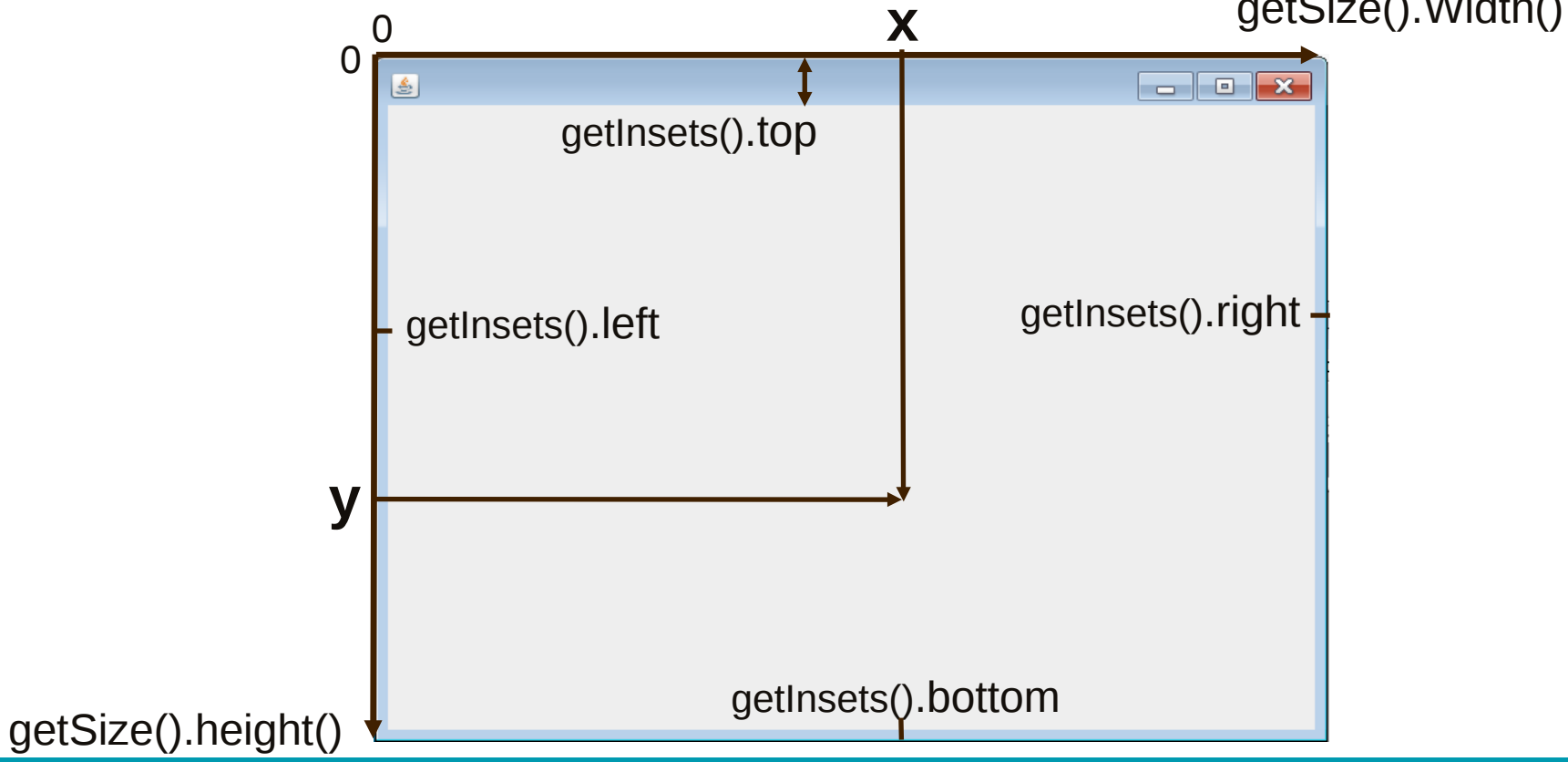
**Ecrans.java**

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

public class Ecrans {

public static void main(String[] args)   {
    // Lecture des variables d'environnement
    GraphicsEnvironment env=GraphicsEnvironment.getLocalGraphicsEnvironment();
    GraphicsDevice screen = env.getDefaultScreenDevice();
    GraphicsConfiguration config = screen.getDefaultConfiguration();
    System.out.println("Taille Ecran physique  : " + config.getBounds());
    Rectangle bounds = env.getMaximumWindowBounds();
    System.out.println("Taille Ecran Visible   : " + bounds );
    // creation d'une fenetre vide
    JFrame frame = new JFrame("Frame Info");
    System.out.println("Allocation d'une fenetre non définie et non visible");
    System.out.println("Fenetre taille         : " + frame.getSize() );
    frame.setSize(400, 200);
    frame.setVisible( true );
    // Fenetre definie en taille 400x200 et visible
    System.out.println("Fenetre définie de taille 400x200  et visible");
    System.out.println("Fenetre taille physique: " + frame.getSize() );
    System.out.println("Positions bordures     : " + frame.getInsets() );
    System.out.println("Fenetre taille visible : " +
                       frame.getContentPane().getSize() );
    System.exit(0);
    }
}
```

# Tailles Ecrans, Frames

```
Administrateur : C:\Windows\system32\cmd.exe
c:\travail\info\Java\Graphisme\COurs7 Java New 2014\ex0-Ecrans>javac *.java

c:\travail\info\Java\Graphisme\COurs7 Java New 2014\ex0-Ecrans>java Ecrans
Taille Ecran physique   : java.awt.Rectangle[x=0,y=0,width=1920,height=1080]
Taille Ecran Visible    : java.awt.Rectangle[x=0,y=0,width=1920,height=1032]
Allocation d'une fenetre non d0finie en taille et non visible
Fenetre taille          : java.awt.Dimension[width=0,height=0]
Fenetre d0finie de taille 400x200 par setSize et visible
Fenetre taille physique: java.awt.Dimension[width=400,height=200]
Positions bordures      : java.awt.Insets[top=36,left=9,bottom=9,right=9]
Fenetre taille visible  : java.awt.Dimension[width=382,height=155]

c:\travail\info\Java\Graph
```

```
Administrateur : C:\Windows\system32\cmd.exe
c:\travail\info\Java\Graphisme\COurs7 Java New 2014\ex0-Ecrans>javac *.java

c:\travail\info\Java\Graphisme\COurs7 Java New 2014\ex0-Ecrans>java Ecrans
Taille Ecran physique  : java.awt.Rectangle[x=0,y=0,width=1920,height=1080]
Taille Ecran Visible   : java.awt.Rectangle[x=0,y=0,width=1920,height=1032]
Allocation d'une fenetre non d0finie en taille et non visible
Fenetre taille         : java.awt.Dimension[width=0,height=0]
Fenetre d0finie de taille 400x200 par setSize et visible
Fenetre taille physique: java.awt.Dimension[width=400,height=200]
Positions bordures     : java.awt.Insets[top=36,left=9,bottom=9,right=9]
Fenetre taille visible : java.awt.Dimension[width=382,height=155]

c:\travail\info\Java\Graphisme\COurs7 Java New 2014\ex0-Ecrans>_
```



# Dessiner

## Void paint (Graphics g)

- Tous les composants graphiques possèdent une méthode public

void paint(Graphics g)

- Cette méthode est appelée automatiquement par le système de gestion graphique a chaque fois que le composant doit être dessinée à l'écran après une modification ou une occultation.

- C'est dans cette méthode que le composant définie le traitement pour réaliser son affichage

- Si on veut modifier  dessiner ses propres composants, il faut surcharger (override) la méthode  paint() et utiliser l'instance g de type Graphics  pour faire les opérations de dessin

# Classe Graphics

## Opérations graphiques élémentaires

- **clearRect**(int x, int y, int width, int height)
- **create**(int x, int y, int width, int height)
- dispose()
- **drawArc**(int x, int y, int width, int height, int startAngle, int arcAngle)
- **drawImage**(Image img, int x, int y, ImageObserver observer)
- **drawLine**(int x1, int y1, int x2, int y2)
- **drawOval** ou **fillOval**(int x, int y, int width, int height)
- **drawPolygon** ou **fillPolygon**(int[] xPoints, int[] yPoints, int nPoints)
- **drawRect** ou **fillRect**(int x, int y, int width, int height)
- **drawRoundRect** ou **fillRoundRect**(int x, int y, int width, int height, int arcWidth, int arcHeight)
- **drawString**(String str, int x, int y)
- **getColor**() **setColor**(Color c)
- **getFont**() **setFont**(Font font)

# Exemple 1 :Dessiner dans un JFrame

**paintJFrame.java**

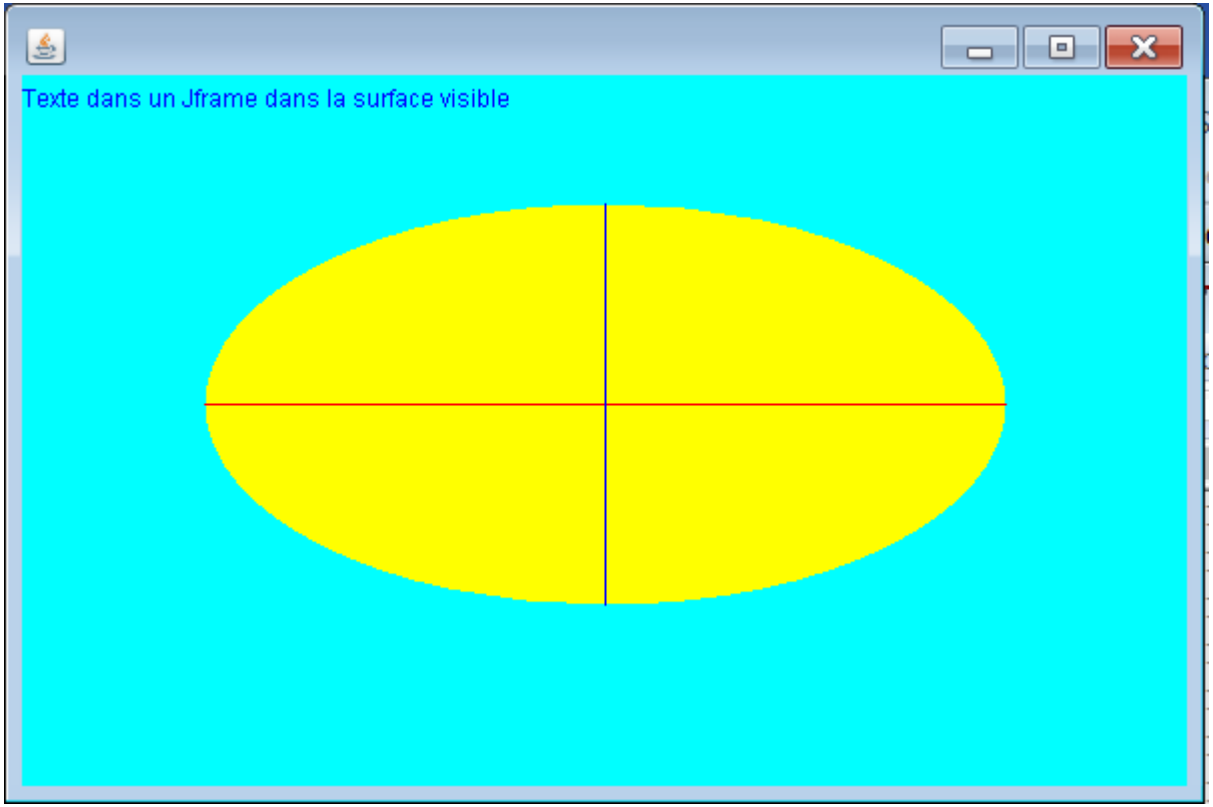
```
import java.awt.*;
import javax.swing.*;

public class paintJFrame extends JFrame {

  // Tout mettre dans le constructeur et définir la taille de la fenetre
   public  paintJFrame(int w, int h)  {
   setSize(w,h);
   setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   setVisible(true);
   }

