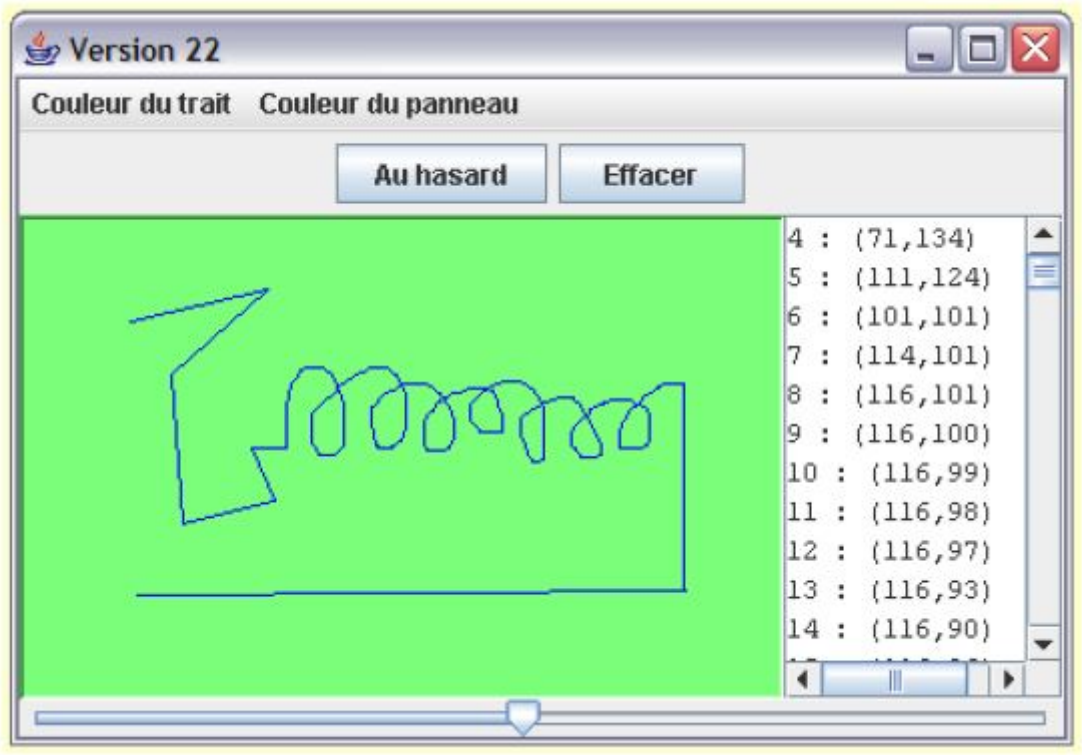
NFP 121 : TP interfaces graphiques

Cette page présente les programmes qui ont illustré le cours sur la construction d'une interface homme-machine en Java, avec la bibliothèque MFC/Swing. Pour fixer les idées, voici le produit final visé :



Voici les différentes étapes (les modifications significatives, de chaque étape à sa suivante, sont montrées en bleu).

Point.java - Une classe auxiliaire bien connue :

```
public class Point {
    public static final int MAX_X = 400;
    public static final int MAX_Y = 400;

    private int x, y;

    public Point(int a, int b) {
        x = a;
        y = b;
    }
    public Point() {
        x = (int)(Math.random() * MAX_X);
        y = (int)(Math.random() * MAX_Y);
    }
    public int getX() {
        return x;
    }
    public int getY() {
        return y;
    }
    public String toString() {
        return "(" + x + "," + y + ")";
    }
}
```

Truc00.java - Pour vérifier le bon fonctionnement de la classe `Point` on crée et on affiche une collection de 20 points aléatoirement disposés :

```
import java.util.Vector;

public class Truc00 {

    public static void main(String args[]) {
        Vector points = new Vector();

        for (int i = 0; i < 20; i++)
            points.add(new Point());

        System.out.println(points);
    }
}
```

Truc01.java - Apparition de l 'interface graphique, pour commencer la plus réduite possible : rien qu'un cadre, vide mais fonctionnel. Notez que la case de fermeture n'est pas opérationnelle (la fenêtre disparaît, mais l'application ne se termine pas) :

```
import javax.swing.*;

public class Truc01 {

    public static void main(String[] args) {
        JFrame cadre = new JFrame("Version 1");
        cadre.setSize(Point.MAX_X, Point.MAX_Y);
        cadre.setVisible(true);
    }
}
```

Truc02.java - Une autre manière d'obtenir le même résultat que ci-dessus : écrire une sous-classe de `JFrame`. Ce n'est pas une obligation mais, en faisant ainsi, de nombreuses opérations deviendront par la suite plus simples à écrire :

```
import javax.swing.*;

public class Truc02 extends JFrame {

    public Truc02() {
        super("Version 2");
        setSize(Point.MAX_X, Point.MAX_Y);
        setVisible(true);
    }

    public static void main(String[] args) {
        /* JFrame cadre = */ new Truc02();
    }
}
```

Truc03.java - Ajout de quelques composants au cadre : un panneau (`JPanel`) supérieur, ici rose, portant deux boutons (`JButton`) et un panneau central :