// Paint() est appelé à chaque fois que l'OS pense devoir redessiner la fenetre
// après un resize, une occultation par une autre fenêtre ou bien
// demandé par l'utilisateur avec un appel à la méthode repaint()

public void paint(Graphics g) {
 // Mettre ICI le code qui appel aux fonctions graphiques par le parametre g


}

public static void main(String[] args) {
   // définit la taille de la fenêtre dynamiquement dans le constructeur
    paintJFrame Mafenetre = new paintJFrame(600,400);
   // il y a bien 1 seule ligne de code dans le main !!!
  }
}
```

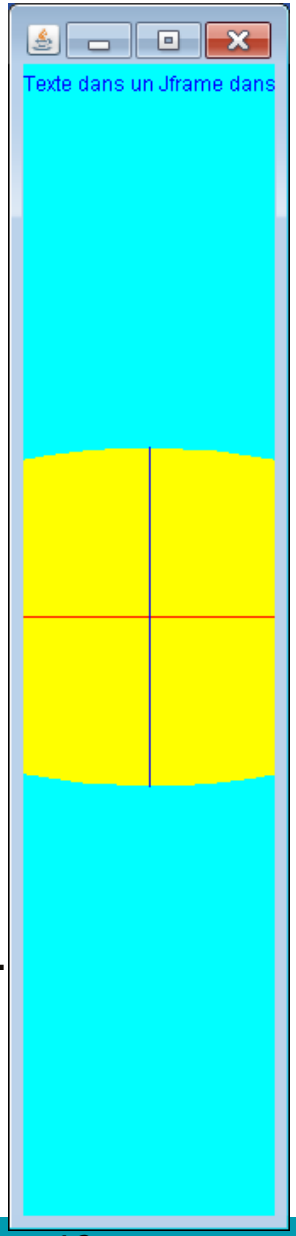
# Exemple 1 :Dessiner dans un JFrame

**paintJFrame.java**

```
public void paint(Graphics g) {
        // Le centre de la fenetre physique
          int Xc=getSize().width/2;
           int Yc=getSize().height/2;
           // Mettre le fond à Cyan
          g.setColor( Color.cyan );
          // Remplir le fond de cyan  Attention setColor définit
          // aussi bien le crayon (pen) que la couleur de l'arrière plan  !
          g.fillRect(0,0,getSize().width,getSize().height);
          // crayon va etre en jaune
          g.setColor( Color.yellow );
          // dessine une ellipse au centre de la fenetre
          //et circonscrit dans un rectangle de taille 400 200
          g.fillOval( Xc-200, Yc-100, 400, 200 );
          // dessine une ligne sur la largeur de l'ellipse au milieu
          g.setColor( Color.red);
          g.drawLine(Xc-200,Yc,Xc+200,Yc);
          // dessine une ligne sur la hauteur de l'ellipse au milieu
          g.setColor( Color.blue);
          g.drawLine(Xc,Yc-100,Xc,Yc+100);
           // Ecrire in texte en bleu en dehors et dans la surface visible
           g.setColor(Color.blue);
           g.drawString("Texte dans un Jframe hors de la surface visible", 0,0);
          Font font=g.getFont();
          int HauteurTexte=getFontMetrics(font).getHeight();
          g.drawString("Texte dans un Jframe dans la surface visible",
                     getInsets().left, getInsets().top + HauteurTexte);
}
```

# Exemple 1 :Dessiner dans un JFrame




Mon premier dessin  dans un JFrame

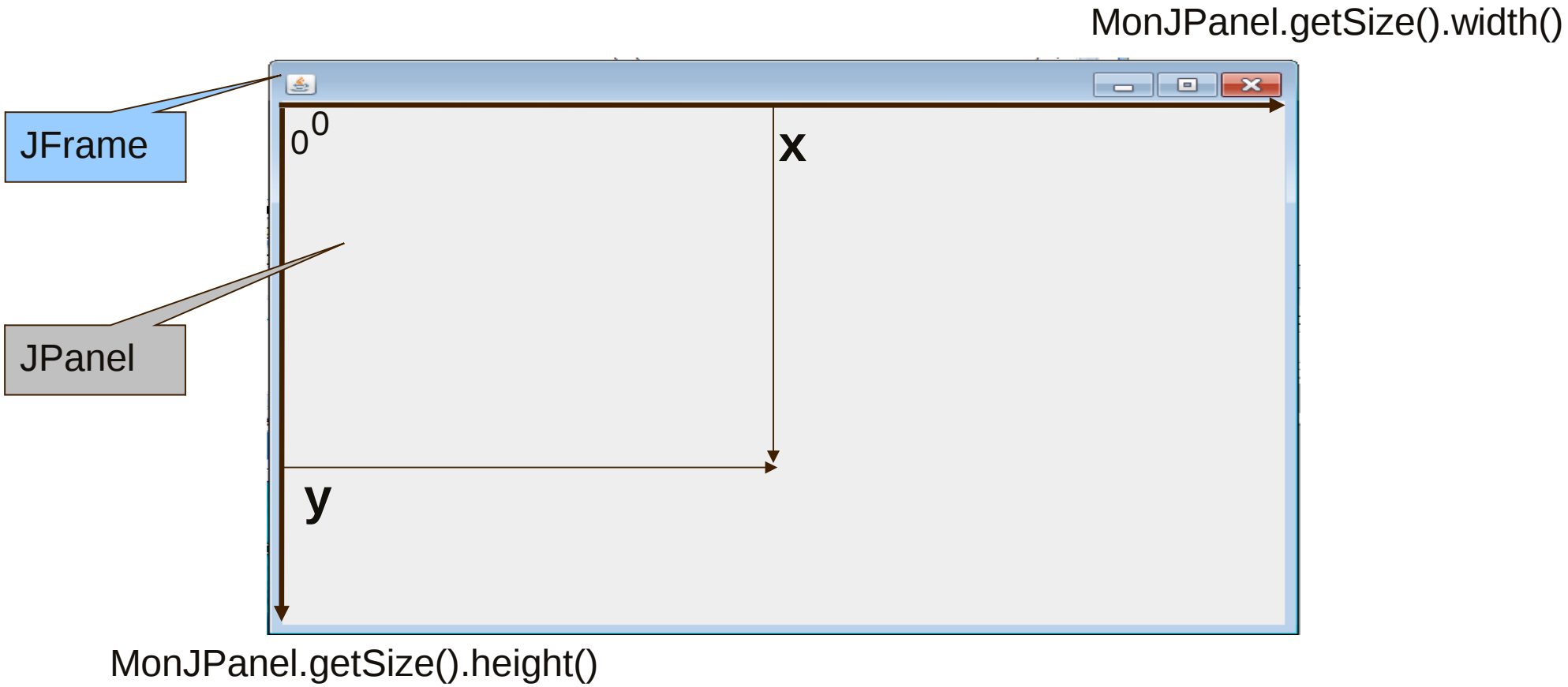
En (0,0) le texte est invisible car il s'affiche sous la bordure de la fenêtre.

Attention si **`setResizable(true);`**

Alors le graphisme déborde en coordonnées absolues

# Exemple 2 : Dessiner dans un Jpanel

## Coordonnées (x,y) absolues dans un Jpanel




En dessinant  dans un JPanel posé sur le ContentPane (vitre) du JFrame,

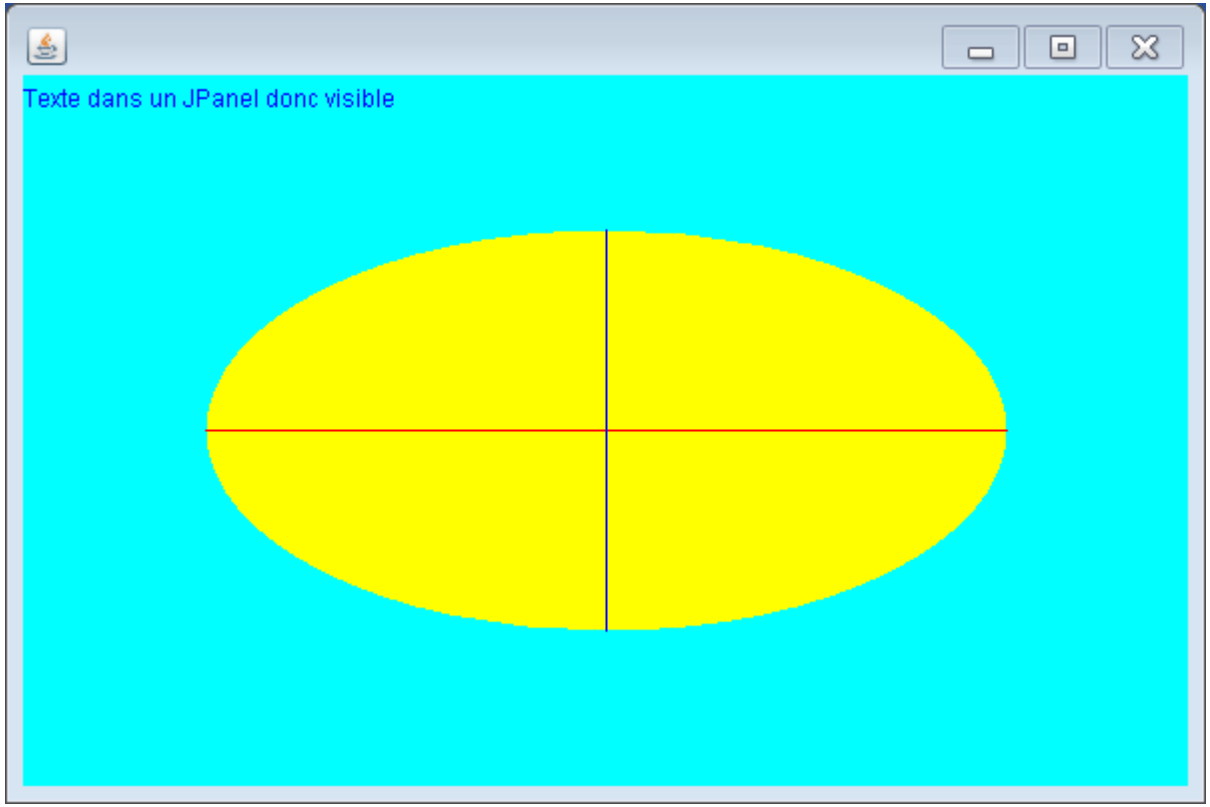
On règle les problèmes du décalage des coordonnées des pixels à cause des bordures

# paintJPanel.java

# Exemple2 : Dessiner dans un JPanel

```
public class paintJPanel extends JFrame {

JPanel drawingPanel;

  public  paintJPanel(int w, int h)  {
   setSize(w,h);
   setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   drawingPanel  = new JPanel();
   getContentPane().add(drawingPanel);
   setVisible(true);
   }

public void paint(Graphics g) {
     // Le centre du JPanel
     int Xc=drawingPanel.getSize().width/2;
     int Yc=drawingPanel.getSize().height/2;
     Graphics gPanel=drawingPanel.getGraphics();

     //...  Même code que précédemment avec gPanel à la place de g
     // mais les coordonnées sont simplifiées entre (0,0)
     // et (drawingPanel.getSize().width,drawingPanel.getSize().height)