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

public class Truc03 extends JFrame {

    public Truc03() {
        super("Version 3");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        panneau.setBackground(Color.PINK);
        getContentPane().add(panneau, BorderLayout.NORTH);
```

```
        JButton bouton = new JButton("Au hasard");
        panneau.add(bouton);

        bouton = new JButton(" Effacer ");
        panneau.add(bouton);

        panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.CENTER);

        setVisible(true);
    }

    public static void main(String[] args) {
        new Truc03();
    }
}
```

Truc04.java - On détecte les actions sur le premier bouton. Pour commencer, on fait cela en faisant en sorte qu'un objet `Truc` soit un auditeur d'événements action (`ActionListener`) et en lui faisant écouter le bouton en question :

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

public class Truc04 extends JFrame implements ActionListener {

    public Truc04() {
        super("Version 4");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        panneau.setBackground(Color.CYAN);
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);

        bouton = new JButton(" Effacer ");
        panneau.add(bouton);

        panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.CENTER);

        setVisible(true);
    }

    public void actionPerformed(ActionEvent evt) {
        System.out.println("Pression du bouton 'Au hasard'");
    }

    public static void main(String[] args) {
        new Truc04();
    }
}
```

Truc05.java - Ici, l'objet `Truc` est l'auditeur d'événements *action* des deux boutons ce qui oblige à identifier la source d'un tel événement lorsqu'il se produit :

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

public class Truc05 extends JFrame implements ActionListener {

    public Truc05() {
```

```
        super("Version 5");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        panneau.setBackground(Color.GREEN);
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);

        bouton = new JButton(" Effacer ");
        bouton.addActionListener(this);
        panneau.add(bouton);

        panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.CENTER);

        setVisible(true);
    }

    public void actionPerformed(ActionEvent evt) {
        if (evt.getActionCommand().equals("Au hasard"))
            System.out.println("Pression du bouton 'Au hasard'");
        else
            System.out.println("Pression du bouton 'Effacer'");
    }

    public static void main(String[] args) {
        new Truc05();
    }
}
```

Truc06.java - Légère variante du précédent : on distingue les sources peut leur référence (pointeur) au lieu de les distinguer par la chaîne de caractères qui leur est attachée. Notez que cela oblige à représenter les boutons par des variables d'instance au lieu de variables locales :

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

public class Truc06 extends JFrame implements ActionListener {

    private JButton boutonAuHasard;
    private JButton boutonEffacer;

    public Truc06() {
        super("Version 6");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        panneau.setBackground(Color.GREEN);
        getContentPane().add(panneau, BorderLayout.NORTH);

        boutonAuHasard = new JButton("Au hasard");
        boutonAuHasard.addActionListener(this);
        panneau.add(boutonAuHasard);

        boutonEffacer = new JButton(" Effacer ");
        boutonEffacer.addActionListener(this);
        panneau.add(boutonEffacer);

        panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.CENTER);

        setVisible(true);
    }
```

```
    public void actionPerformed(ActionEvent evt) {
        if (evt.getSource() == boutonAuHasard)
            System.out.println("Pression du bouton 'Au hasard'");
        else
            System.out.println("Pression du bouton 'Effacer'");
    }

    public static void main(String[] args) {
        new Truc06();
    }
}
```

Truc07.java - Une troisième manière de distinguer les sources consiste à créer un objet auditeur (ou même une classe, comme ici) spécifique à chaque source :

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

public class Truc07 extends JFrame {

    public Truc07() {
        super("Version 7");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        panneau.setBackground(Color.MAGENTA);
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(new AuditeurDeBoutonHasard());
        panneau.add(bouton);

        bouton = new JButton(" Effacer ");
        bouton.addActionListener(new AuditeurDeBoutonEffacer());
        panneau.add(bouton);

        panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.CENTER);

        setVisible(true);
    }

    class AuditeurDeBoutonHasard implements ActionListener {
        public void actionPerformed(ActionEvent evt) {
            System.out.println("Pression du bouton 'Au hasard'");
        }
    }

    class AuditeurDeBoutonEffacer implements ActionListener {
        public void actionPerformed(ActionEvent evt) {
            System.out.println("Pression du bouton 'Effacer'");
        }
    }

    public static void main(String[] args) {
        new Truc07();
    }
}
```

Truc08.java - Légère variation du précédent. Les classes anonymes permettent d'écrire les auditeurs spécifiques plus simplement :

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

public class Truc08 extends JFrame {
```

```
    public Truc08() {
        super("Version 8");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        panneau.setBackground(Color.MAGENTA);
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                System.out.println("Pression du bouton ’Au hasard’");
            }
        });
        panneau.add(bouton);

        bouton = new JButton(" Effacer ");
        bouton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                System.out.println("Pression du bouton ’Effacer’");
            }
        });
        panneau.add(bouton);

        panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.CENTER);

        setVisible(true);
    }

    public static void main(String[] args) {
        new Truc08();
    }
}
```

Truc09.java - Dans la foulée, détectons aussi les actions sur la case de fermeture. Première manière :