     // Ecrire un texte en bleu dans la surface visible
     gPanel.setColor(Color.blue);
     Font font=ggPanel.getFont();
     int HauteurTexte=getFontMetrics(font).getHeight();
     gPanel.drawString("Texte dans un JPanel donc visible",0,HauteurTexte);
}
```

# `paintJPanel.java` Exemple2 : Dessiner dans un JPanel




Même résultat mais avec des coordonnées absolues plus simples car elles sont relatives au Jpanel qui remplit la partie visible du JFrame et commencent en (0,0) et finissent en (getSize().width,getSize().height)

# Exemple 3 : Tracer une fonction

**Courbe2.java**

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

public class Courbe2 extends JFrame {
// dessiner entre Xa et Xb la suite de fonction fn(x)
static double Xa=-5;
static double Xb=+5;

// Constructeur
public  Courbe2(int w, int h)    {
   setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setSize(w,h);
    setVisible(true);
   setTitle("Courbe fn(x)=sin((2n+1)*PI*x/20) entre "+Xa+" et"+Xb);

   }

// coordonnées dans le monde mathématique
// avec des valeurs de y dans [-1,+1] et x dans [Xa,Xb]
    double fn(int n, double x)  {
     return Math.sin((2*n+1)*(Math.PI) * x/20);
    }
```

# Exemple 3 : Tracer une fonction

**Courbe2.java**

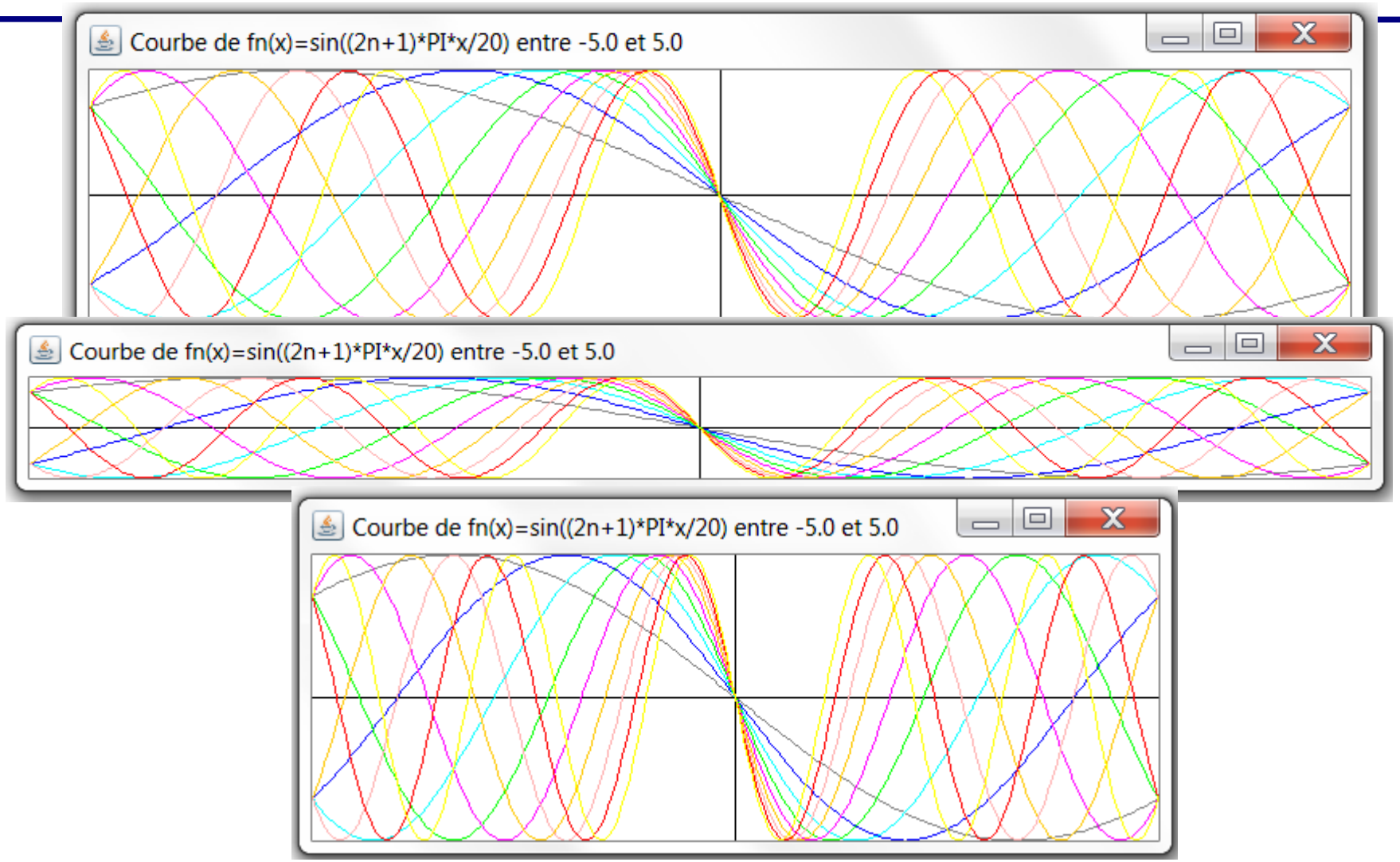
```
// coordonnées dans le monde numérique graphique
// convertit x dans [Xa,Xb] en Xe dans [0,H]
// convertit y dans [-1,+1] en Ye dans [0,L]

   Point  ToScreen(double x, double y)  {
// calcul la hauteur&largeur de la surface visible de la fenêtre
   int H=getSize().height-getInsets().top-getInsets().bottom;
   int L=getSize().width-getInsets().left-getInsets().right;
   int Xe=(int)((x-Xa)/(Xb-Xa)*L);
   int Ye=(int)((y+1)/2*H);
   return new Point(Xe+getInsets().left,Ye+getInsets().top);
   }

public void paint(Graphics g) {
     //  ... Voir page suivante ...
}

public static void main(String[] args) {
     Courbe2 Mafenetre = new Courbe2(800,200);
   }
}
```

# Exemple 3 : Tracer une fonction

**Courbe2.java**

```
 public void paint(Graphics g) {
  Color[] C = {Color.black,Color.gray,Color.blue,Color.cyan,Color.green,

   Color.magenta,Color.orange,Color.pink,Color.red,Color.yellow};
 Point A,B;  double x,y;
 // Mettre le fond à blanc
 g.setColor(Color.white);
 g.fillRect(0,0,getSize().width,getSize().height);
 // Dessiner le repere centré
 g.setColor(Color.black);
 A=ToScreen(0,+1);  B=ToScreen(0,-1);  g.drawLine(A.x,A.y,B.x,B.y);
 A=ToScreen(Xa,0);  B=ToScreen(Xb,0);  g.drawLine(A.x,A.y,B.x,B.y);
// Tracer 10 courbes fn(x)
 for (int n=1; n<10; n++) {
       g.setColor(C[n]);
       x=Xa; y=fn(n,Xa); A=ToScreen(x,y);
       // Trace un courbe fn(x) avec x de Xa à Xb
       while (x<=Xb)
              {
              x+=0.05;   y=fn(n,x);
              B=ToScreen(x,y); g.drawLine(A.x,A.y,B.x,B.y);
              A=B; // on sauve le point précédent
              }
       }
```

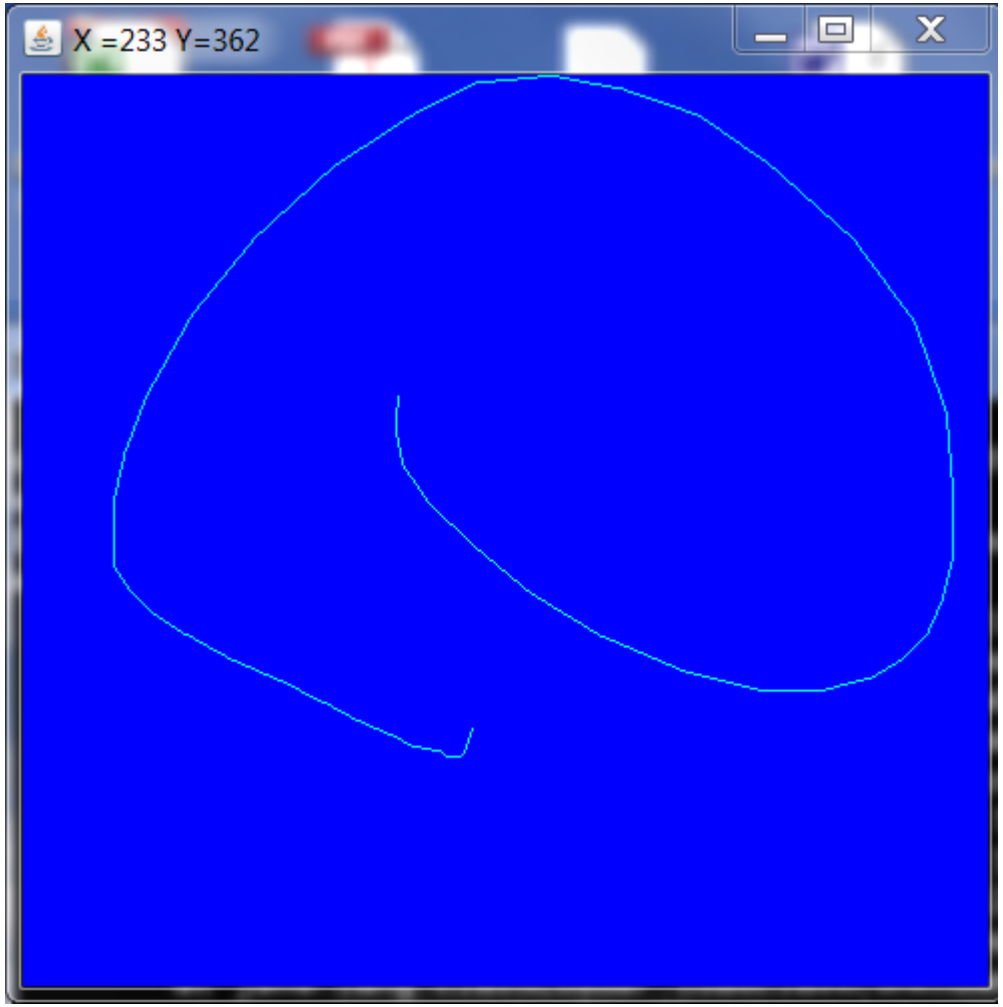
# Exemple 3 : Tracer une fonction




En mettant le tracé de la courbe dans paint() la courbe suit la taille de la fenêtre. L'évènement resize déclenche un paint();

# Exemple 4 : Centipede1 : Graphisme et Souris

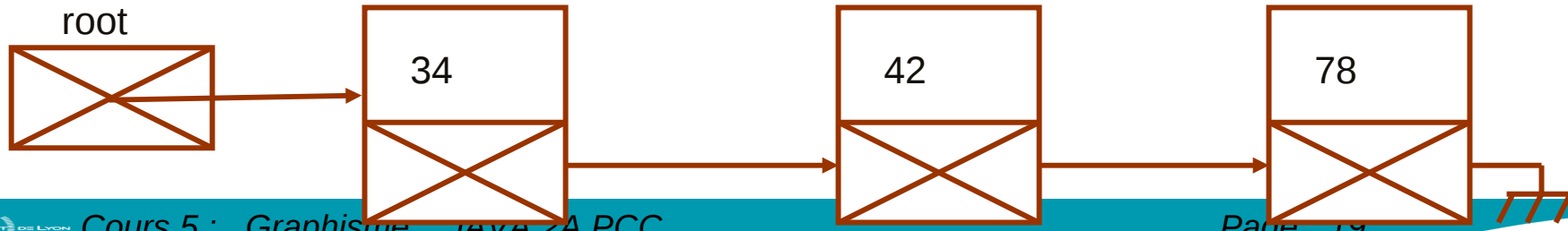
Tracer un centipède à la souris de N pixels dont la taille augmente avec la vitesse de la souris



# Rappel des conteneurs : LinkedList

- Liste chaînée  manuelle :  Complexité de l'insertion/suppression
- A

```
// Création d'une liste chainée
LinkedList<Integer> Maliste = new LinkedList<Integer>();
// Remplissage en fin de liste  : la structure croit automatiquement!
Maliste.add(34);
Maliste.add(42);
Maliste.add(78);
// Accès direct aléatoire
Int i=Maliste.get(1);  // 42
// Suppression direct aléatoire
Maliste.remove(1);
// Suppression en tête de liste  donc le premier élément entré
 Maliste.remove(0);
// Nombre d'éléments  de la liste
Int N= Maliste.size();
```



# Exemple 4 : Centipede1 : Graphisme et Souris

**Centipede1.java**

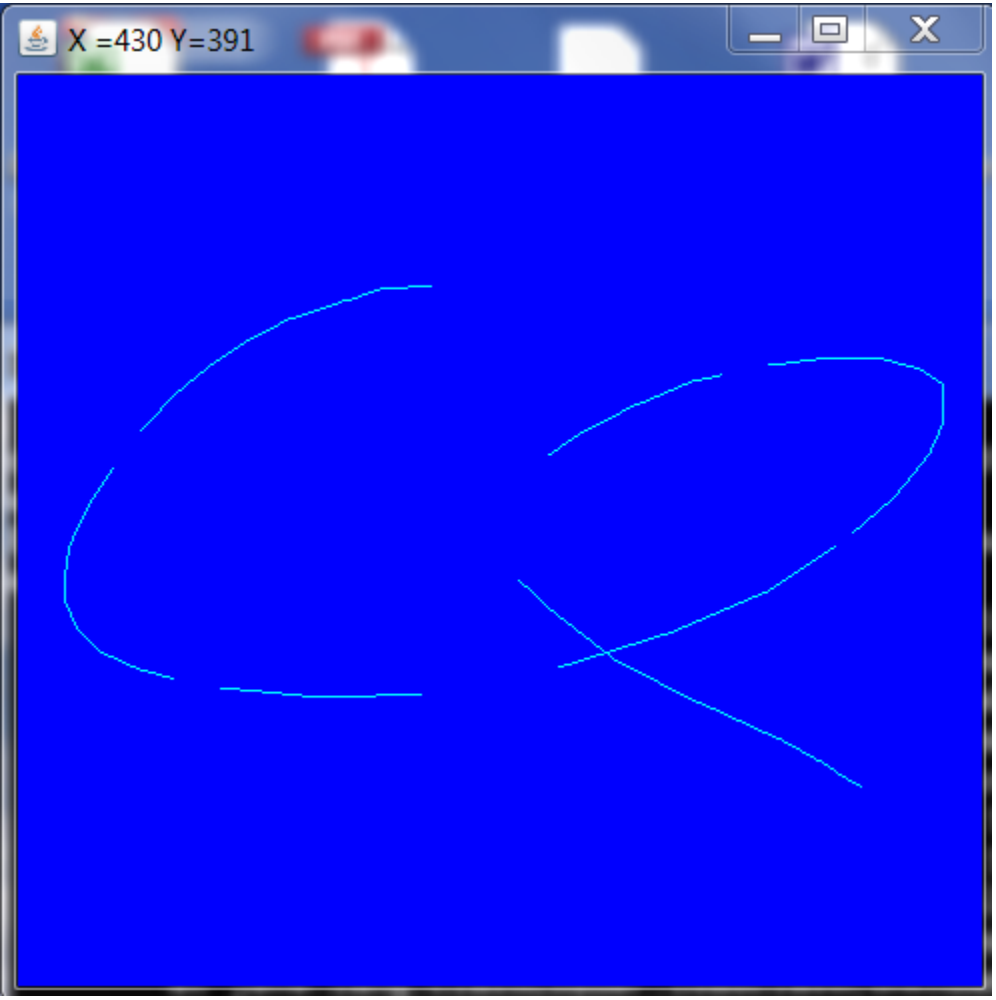
```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
public class Centipede1 extends JFrame
       implements MouseListener, MouseMotionListener {
   // Liste des points a tracer
   LinkedList<Point> listePoints;
   static final int N = 50;  // nbr d'elements Max du centipede

  public  Centipede1(int w, int h) {
  // alloue la liste chainee
   listePoints = new LinkedList<Point>();
  // place les ecouteurs de la souris
   addMouseListener( this );
   addMouseMotionListener( this );
   setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   setSize(w,h); setVisible(true);
  }
 //oblige de surcharger a vide pour compiler
   public void mouseExited( MouseEvent e ) { }
   public void mouseClicked( MouseEvent e ) { }
   public void mousePressed( MouseEvent e ) { }
   public void mouseReleased( MouseEvent e ) { }
   public void mouseDragged( MouseEvent e ) { }
   public void mouseEntered( MouseEvent e ) { }
```

# Exemple 4 : Centipede1 : Graphisme et Souris

```
   // seul ecouteur d'evenement implemente lorsque la souris bouge
   public void mouseMoved( MouseEvent e ) {
      // Supprime le premier element de la liste
      if ( listePoints.size() >= N ) { listePoints.remove( 0 );  }
      // ajoute la nouvelle position à la fin de la liste
      listePoints.add( new Point( e.getX(), e.getY() ) );
     // Affiche les coordonnees de la souris comme titre du JFrame
       setTitle( "X =" + e.getX() + " Y=" + e.getY() );
     // invoque la methode paint(); uniquement si la souris bouge !
      repaint();
     }

  public void paint( Graphics g ) {
     // Remplit le fond de bleu
      g.setColor( Color.blue );
       g.fillRect(0,0,getSize().width,getSize().height);
    // on trace des droites directement dans g de classe Graphics
     g.setColor( Color.cyan );
     for ( int j = 1; j < listePoints.size(); ++j ) {
        Point A = listePoints.get(j-1);
         Point B = listePoints.get(j);
         g.drawLine( A.x, A.y, B.x, B.y );
         }
   }
public static void main(String[] args) {
       Centipede1 Mafenetre = new Centipede1(500,500);     }
}
```

# Exemple 4 : Centipede1 : Graphisme et Souris

## Problème de scintillement

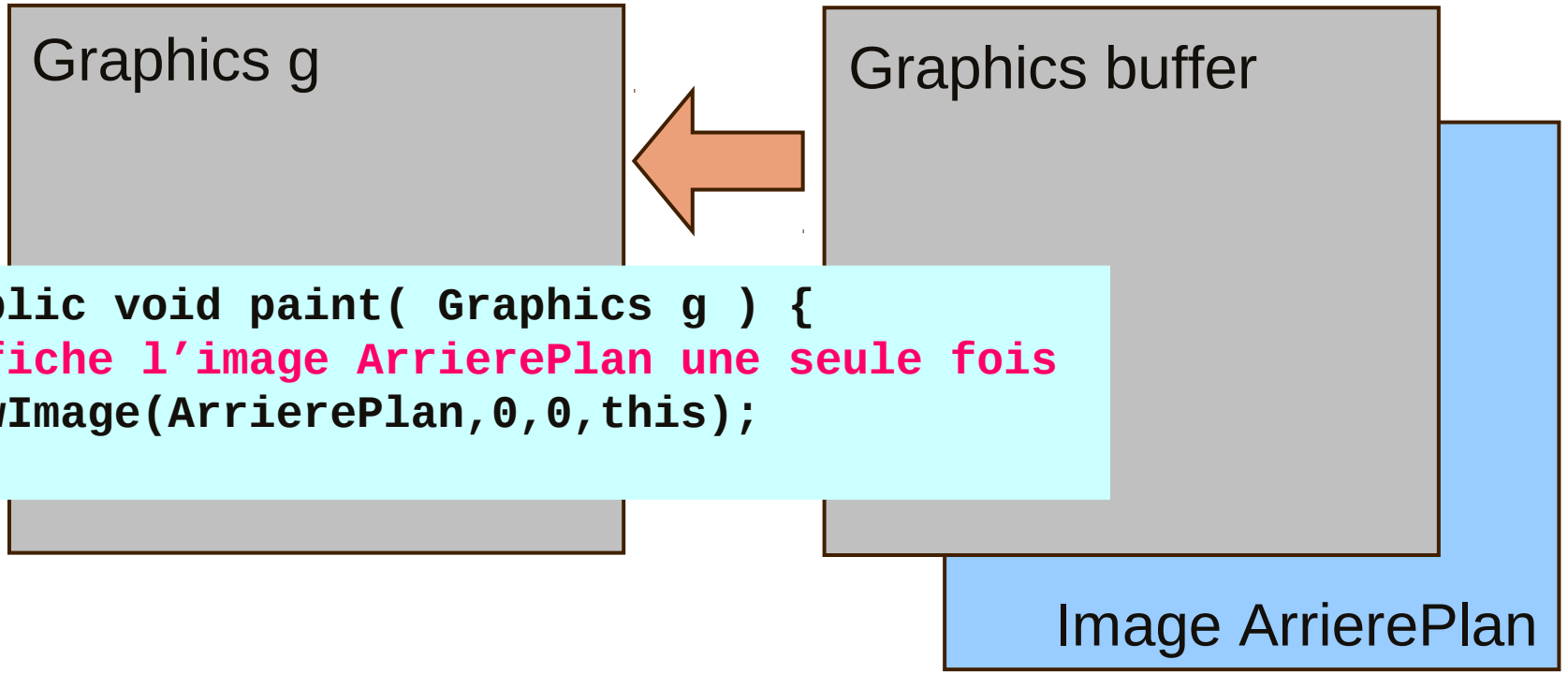



Si la souris bouge vite ou si N>50 alors l'affichage scintille :

le centipede apparaît fragmenté

# Problème de fluidité : Double Buffering

On associe un buffer à une image, on trace dans le buffer sans afficher, on affiche une seule fois l'image par paint().




```
   public void paint( Graphics g ) {
// Affiche l'image ArrierePlan une seule fois
g.drawImage(ArrierePlan,0,0,this);
}
```

```
// Creation de l'arriere plan sur laquelle on va dessiner
BufferedImage ArrierePlan = new
BufferedImage(dim.width,dim.height,BufferedImage.TYPE_INT_RGB);
// Tout ce qui sera dessiné par le buffer sera écrit dans
l'ArrierePlan
```

# Exemple 5 : Centipede2 : Graphisme fluide

## Centipede2.java

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

public class Centipede2 extends JFrame
             implements MouseListener, MouseMotionListener {

   LinkedList<Point> listePoints;
   Color[] PaletteCouleur; // Palette couleur
  // nbr d'elements de la liste chainee
   static final int N = 50;
 // nbr de couleurs de la palette
   static final int Nbcouleurs = 1024;
   Graphics buffer;   // Pour la fluidite des animations
   Dimension dim;     // Dimension de la fenetre de l'Applet
   BufferedImage ArrierePlan; // L'arriere plan sur lequel on dessine
   int Retx,Rety;     // retient la position de la souris
```

# Exemple 5 : Centipede2 : Graphisme fluide

```
public Centipede2(int w, int h)          {
  setSize(w,h);
  setResizable(false);  // Figer la taille de la fenetre
   // recupere la taille de la fenetre de l applet
   dim = getSize();    // récupère la taille réelle de la fenêtre
   // Creation de l'arriere plan sur laquelle on va dessiner
   ArrierePlan = new BufferedImage(dim.width,dim.height,