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

public class Truc09 extends JFrame {

    public Truc09() {
        super("Version 9");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        panneau.setBackground(Color.MAGENTA);
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                System.out.println("Pression du bouton ’Au hasard’");
            }
        });
        panneau.add(bouton);

        bouton = new JButton(" Effacer ");
        bouton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                System.out.println("Pression du bouton ’Effacer’");
            }
        });
        panneau.add(bouton);
```

```
        panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.CENTER);

        addWindowListener(new WindowListener() {
            public void windowClosing(WindowEvent e)  {
                System.exit(0);
            }
            public void windowOpened(WindowEvent e) {
            }
            public void windowClosed(WindowEvent e) {
            }
            public void windowActivated(WindowEvent e) {
            }
            public void windowDeactivated(WindowEvent e) {
            }
            public void windowIconified(WindowEvent e) {
            }
            public void windowDeiconified(WindowEvent e) {
            }
        });
        setVisible(true);
    }

    public static void main(String[] args) {
        new Truc09();
    }
}
```

Truc10.java - Légère variation du précédent : les *adaptateurs* simplifient l'implémentation des interfaces :

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

public class Truc10 extends JFrame {

    public Truc10() {
        super("Version 10");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        panneau.setBackground(Color.ORANGE);
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                System.out.println("Pression du bouton 'Au hasard'");
            }
        });
        panneau.add(bouton);

        bouton = new JButton(" Effacer ");
        bouton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                System.out.println("Pression du bouton 'Effacer'");
            }
        });
        panneau.add(bouton);

        panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.CENTER);

        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e)  {
                System.exit(0);
            }
        });
```

```
        setVisible(true);
    }

    public static void main(String[] args) {
        new Truc10();
    }
}
```

Truc11.java - Simplification. Si la fermeture du cadre ne doit provoquer rien d'autre que la terminaison du programme, la méthode **`setDefaultCloseOperation`** convient tout à fait :

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

public class Truc11 extends JFrame {

    public Truc11() {
        super("Version 11");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        panneau.setBackground(Color.ORANGE);
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                System.out.println("Pression du bouton 'Au hasard'");
            }
        });
        panneau.add(bouton);

        bouton = new JButton(" Effacer ");
        bouton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                System.out.println("Pression du bouton 'Effacer'");
            }
        });
        panneau.add(bouton);

        panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.CENTER);

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    public static void main(String[] args) {
        new Truc11();
    }
}
```

Truc12.java - Autre cas, on soutaite demander une confirmation avant de détruire le cadre (notez qu'il faut quand-même employer la méthode **`setDefaultCloseOperation`**) :

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

public class Truc12 extends JFrame {

    public Truc12() {
        super("Version 11");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        panneau.setBackground(Color.ORANGE);
```

```
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                System.out.println("Pression du bouton 'Au hasard'");
            }
        });
        panneau.add(bouton);

        bouton = new JButton(" Effacer ");
        bouton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                System.out.println("Pression du bouton 'Effacer'");
            }
        });
        panneau.add(bouton);

        panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.CENTER);

        setDefaultCloseOperation(DO_NOTHING_ON_CLOSE);
        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                if (JOptionPane.showConfirmDialog(null,
                        "Vous voulez vraiment quitter ?",
                        "Attention !",
                        JOptionPane.YES_NO_OPTION,
                        JOptionPane.QUESTION_MESSAGE) == JOptionPane.YES_OPTION)
                    System.exit(0);
            }
        });
        setVisible(true);
    }

    public static void main(String[] args) {
        new Truc12();
    }
}
```

Truc13.java - Créons – pour commencer avec le bouton *Au hasard* – et dessinons des points sur le panneau central. Voici la mauvaise façon de dessiner, « sans mémoire » (quand on tient la donnée on la dessine, puis on l'oublie) :

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

public class Truc13 extends JFrame implements ActionListener {

    private JPanel dessin;
    private Point pointPrec = null;

    public Truc13() {
        super("Version 13");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);

        bouton = new JButton(" Effacer ");
        bouton.addActionListener(this);
        panneau.add(bouton);
```

```
        dessin = new JPanel();
        dessin.setBackground(Color.WHITE);
        getContentPane().add(dessin, BorderLayout.CENTER);

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent evt) {
        if (evt.getActionCommand().equals("Au hasard"))
            nouveauPoint(new Point());
        else
            System.out.println("Pression du bouton 'Effacer'");
    }

    private void nouveauPoint(Point p) {
        if (pointPrec != null) {
            Graphics g = dessin.getGraphics();
            g.drawLine(pointPrec.getX(), pointPrec.getY(), p.getX(), p.getY());
        }
        pointPrec = p;
    }

    public static void main(String[] args) {
        new Truc13();
    }
}
```

Truc14.java - La bonne manière de dessiner consiste à mémoriser tout ce qui est à tracer et à effectuer *tout* le dessin dans la méthode `paint` du composant sur lequel on dessine (cela oblige à écrire pour lui une classe spécifique, généralement sous-classe de `JPanel`) :

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

public class Truc14 extends JFrame implements ActionListener {

    public static final int MAX_POINTS = 20;
    private Point[] liste = new Point[MAX_POINTS];
    private int nombre = 0;
    private JPanel dessin;

    public Truc14() {
        super("Version 14");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);

        bouton = new JButton(" Effacer ");
        bouton.addActionListener(this);
        panneau.add(bouton);

        dessin = new PanneauADessin();
        dessin.setBackground(Color.WHITE);
        getContentPane().add(dessin, BorderLayout.CENTER);

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent evt) {
        if (evt.getActionCommand().equals("Au hasard"))
```

```
            nouveauPoint(new Point());
        else
            System.out.println("Pression du bouton ’Effacer’");
    }

    private void nouveauPoint(Point p) {
        if (nombre < MAX_POINTS)
            liste[nombre++] = p;
        dessin.repaint();
    }

    private class PanneauADessin extends JPanel {
        public void paint(Graphics g) {
            super.paint(g);
            for (int i = 1; i < nombre; i++)
                g.drawLine(liste[i - 1].getX(), liste[i - 1].getY(),
                        liste[i].getX(), liste[i].getY());
        }
    }

    public static void main(String[] args) {
        new Truc14();
    }
}
```

Truc15.java - Une *collection* de points c’est plus commode qu’un tableau :

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

public class Truc15 extends JFrame implements ActionListener {
    private Collection<Point> liste = new Vector<Point>();
    private JPanel dessin;

    public Truc15() {
        super("Version 15");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);
        bouton = new JButton(" Effacer ");
        bouton.addActionListener(this);
        panneau.add(bouton);

        dessin = new PanneauADessin();
        dessin.setBackground(Color.WHITE);
        getContentPane().add(dessin, BorderLayout.CENTER);

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent evt) {
        if (evt.getActionCommand().equals("Au hasard"))
            nouveauPoint(new Point());
        else
            System.out.println("Pression du bouton ’Effacer’");
    }

    private void nouveauPoint(Point p) {
        liste.add(p);
        dessin.repaint();
    }
```

```
    private class PanneauADessin extends JPanel {
        public void paint(Graphics g) {
            super.paint(g);
            Iterator<Point> it = liste.iterator();
            if (it.hasNext()) {
                Point p0 = it.next();
                while (it.hasNext()) {
                    Point p1 = it.next();
                    g.drawLine(p0.getX(), p0.getY(), p1.getX(), p1.getY());
                    p0 = p1;
                }
            }
        }
    }

    public static void main(String[] args) {
        new Truc15();
    }
}
```

Truc16.java - Détecter la souris. On crée un auditeur d'événements souris (`MouseListener`) à l'aide d'une classe interne, ce qui permet de bénéficier du service rendu par un adaptateur (`MouseAdapter`) tout en conservant l'accès aux variables d'instance de la classe englobante :

```
public class Truc16 extends JFrame implements ActionListener {
    private Collection<Point> liste = new Vector<Point>();
    private JPanel dessin;

    public Truc16() {
        super("Version 16");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);

        bouton = new JButton(" Effacer ");
        bouton.addActionListener(this);
        panneau.add(bouton);

        dessin = new PanneauADessin();
        dessin.setBackground(Color.WHITE);
        getContentPane().add(dessin, BorderLayout.CENTER);
        dessin.addMouseListener(new AuditeurDeSouris());

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    private void nouveauPoint(Point p) {
        liste.add(p);
        dessin.repaint();
    }

    private class AuditeurDeSouris extends MouseAdapter {
        public void mousePressed(MouseEvent evt) {
            nouveauPoint(new Point(evt.getX(), evt.getY()));
        }
    }

    private class PanneauADessin extends JPanel {
        public void paint(Graphics g) {
            super.paint(g);
            Iterator<Point> i = liste.iterator();
```

```
            if (i.hasNext()) {
                Point p0 = i.next();
                while (i.hasNext()) {
                    Point p1 = i.next();
                    g.drawLine(p0.getX(), p0.getY(), p1.getX(), p1.getY());
                    p0 = p1;
                }
            }
        }
    }

    public void actionPerformed(ActionEvent evt) {
        if (evt.getActionCommand().equals("Au hasard"))
            nouveauPoint(new Point());
        else
            System.out.println("Pression du bouton 'Effacer'");
    }