BufferedImage.TYPE_INT_RGB);
  // Récupération du buffer graphique sera associé à l'ArrierePlan
  buffer = ArrierePlan.getGraphics();
  // L'arrière-plan va être en noir
  buffer.setColor( Color.black );
  // alloue la liste chainee
  listePoints = new LinkedList<Point>();
  // alloue une palette couleur
  PaletteCouleur = new Color[ Nbcouleurs ];
   // On parcourt le spectre de la teinte H ie toutes les couleurs
  // la saturation S et la luminance B sont au maximum à 1
    for ( int i = 1; i <= Nbcouleurs; ++i )
          PaletteCouleur[ i-1 ] =
           new Color( Color.HSBtoRGB(i/(float)Nbcouleurs,1,1) );
  // place les ecouteurs de la souris
    addMouseListener( this );
    addMouseMotionListener( this );
 }
```

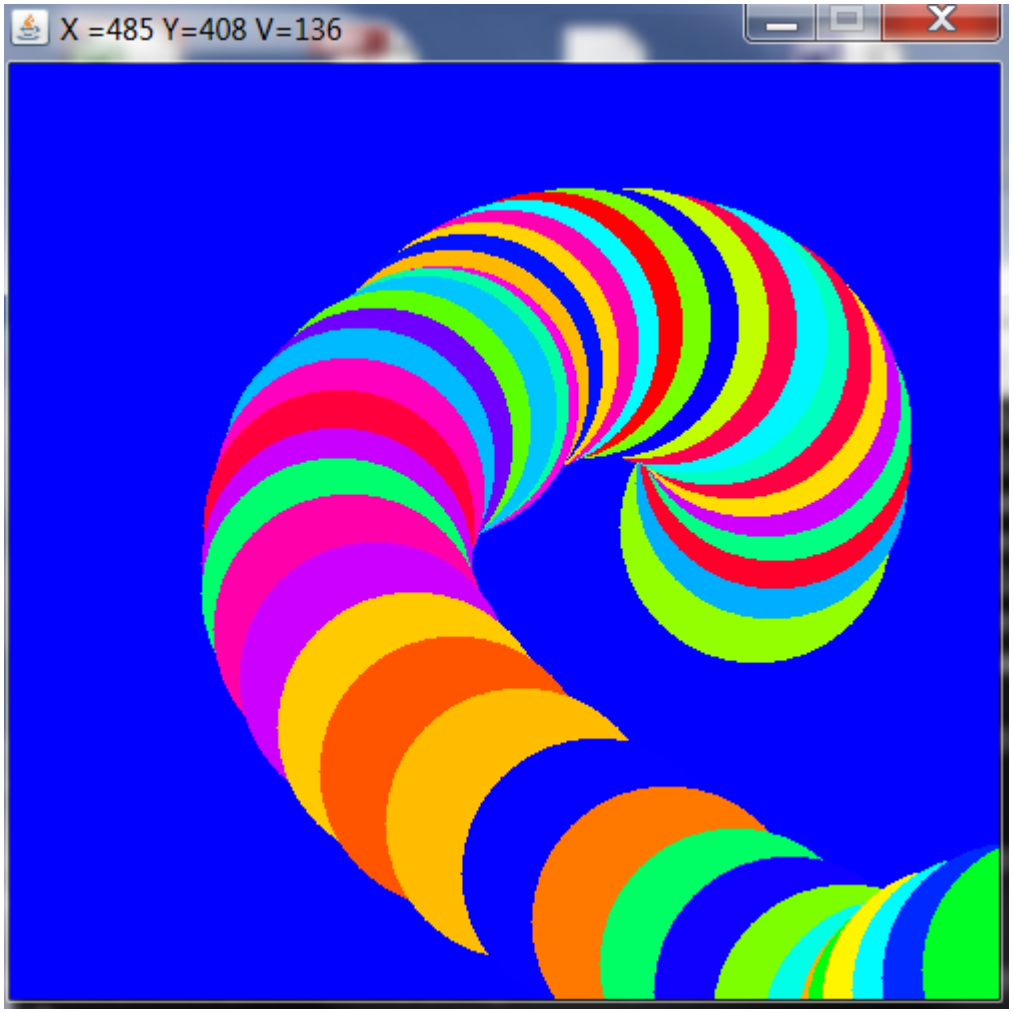
# Exemple 5 : Centipede2 : Graphisme fluide

```
//oblige de surcharger a vide pour compiler
public void mouseExited( MouseEvent e ) { }
public void mouseClicked( MouseEvent e ) { }
public void mousePressed( MouseEvent e ) { }
public void mouseReleased( MouseEvent e ) { }
public void mouseDragged( MouseEvent e ) { }

// Seules methodes implementees

 // MouseEntered appellé quand la souris entre en action
 // Elle permet de conserver la position initiale
public void mouseEntered( MouseEvent e ) {
Retx=e.getX();  Rety=e.getY();
 }

// Paint() ne recopie l'image uniquement quand tous les éléments
// graphiques sont placés dans le buffer associé à l'ArrièrePlan
 public void paint( Graphics g ) {
// Trace le buffer associé à  ArrierePlan dans le JFrame
          g.drawImage(ArrierePlan,0,0,this);
}
```

# Exemple 5 : Centipede2 : Graphisme fluide

```
public void mouseMoved( MouseEvent e ) {
  // Supprime le premier element de la liste
  if ( listePoints.size() >= N ) listePoints.remove( 0 );
  // ajoute la nouvelle position à la fin de la liste
  listePoints.add( new Point( e.getX(), e.getY() ) );
   // on efface le buffer graphique
   buffer.setColor(Color.black);
   buffer.clearRect(0,0,dim.width,dim.width);
   // on définit la taille des ellispes par la vitesse de la souris
   int vitesse=4*(1+Math.abs(Retx-e.getX())+Math.abs(Rety-e.getY()));
   // on conserve la nouvelle position de la souris
   Retx=e.getX(); Rety=e.getY();
   // Affiche La position de la souris en temps réel
    setTitle( "X =" + e.getX() + " Y=" + e.getY() + " V=" + vitesse);
    // on trace N ellispes sans les afficher dans le buffer Graphique
   for ( int j = 1; j < listePoints.size(); ++j ) {
       buffer.setColor(PaletteCouleur[(int)
   (Math.random()*Nbcouleurs)]);
        Point A = listePoints.get(j-1);
        buffer.fillOval(A.x,A.y,vitesse,vitesse);
        }
   // invoque la methode paint(); uniquement si la souris bouge !
  repaint();
  }
```

# Exemple 5 : Centipede2 : Graphisme fluide



# Timer

- Necessité de gérer le temps pour synchroniser des actions
- Timer, permet d'appeler une action tous les x m

```
// Création d'un timer qui va battre tous les 100 millisecondes
Timer timer = new Timer(100, new TimerAction());
// active le timer qui va appeler actionPerformed()
// de la classe TimerAction  10 fois par seconde
timer.start();
// Arret du timer lors d'un clique d'un bouton
timer.stop();
 //  Implémentation de TimerAction
class TimerAction implements ActionListener {

public void actionPerformed(ActionEvent e) {
   // Réalise les opérations sur les objets qui dépendent du temps
   // Rafraichit l'image si nécessaire
    repaint();
  // Compte le temps (utile pou savoir combien de temps s'est écoulé)
    temps++;
    }
}
```

# Exemple n°6 Animation

## Jeu : Taper sur la tête de régis

**Gereimage1.java**

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

public class Gereimage extends JFrame {
 Graphics buffer;   // Pour la fluidite des animations
 Image regis;       // La tête de Regis qui va s'animer
 int x,y;           // La position du Regis
 double dx,dy;      // le vecteur de deplacement
 int vitesse;       // La vitesse de deplacement
 Dimension dim;     // Dimension de la fenetre
 BufferedImage ArrierePlan; // L'arriere plan sur lequel on dessine
 Image wall;         // L'image du papier peint
 int   interval     // Millisecondes entre les mises a jour
 Timer timer;       // Timer pour l'animation
 long cycles        // compte le nombre de cycles
```

```
public  Gereimage(int w, int h) {
     // Fixe les parametres
     interval= 40;  cycles=0; setSize(w,h); setResizable(false);
     // recupere la taille de la fenetre réelle et pas (w,h)
     dim = getSize();
     // Recupere localement la tete de regis la ou est le code
     Toolkit T=Toolkit.getDefaultToolkit();
     regis = T.getImage("regis.png");
     // Creation de l'arriere plan sur laquelle on va dessiner
     ArrierePlan BufferedImage(dim.width,dim.height,
                                          BufferedImage.TYPE_INT_RGB);
     // Tout ce qui sera dessine par le buffer sera dans l'ArrierePlan
     buffer = ArrierePlan.getGraphics();
     // charge une image 400x400 d un mur comme papier peint
     wall = T.getImage("wall.png");
     // creation d un timer qui va cycler tous les interval ms
     // associe a une classe TimerAction qui implemente ActionListener
     timer = new Timer(interval, new TimerAction());
     // donne une position aleatoire au depart
     x=(int)(Math.random()*(dim.width-100) + 100);
     y=(int)(Math.random()*(dim.height-100) + 100);
     // donne une direction et une vitesse fixe initiale
     dx=-1; dy=1; vitesse=16;
     timer.start();  // Lance le Timer
```

# Exemple n°6 Animation

```
// gere les rebonds sur les bords de la fenetre
[0,dim.width]x[0,dim.height]
public void move() {
        x += (int)vitesse*dx;
        y += (int)vitesse*dy;
        if (x < 0)  { x=0;  dx=-dx; }
                    else
                    if (x+regis.getWidth(null)>dim.width)
                        { x=dim.width-regis.getWidth(null); dx=-dx;}

        if (y < getInsets().top ) { y=getInsets().top ; dy = -dy; }
                   else
                      if (y+regis.getHeight(null)>dim.height)
                      {  y=dim.height-regis.getHeight(null); dy=-dy; }
    }
```

Inutile de gérer les bordures gauche (**getInsets().left** ), droite (**getInsets().right** ) et bas **getInsets().bottom** cat elles sont trop minimes pour impacter le jeu

# Exemple n°6 Animation

```
public void paint(Graphics g)      {
       // Efface l'image de Regis a la position précédente
        buffer.drawImage( wall, 0, 0, this);
      // NE JAMAIS ECRIRE DIRECTEMENT g.drawImage( regis, x, y, this);
      // tres lent et cela scintille
      // plus rapide consiste a tout ecrire dans un Graphics
        buffer.drawImage( regis, x, y, this);
        // puis d'afficher le Graphics comme une image normale.
        g.drawImage(ArrierePlan,0,0,this);
     }
```

**NE JAMAIS** faire de calculs dans la méthode Paint ni faire de lecture/ecriture de fichier ni d'opérations lentes car paint() va être appelée 25 fois par secondes !!!!!!!!!!!!!!!

Jamais ~~`regis = getImage("regis.png");`~~ dans un paint() !