    public static void main(String[] args) {
        new Truc16();
    }
}
```

Truc17.java - Petits arrangements. D'une part, la classe interne **`AuditeurDeSouris`** devient anonyme, cela rend notre programme encore plus simple. Ensuite, le bouton « Effacer » devient opérationnel :

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

public class Truc17 extends JFrame implements ActionListener {
    private Collection<Point> liste = new Vector<Point>();
    private JPanel dessin;

    public Truc17() {
        super("Version 17");
        setSize(Point.MAX_X, Point.MAX_Y);

        JPanel panneau = new JPanel();
        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);
        bouton = new JButton(" Effacer ");
        bouton.addActionListener(this);
        panneau.add(bouton);
        getContentPane().add(panneau, BorderLayout.NORTH);

        dessin = new PanneauADessin();
        dessin.setBackground(Color.WHITE);
        dessin.addMouseListener(new MouseAdapter() {
            public void mousePressed(MouseEvent evt) {
                nouveauPoint(new Point(evt.getX(), evt.getY()));
            }
        });
        getContentPane().add(dessin, BorderLayout.CENTER);

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent evt) {
        if (evt.getActionCommand().equals("Au hasard"))
            nouveauPoint(new Point());
        else {
            liste.clear();
            dessin.repaint();
```

```
        }
    }

    private void nouveauPoint(Point p) {
        liste.add(p);
        dessin.repaint();
    }

    private class PanneauADessin extends JPanel {
        public void paint(Graphics g) {
            super.paint(g);
            Iterator<Point> i = liste.iterator();
            if (i.hasNext()) {
                Point p0 = i.next();
                while (i.hasNext()) {
                    Point p1 = i.next();
                    g.drawLine(p0.getX(), p0.getY(), p1.getX(), p1.getY());
                    p0 = p1;
                }
            }
        }
    }

    public static void main(String[] args) {
        new Truc17();
    }
}
```

Truc18.java - Pour avoir deux vues de la même structure de données on introduit un panneau de texte à droite (à l'est) du cadre, dans lequel s'afficheront les coordonnées des points créés :

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

public class Truc18 extends JFrame implements ActionListener {
    private Collection<Point> liste = new Vector<Point>();
    private JPanel dessin;
    private JTextArea texte;

    public Truc18() {
        super("Version 18");
        setSize(Point.MAX_X + 80, Point.MAX_Y);

        JPanel panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);
        bouton = new JButton(" Effacer ");
        bouton.addActionListener(this);
        panneau.add(bouton);

        texte = new JTextArea(1, 15);
        texte.setFont(new Font("Monospaced", Font.PLAIN, 12));
        texte.setBackground(Color.yellow);
        getContentPane().add(texte, BorderLayout.EAST);

        dessin = new PanneauADessin();
        dessin.setBackground(Color.WHITE);
        dessin.addMouseListener(new MouseAdapter() {
            public void mousePressed(MouseEvent evt) {
                nouveauPoint(new Point(evt.getX(), evt.getY()));
            }
        });
        getContentPane().add(dessin, BorderLayout.CENTER);
```

```
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent evt) {
        if (evt.getActionCommand().equals("Au hasard"))
            nouveauPoint(new Point());
        else {
            liste.clear();
            dessin.repaint();
            texte.setText("");
        }
    }

    private void nouveauPoint(Point p) {
        liste.add(p);
        texte.append(liste.size() + " : " + p + "\n");
        dessin.repaint();
    }

    private class PanneauADessin extends JPanel {
        public void paint(Graphics g) {
            super.paint(g);
            Iterator<Point> i = liste.iterator();
            if (i.hasNext()) {
                Point p0 = i.next();
                while (i.hasNext()) {
                    Point p1 = i.next();
                    g.drawLine(p0.getX(), p0.getY(), p1.getX(), p1.getY());
                    p0 = p1;
                }
            }
        }
    }

    public static void main(String[] args) {
        new Truc18();
    }
}
```

Truc19.java - Petite modification, gros bénéfice : pour bien afficher des textes qui dépassent la taille de la zone de texte on insère des barres de défilement :

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

public class Truc19 extends JFrame implements ActionListener {
    private Collection<Point> liste = new Vector<Point>();
    private JPanel dessin;
    private JTextArea texte;

    public Truc19() {
        super("Version 19");
        setSize(Point.MAX_X + 80, Point.MAX_Y);

        JPanel panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);
        bouton = new JButton(" Effacer ");
        bouton.addActionListener(this);
        panneau.add(bouton);

        texte = new JTextArea(1, 17);
        texte.setFont(new Font("Monospaced", Font.PLAIN, 12));
```

```
        getContentPane().add(new JScrollPane(texte), BorderLayout.EAST);

        dessin = new PanneauADessin();
        dessin.setBackground(Color.WHITE);
        dessin.addMouseListener(new MouseAdapter() {
            public void mousePressed(MouseEvent evt) {
                nouveauPoint(new Point(evt.getX(), evt.getY()));
            }
        });
        getContentPane().add(dessin, BorderLayout.CENTER);

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent evt) {
        if (evt.getActionCommand().equals("Au hasard"))
            nouveauPoint(new Point());
        else {
            liste.clear();
            dessin.repaint();
            texte.setText("");
        }
    }

    private void nouveauPoint(Point p) {
        liste.add(p);
        texte.append(liste.size() + " : " + p + "\n");
        dessin.repaint();
    }

    private class PanneauADessin extends JPanel {
        public void paint(Graphics g) {
            super.paint(g);
            Iterator<Point> i = liste.iterator();
            if (i.hasNext()) {
                Point p0 = i.next();
                while (i.hasNext()) {
                    Point p1 = i.next();
                    g.drawLine(p0.getX(), p0.getY(), p1.getX(), p1.getY());
                    p0 = p1;
                }
            }
        }
    }

    public static void main(String[] args) {
        new Truc19();
    }
}
```

Truc20.java - Détection des événements traduisant le mouvement de la souris (`MouseMotionEvent`). Cela permettra de créer des lignes (presque) lisses :

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

public class Truc20 extends JFrame implements ActionListener {
    private Collection<Point> liste = new Vector<Point>();
    private JPanel dessin;
    private JTextArea texte;

    public Truc20() {
        super("Version 20");
        setSize(Point.MAX_X + 80, Point.MAX_Y);

        JPanel panneau = new JPanel();
```

```
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);
        bouton = new JButton(" Effacer ");
        bouton.addActionListener(this);
        panneau.add(bouton);

        texte = new JTextArea(1, 17);
        texte.setFont(new Font("Monospaced", Font.PLAIN, 12));
        getContentPane().add(new JScrollPane(texte), BorderLayout.EAST);

        dessin = new PanneauADessin();
        dessin.setBackground(Color.WHITE);
        dessin.addMouseListener(new MouseAdapter() {
            public void mousePressed(MouseEvent evt) {
                nouveauPoint(new Point(evt.getX(), evt.getY()));
            }
        });
        dessin.addMouseMotionListener(new MouseMotionAdapter() {
            public void mouseDragged(MouseEvent evt) {
                nouveauPoint(new Point(evt.getX(), evt.getY()));
            }
        });
        getContentPane().add(dessin, BorderLayout.CENTER);

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent evt) {
        if (evt.getActionCommand().equals("Au hasard"))
            nouveauPoint(new Point());
        else {
            liste.clear();
            dessin.repaint();
            texte.setText("");
        }
    }

    private void nouveauPoint(Point p) {
        liste.add(p);
        texte.append(liste.size() + " : " + p + "\n");
        dessin.repaint();
    }

    private class PanneauADessin extends JPanel {
        public void paint(Graphics g) {
            super.paint(g);
            Iterator<Point> i = liste.iterator();
            if (i.hasNext()) {
                Point p0 = i.next();
                while (i.hasNext()) {
                    Point p1 = i.next();
                    g.drawLine(p0.getX(), p0.getY(), p1.getX(), p1.getY());
                    p0 = p1;
                }
            }
        }
    }

    public static void main(String[] args) {
        new Truc20();
    }
}
```

Truc21.java - Introduction d'un menu pour choisir la couleur :

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

public class Truc21 extends JFrame implements ActionListener {
    private Collection<Point> liste = new Vector<Point>();
    private JPanel dessin;
    private JTextArea texte;
    private Color couleurVoulue = Color.BLACK;

    public Truc21() {
        super("Version 21");
        setSize(Point.MAX_X + 80, Point.MAX_Y);

        JPanel panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);
        bouton = new JButton(" Effacer ");
        bouton.addActionListener(this);
        panneau.add(bouton);

        texte = new JTextArea(1, 17);
        texte.setFont(new Font("Monospaced", Font.PLAIN, 12));
        getContentPane().add(new JScrollPane(texte), BorderLayout.EAST);

        dessin = new PanneauADessin();
        dessin.setBackground(Color.WHITE);
        dessin.addMouseListener(new MouseAdapter() {
            public void mousePressed(MouseEvent evt) {
                nouveauPoint(new Point(evt.getX(), evt.getY()));
            }
        });
        dessin.addMouseMotionListener(new MouseMotionAdapter() {
            public void mouseDragged(MouseEvent evt) {
                nouveauPoint(new Point(evt.getX(), evt.getY()));
            }
        });
        getContentPane().add(dessin, BorderLayout.CENTER);

        setJMenuBar(creerBarreDeMenus());

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    JMenuBar creerBarreDeMenus() {
        JMenuBar barre = new JMenuBar();

        JMenu menu = new JMenu("Couleur");
        barre.add(menu);

        JMenuItem item = new JMenuItem("Noir");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Rouge");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Vert");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Bleu");
        item.addActionListener(this);
```

```
            menu.add(item);

            return barre;
        }

        public void actionPerformed(ActionEvent evt) {
            if (evt.getActionCommand().equals("Noir")) {
                couleurVoulue = Color.BLACK;
                dessin.repaint();
            }
            else if (evt.getActionCommand().equals("Rouge")) {
                couleurVoulue = Color.RED;
                dessin.repaint();
            }
            else if (evt.getActionCommand().equals("Bleu")) {
                couleurVoulue = Color.BLUE;
                dessin.repaint();
            }
            else if (evt.getActionCommand().equals("Vert")) {
                couleurVoulue = Color.GREEN;
                dessin.repaint();
            }
            else if (evt.getActionCommand().equals("Au hasard"))
                nouveauPoint(new Point());
            else {
                liste.clear();
                dessin.repaint();
                texte.setText("");
            }
        }

        private void nouveauPoint(Point p) {
            liste.add(p);
            texte.append(liste.size() + " : " + p + "\n");
            dessin.repaint();
        }

        private class PanneauADessin extends JPanel {
            public void paint(Graphics g) {
                super.paint(g);

                Color couleurCourante = g.getColor();
                g.setColor(couleurVoulue);

                Iterator<Point> i = liste.iterator();
                if (i.hasNext()) {
                    Point p0 = i.next();
                    while (i.hasNext()) {
                        Point p1 = i.next();
                        g.drawLine(p0.getX(), p0.getY(), p1.getX(), p1.getY());
                        p0 = p1;
                    }
                }

                g.setColor(couleurCourante);
            }
        }

        public static void main(String[] args) {
            new Truc21();
        }
    }
```