# Exemple n°6 Animation

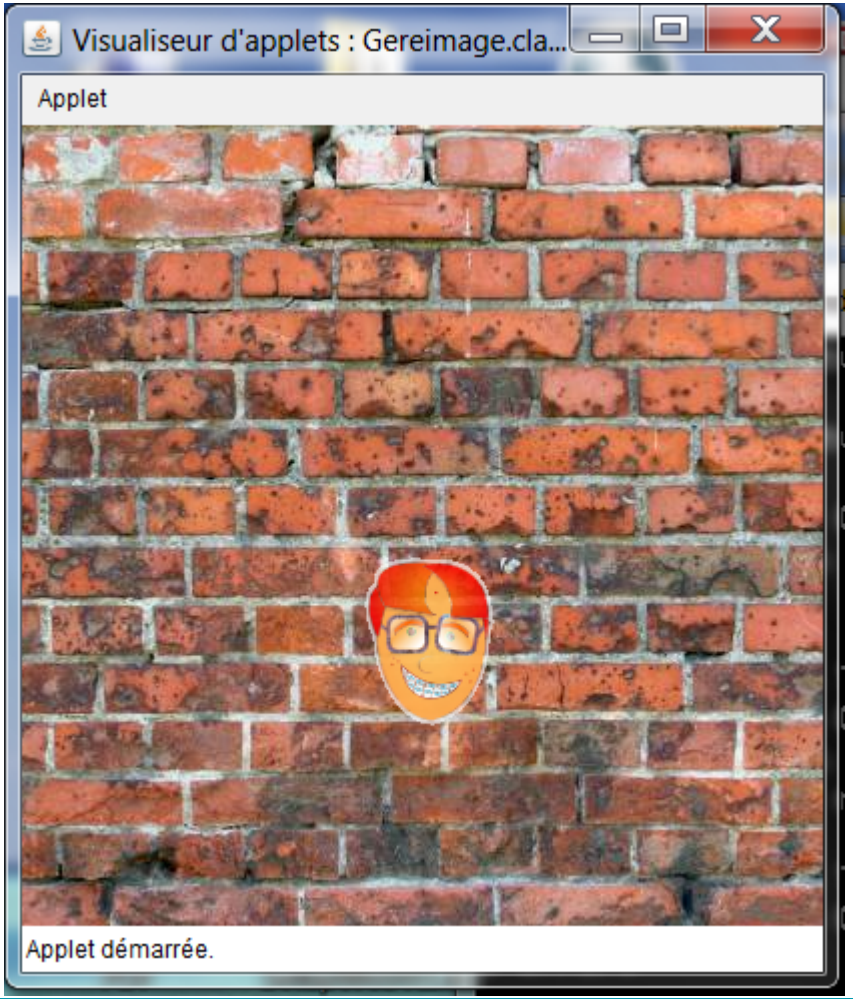
Dans ma classe **Gereimage** on introduit une classe **TimerAction** qui implémente un **ActionListener** dont la méthode **actionPerformed()** est appellée tous les (interval=40) millisecondes

```
class TimerAction implements ActionListener {
    // ActionListener appelee 25 fois par secondes
    public void actionPerformed(ActionEvent e) {
    move();   repaint();
    cycles++;
    if (cycles%100==0) // toutes les 4 sec on augmente la vitesse
              {
              vitesse+=16;
              vitesse=16+vitesse%64; // Mais avec une limite et min=16
              }
    if (cycles%50==0) // toutes les 2 sec on change de direction
              {          // pour un vecteur de direction entre [-1,1]
              dx=(Math.random()*2.0-1.0);
              dy=(Math.random()*2.0-1.0);
              }
       }
}
```

# Exemple n°6 Transparence

Le déplacement fonctionne mais l'image affichée est de forme carrée

Afficher une forme quelconque → Gestion de la transparence

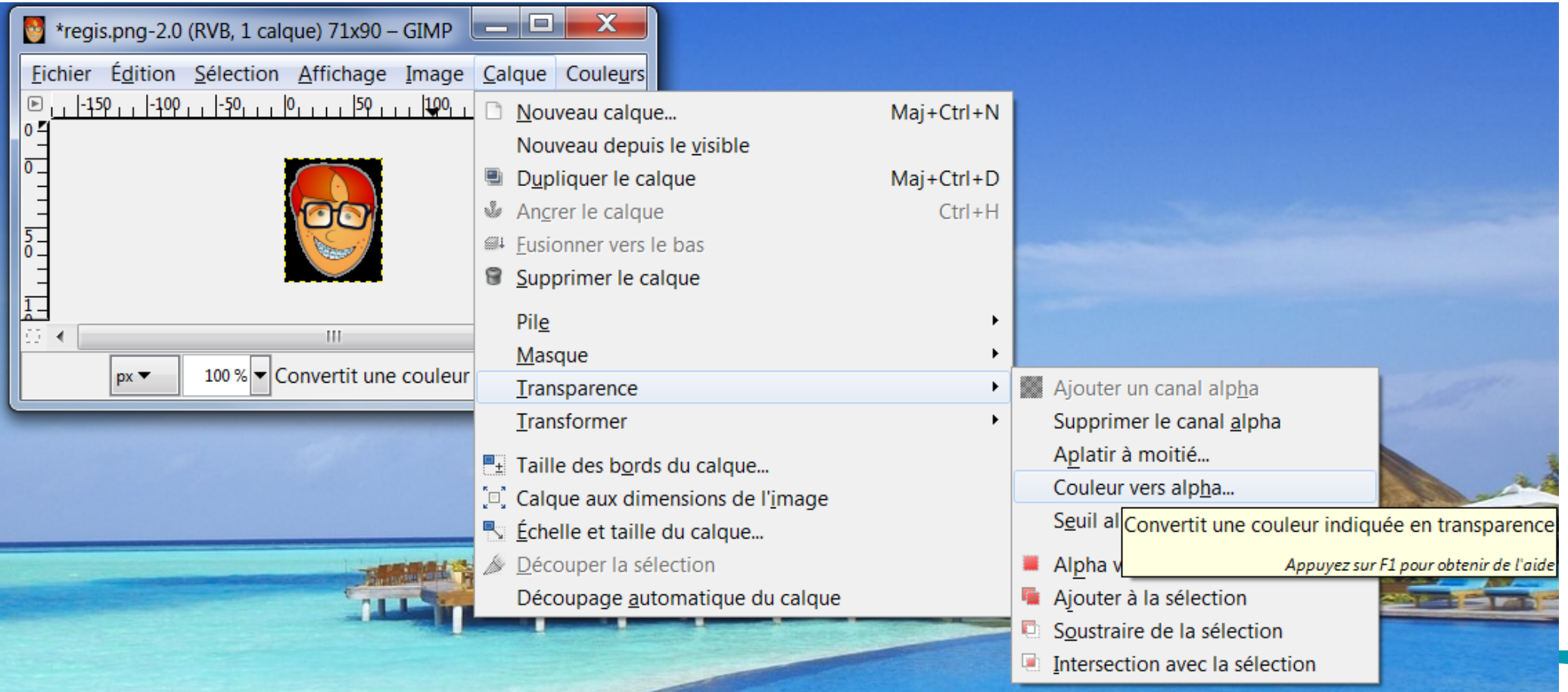



# Exemple n°6  Transparence

Pour gérer la transparence il n'y a rien à rajouter au code !

Il faut tout simplement créer un plan alpha définissant la transparence

Certains logiciels comme Gimp permettent de définir la transparence

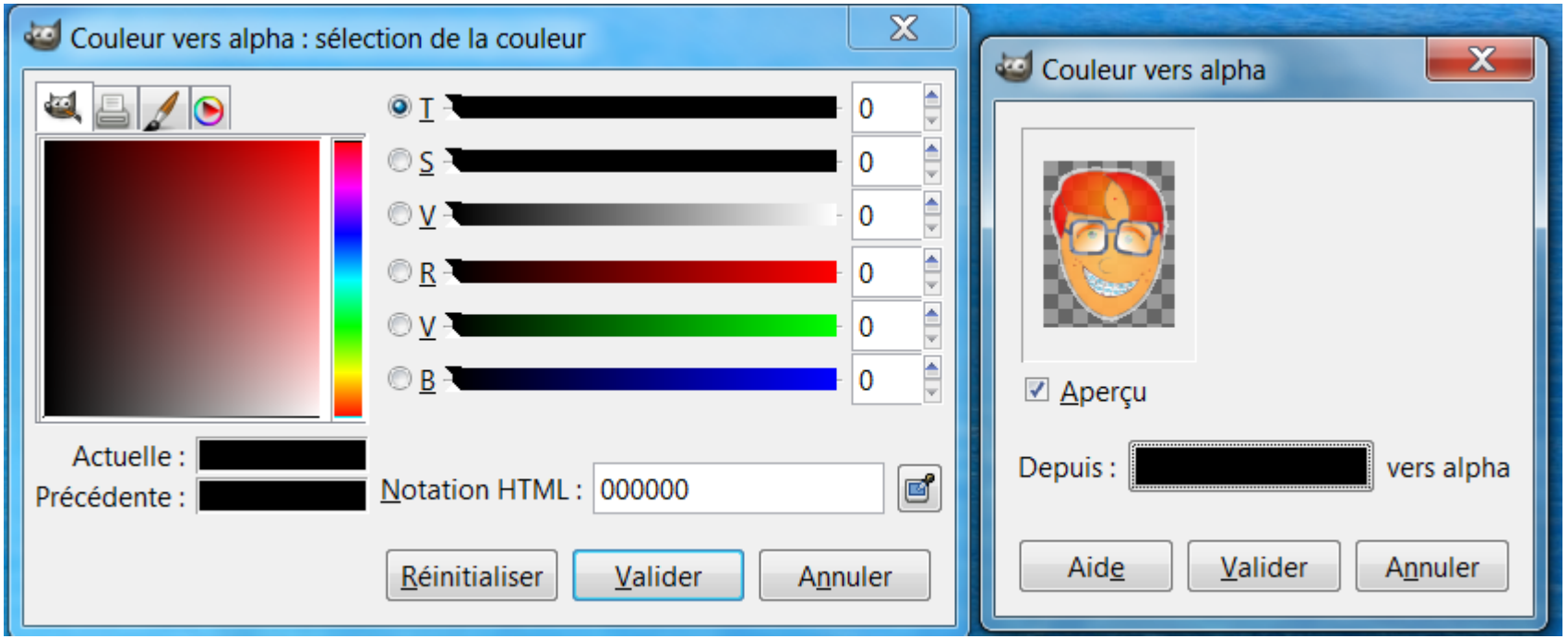


# Exemple n°6 Transparence

Choisir la couleur qui servira de repère à la localisation des pixels transparents à ne pas afficher.

Ici nous choisirons la couleur noir pure

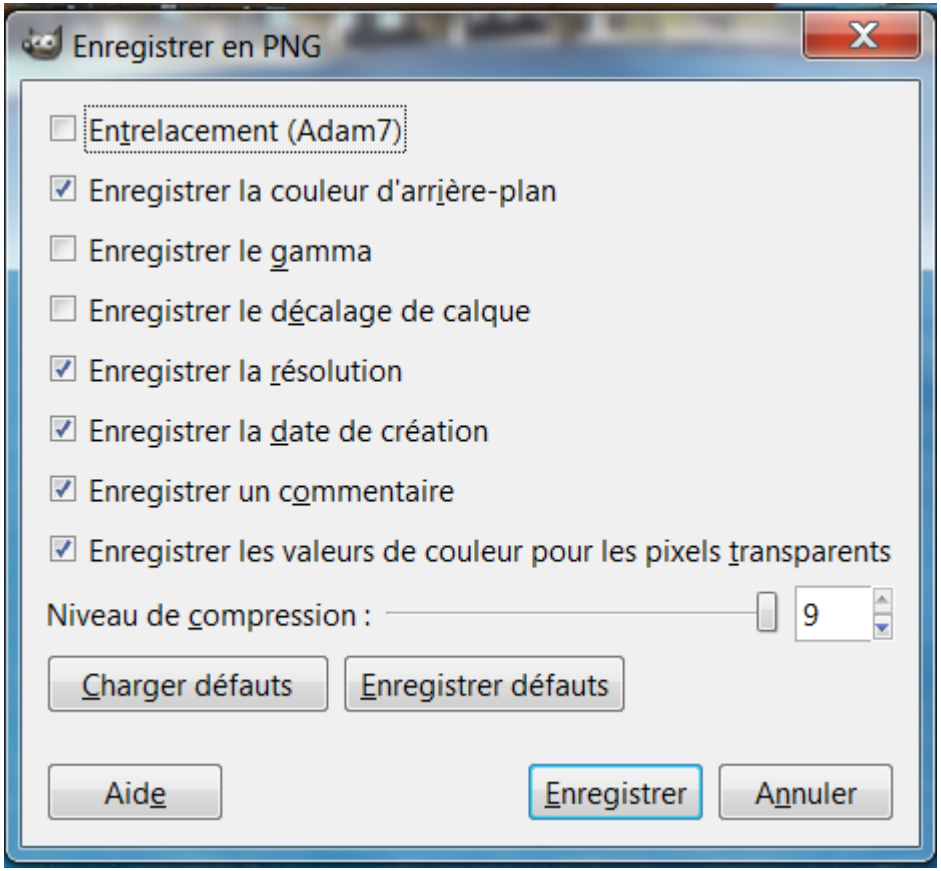
Si cette couleur est aussi présente parmi les pixels à afficher, il faut changer légèrement les couleurs de ceux-ci avec Gimp



# Exemple n°6 Transparence

Ne pas oublier de sauver au format PNG

Ne pas oublier de sauver les informations de transparence en cochant les bonnes cases des paramètres de sauvegarde



# Exemple n°7 Jeu « Taper Regis »

Gereimage2.java

```
public class Gereimage2 extends JFrame
                    implements MouseListener, MouseMotionListener {
// attributs de la classe Gereimage auxquelles on ajoute celles-ci :
Image boom;             // Image de l explosion
boolean explosion;      // explosion declenchee ?
long    nbcoups;        // nb coups gagnants
long    tempsjeu;       // nombre de secondes max que dure le jeu
 public Gereimage2(int w, int h) {
   // La partie initialisation de Gereimage auquelle on ajoute :
   boom = T.getImage("boom.png");
   addMouseListener( this );
   addMouseMotionListener( this );
   explosion=false;  nbcoups=0;  tempsjeu=10*(1000/interval); //=10s
   timer.start(); // Lance le jeu dès sa construction }
  // code implemente quand on clic sur un des boutons de la souris
   public void mousePressed( MouseEvent e ) {
      int mx = e.getX();   int my = e.getY();
      if ( x < mx && mx < x+regis.getWidth(null) &&
           y < my && my < y+regis.getHeight(null) )
                          { explosion = true;  }

   }
```

# Exemple n°7  Jeu « Taper Regis »

```
// code non implementé mais necessaire pour que cela compile
   public void mouseEntered( MouseEvent e ) { }
   public void mouseExited( MouseEvent e ) { }
   public void mouseClicked( MouseEvent e ) { }
   public void mouseReleased( MouseEvent e ) { }
   public void mouseMoved( MouseEvent e ) { }
   public void mouseDragged( MouseEvent e ) { }

class TimerAction implements ActionListener {
        // ActionListener appelee tous les interval millisecondes

        public void actionPerformed(ActionEvent e) {
        move();
        repaint();
       cycles++;
        if (cycles>=tempsjeu) timer.stop(); // On arrête le jeu
      // le score sera affiché
       // le reste du code ne change pas
}
```

# Exemple n°7 Jeu « Taper Regis »

```
public void paint(Graphics g)  {
  if (timer.isRunning()==false) {
       setTitle("Score ="+nbcoups+" ! "); }
 else {
       buffer.drawImage( wall, 0, 0, this);
       if (explosion) {
             buffer.drawImage( boom, x, y, this);
             nbcoups++; explosion=false; // on compte un coup de plus
             // on ralentit le temps de 0.5s pour voir l'explosion
             timer.setDelay(500); }
                    else {
             // si temps est ralenti on le remet comme avant
             if (timer.getDelay()==500)  {
                   buffer.drawImage( boom, x, y, this);
                   timer.setDelay(interval);
                   }
                    else buffer.drawImage( regis, x, y, this);
             }
       g.drawImage(ArrierePlan,0,0,this);
      if (cycles%25==0) // on indique le temps restant
      showStatus( "Reste" +(int)(((tempsjeu-cycles)*interval)/1000));
      }
```

# Exemple n°7  Jeu « Taper Regis »

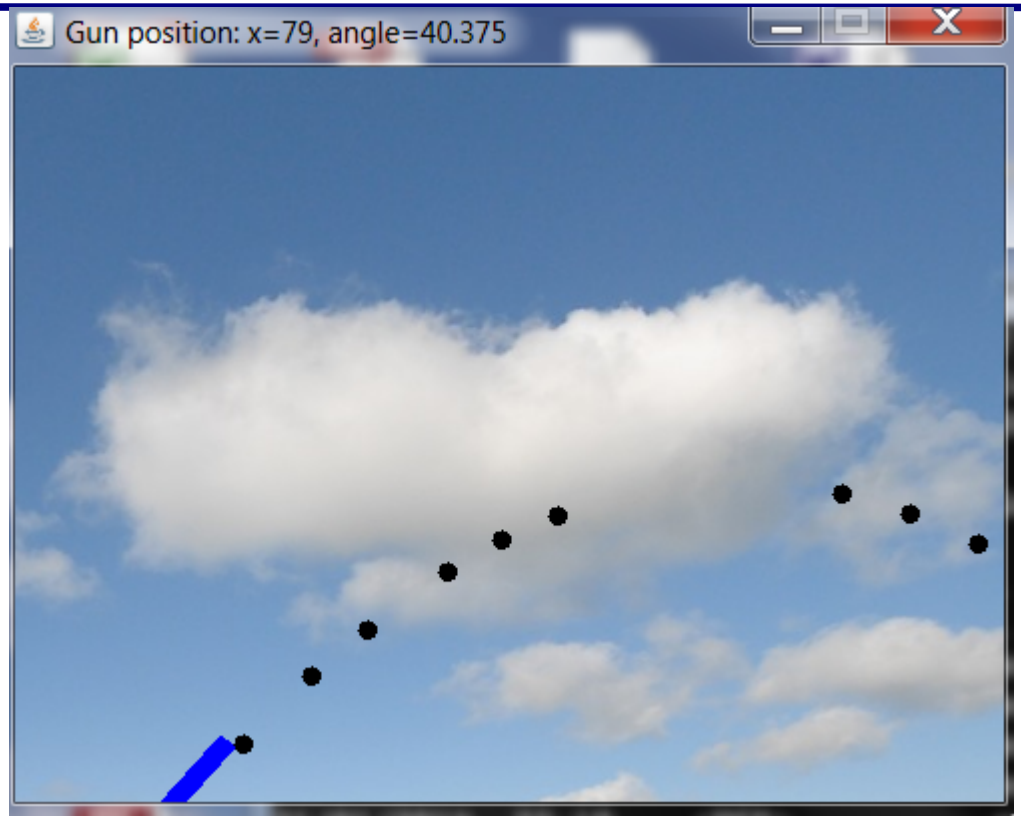


En prennant une image GIF animée, on réalise des animations d'objets graphiques de façon simple et fluide

# Opérations graphiques avancées

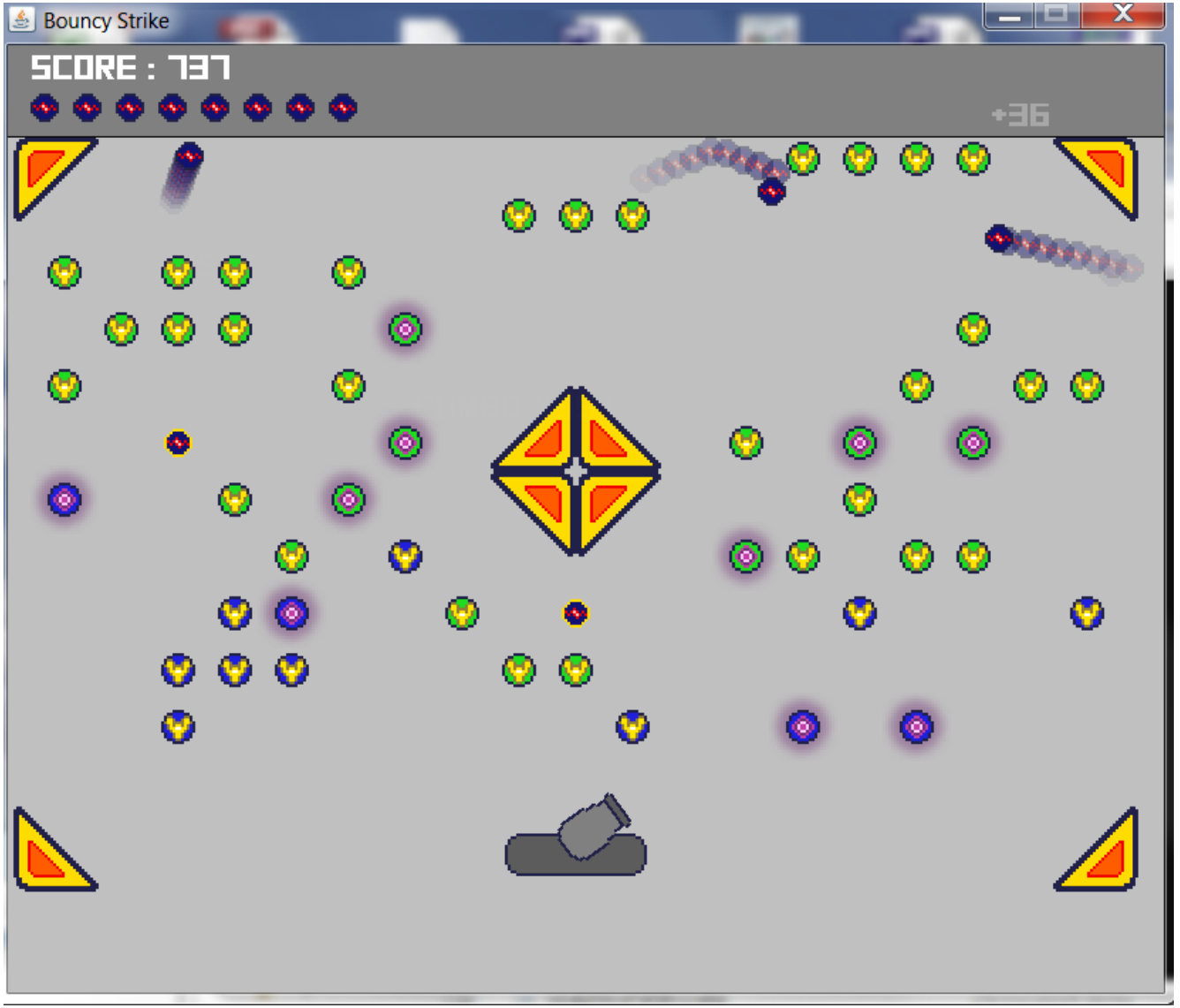
- Dérivée de Graphics contient des méthodes avancées pour manipuler des formes (shape) et leurs contours (strokes) réaliser des transformations affines (AffineTransform), des rotations (rotate), des translations (translate), des étirements (scale)
- **draw**(Shape s)
- **drawImage**(BufferedImage img, BufferedImageOp op, int x, int y)
- **drawImage**(Image img, AffineTransform xform, ImageObserver obs)
- **drawRenderableImage**(RenderableImage img, AffineTransform xform)
- **fill**(Shape s)
- **rotate**(double theta, double x, double y)
- **scale**(double sx, double sy)
- **setStroke**(Stroke s)

# Exemple 8 : Canon




```
public void paint (Graphics g) {
// Pour accéder aux fonctions graphiques avancées il suffit de transtyper g

Graphics2D g2 = (Graphics2D) g;
// g2 permet de définir l'épaisseur des traits ce que ne peut pas faire g
g2.setStroke(new BasicStroke(10));
g2.setColor(Color.blue);
g2.drawLine(xa,ya,xb,yb);
```