Truc22.java - Pour faire bonne mesure, on ajoute un deuxième menu, permettant de choisir la couleur du fond. En même temps, on met une bordure en « creux » :

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

```
import java.awt.event.*;
import java.util.*;

public class Truc22 extends JFrame implements ActionListener  {
    private Collection<Point> liste = new Vector<Point>();
    private JPanel dessin;
    private JTextArea texte;
    private Color couleurVoulue = Color.BLACK;

    public Truc22() {
        super("Version 22");
        setSize(Point.MAX_X + 80, Point.MAX_Y);

        JPanel panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);
        bouton = new JButton(" Effacer ");
        bouton.addActionListener(this);
        panneau.add(bouton);

        texte = new JTextArea(1, 17);
        texte.setFont(new Font("Monospaced", Font.PLAIN, 12));

        getContentPane().add(new JScrollPane(texte), BorderLayout.EAST);

        dessin = new PanneauADessin();
        dessin.setBorder(BorderFactory.createBevelBorder(BevelBorder.LOWERED));
        dessin.setBackground(Color.WHITE);
        dessin.addMouseListener(new MouseAdapter() {
            public void mousePressed(MouseEvent evt) {
                nouveauPoint(new Point(evt.getX(), evt.getY()));
            }
        });
        dessin.addMouseMotionListener(new MouseMotionAdapter() {
            public void mouseDragged(MouseEvent evt) {
                nouveauPoint(new Point(evt.getX(), evt.getY()));
            }
        });
        getContentPane().add(dessin, BorderLayout.CENTER);

        setJMenuBar(creerBarreDeMenus());

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    JMenuBar creerBarreDeMenus() {
        JMenuBar barre = new JMenuBar();

        JMenu menu = new JMenu("Couleur du trait");
        barre.add(menu);

        JMenuItem item = new JMenuItem("Noir");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Rouge");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Vert");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Bleu");
        item.addActionListener(this);
```

```
        menu.add(item);

        menu = new JMenu("Couleur du panneau");
        barre.add(menu);

        item = new JMenuItem("Blanc");
        item.setActionCommand("Fond blanc");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Rouge");
        item.setActionCommand("Fond rouge");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Vert");
        item.setActionCommand("Fond vert");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Bleu");
        item.setActionCommand("Fond bleu");
        item.addActionListener(this);
        menu.add(item);

        return barre;
    }

    public void actionPerformed(ActionEvent evt) {
        if (evt.getActionCommand().equals("Noir")) {
            couleurVoulue = Color.BLACK;
            dessin.repaint();
        }
        else if (evt.getActionCommand().equals("Rouge")) {
            couleurVoulue = Color.RED;
            dessin.repaint();
        }
        else if (evt.getActionCommand().equals("Vert")) {
            couleurVoulue = Color.GREEN;
            dessin.repaint();
        }
        else if (evt.getActionCommand().equals("Bleu")) {
            couleurVoulue = Color.BLUE;
            dessin.repaint();
        }
        else if (evt.getActionCommand().equals("Fond blanc"))
            dessin.setBackground(Color.WHITE);
        else if (evt.getActionCommand().equals("Fond rouge"))
            dessin.setBackground(Color.RED);
        else if (evt.getActionCommand().equals("Fond vert"))
            dessin.setBackground(Color.GREEN);
        else if (evt.getActionCommand().equals("Fond bleu"))
            dessin.setBackground(Color.BLUE);
        else if (evt.getActionCommand().equals("Au hasard"))
            nouveauPoint(new Point());
        else {
            liste.clear();
            dessin.repaint();
            texte.setText("");
        }
    }

    private void nouveauPoint(Point p) {
        liste.add(p);
        texte.append(liste.size() + " : " + p + "\n");
        dessin.repaint();
    }

    private class PanneauADessin extends JPanel {
```

```
        public void paint(Graphics g) {
            super.paint(g);

            Color couleurCourante = g.getColor();
            g.setColor(couleurVoulue);

            Iterator<Point> i = liste.iterator();
            if (i.hasNext()) {
                Point p0 = i.next();
                while (i.hasNext()) {
                    Point p1 = i.next();
                    g.drawLine(p0.getX(), p0.getY(), p1.getX(), p1.getY());
                    p0 = p1;
                }
            }

            g.setColor(couleurCourante);
        }
    }

    public static void main(String[] args) {
        new Truc22();
    }
}
```

Truc23.java - Gadget final : un curseur pour éclaircir la couleur du fond du panneau :

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

public class Truc23 extends JFrame implements ActionListener {
    private Collection<Point> liste = new Vector<Point>();
    private JPanel dessin;
    private JTextArea texte;
    private Color couleuDuTrait = Color.BLACK;
    private Color couleurDuFond = Color.WHITE;
    private JSlider curseur;

    public Truc23() {
        super("Version 23");
        setSize(Point.MAX_X + 80, Point.MAX_Y);