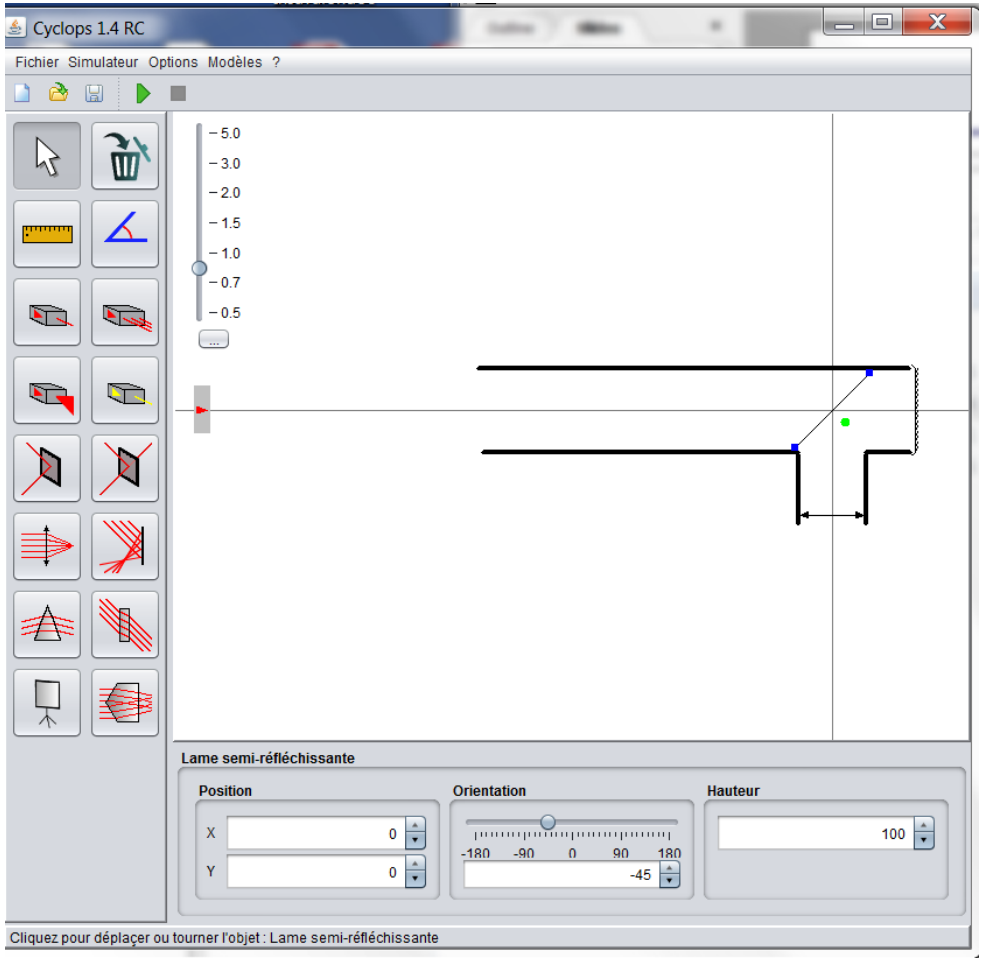
# Exemple Projet 2013 : Bouncy Strike

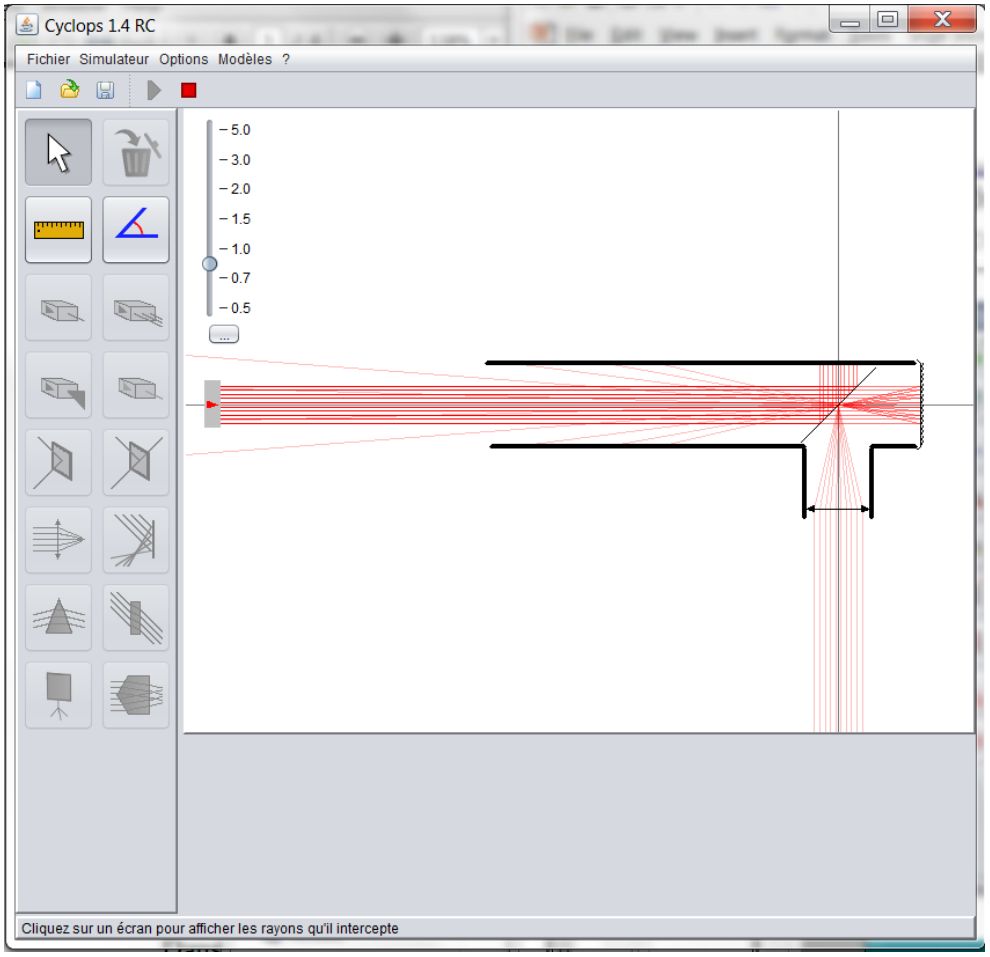



Compile.bat exec.bat ou bien javac *.java java BouncyStrike

# Exemple Projet 2013 : Cyclops

Banc de simulation optique proposant une gamme complète d'objets optiques (miroirs plans ou sphériques, lentilles, prismes, sources ponctuelles ou étendues, écrans...).

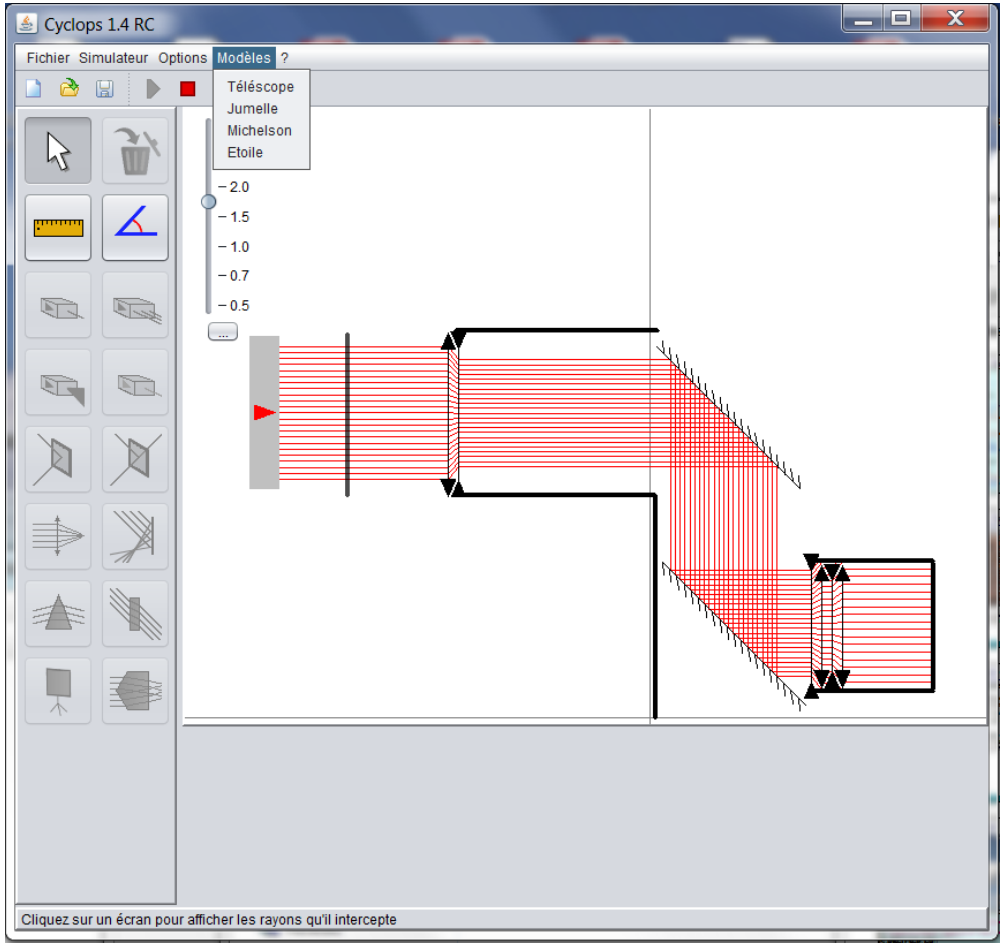


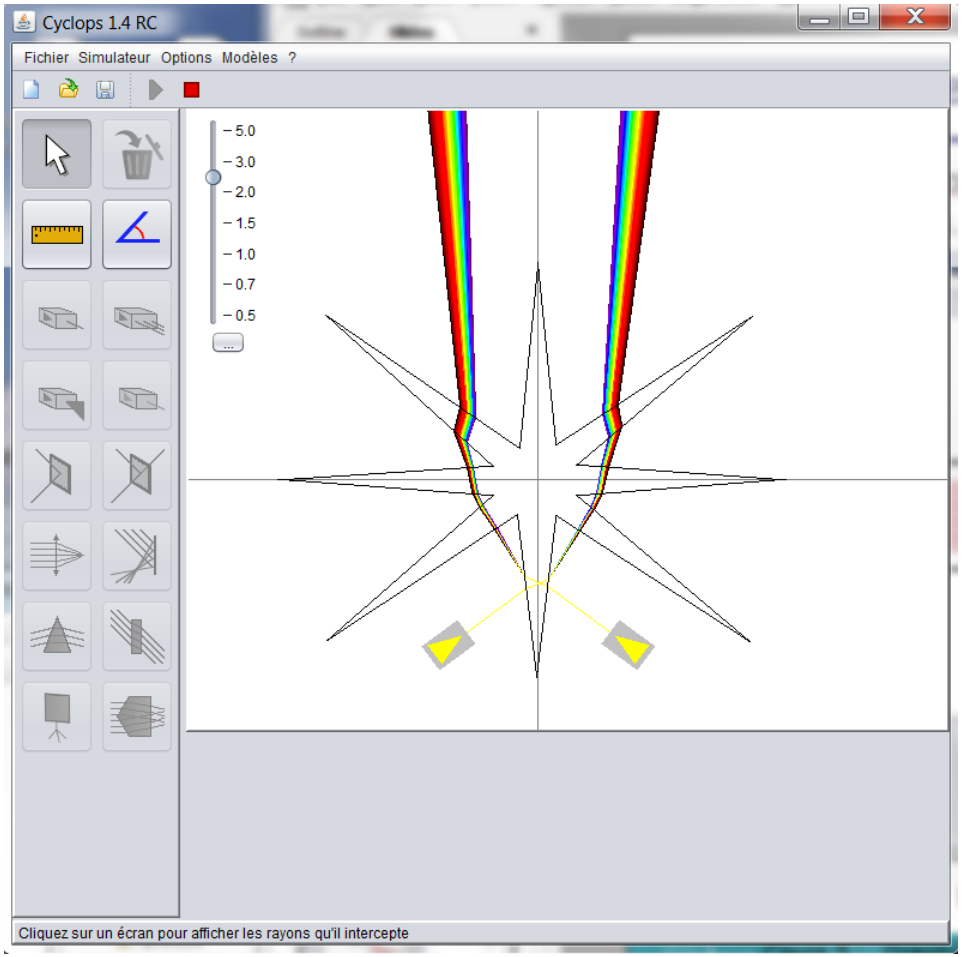




Executer Cyclops.jar (**J**ava **AR**chive)

# Exemple Projet 2013 : Cyclops

Banc de simulation optique proposant une gamme complète d'objets optiques (miroirs plans ou sphériques, lentilles, prismes, sources ponctuelles ou étendues, écrans...).




Executer Cyclops.jar (**J**ava **AR**chive)