        JPanel panneau = new JPanel();
        getContentPane().add(panneau, BorderLayout.NORTH);

        JButton bouton = new JButton("Au hasard");
        bouton.addActionListener(this);
        panneau.add(bouton);
        bouton = new JButton(" Effacer ");
        bouton.addActionListener(this);
        panneau.add(bouton);

        texte = new JTextArea(1, 20);
        texte.setFont(new Font("Monospaced", Font.PLAIN, 12));
        getContentPane().add(new JScrollPane(texte), BorderLayout.EAST);

        dessin = new PanneauADessin();
        dessin.setBorder(BorderFactory.createBevelBorder(BevelBorder.LOWERED));
        dessin.setBackground(Color.WHITE);
        dessin.addMouseListener(new MouseAdapter() {
            public void mousePressed(MouseEvent evt) {
                nouveauPoint(new Point(evt.getX(), evt.getY()));
            }
        });
        dessin.addMouseMotionListener(new MouseMotionAdapter() {
```

```
            public void mouseDragged(MouseEvent evt) {
                nouveauPoint(new Point(evt.getX(), evt.getY()));
            }
        });
        getContentPane().add(dessin, BorderLayout.CENTER);

        curseur = new JSlider(0, 255, 0);
        curseur.addChangeListener(new ChangeListener() {
            public void stateChanged(ChangeEvent e) {
                eclaircir();
            }
        });
        getContentPane().add(curseur, BorderLayout.SOUTH);

        setJMenuBar(creerBarreDeMenus());

        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setVisible(true);
    }

    private void eclaircir() {
        int val = curseur.getValue();
        int rouge = val;
        int vert  = val;
        int bleu  = val;
        if (couleurDuFond == Color.RED)
            rouge = 255;
        else if (couleurDuFond == Color.GREEN)
            vert = 255;
        else if (couleurDuFond == Color.BLUE)
            bleu = 255;
        dessin.setBackground(new Color(rouge, vert, bleu));
    }

    JMenuBar creerBarreDeMenus() {
        JMenuBar barre = new JMenuBar();

        JMenu menu = new JMenu("Couleur du trait");
        barre.add(menu);

        JMenuItem item = new JMenuItem("Noir");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Rouge");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Vert");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Bleu");
        item.addActionListener(this);
        menu.add(item);

        menu = new JMenu("Couleur du panneau");
        barre.add(menu);

        item = new JMenuItem("Blanc");
        item.setActionCommand("Fond blanc");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Rouge");
        item.setActionCommand("Fond rouge");
        item.addActionListener(this);
        menu.add(item);
```

```
        item = new JMenuItem("Vert");
        item.setActionCommand("Fond vert");
        item.addActionListener(this);
        menu.add(item);

        item = new JMenuItem("Bleu");
        item.setActionCommand("Fond bleu");
        item.addActionListener(this);
        menu.add(item);

        return barre;
    }

    public void actionPerformed(ActionEvent evt) {
        if (evt.getActionCommand().equals("Noir")) {
            couleuDuTrait = Color.BLACK;
            dessin.repaint();
        }
        else if (evt.getActionCommand().equals("Rouge")) {
            couleuDuTrait = Color.RED;
            dessin.repaint();
        }
        else if (evt.getActionCommand().equals("Vert")) {
            couleuDuTrait = Color.GREEN;
            dessin.repaint();
        }
        else if (evt.getActionCommand().equals("Bleu")) {
            couleuDuTrait = Color.BLUE;
            dessin.repaint();
        }
        else if (evt.getActionCommand().equals("Fond blanc")) {
            couleurDuFond = Color.WHITE;
            eclaircir();
        }
        else if (evt.getActionCommand().equals("Fond rouge")) {
            couleurDuFond = Color.RED;
            eclaircir();
        }
        else if (evt.getActionCommand().equals("Fond vert")) {
            couleurDuFond = Color.GREEN;
            eclaircir();
        }
        else if (evt.getActionCommand().equals("Fond bleu")) {
            couleurDuFond = Color.BLUE;
            eclaircir();
        }
        else if (evt.getActionCommand().equals("Au hasard"))
            nouveauPoint(new Point());
        else {
            liste.clear();
            dessin.repaint();
            texte.setText("");
        }
    }

    private void nouveauPoint(Point p) {
        liste.add(p);
        texte.append(liste.size() + " : " + p + "\n");
        dessin.repaint();
    }

    private class PanneauADessin extends JPanel {
        public void paint(Graphics g) {
            super.paint(g);

            Color couleurCourante = g.getColor();
            g.setColor(couleuDuTrait);

            Iterator<Point> i = liste.iterator();
```

```
            if (i.hasNext()) {
                Point p0 = i.next();
                while (i.hasNext()) {
                    Point p1 = i.next();
                    g.drawLine(p0.getX(), p0.getY(), p1.getX(), p1.getY());
                    p0 = p1;
                }
            }

            g.setColor(couleurCourante);
        }
    }

    public static void main(String[] args) {
        new Truc23();
    }
}
```
