Projet

et tutti quanti

Interfaces Graphiques

Un exemple complet

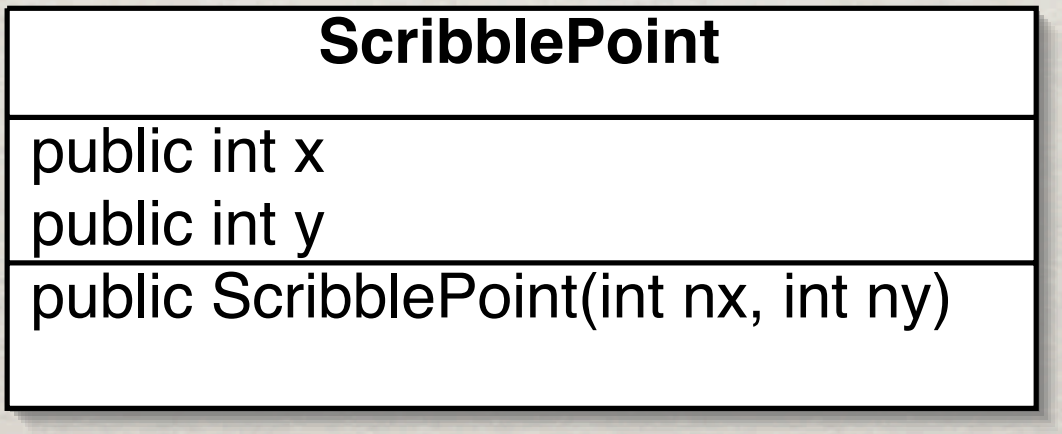
Objectif du jour : Scribble

- Application de dessin
- Gestion de l'historique des formes
- Multi-"document"

Préparation

- Hypothèse simplificatrice : une forme est une succession de lignes entre des points consécutifs
- Hypothèse simplificatrice : le dessin étant à la souris, on raisonne en pixels (pas de zoom)

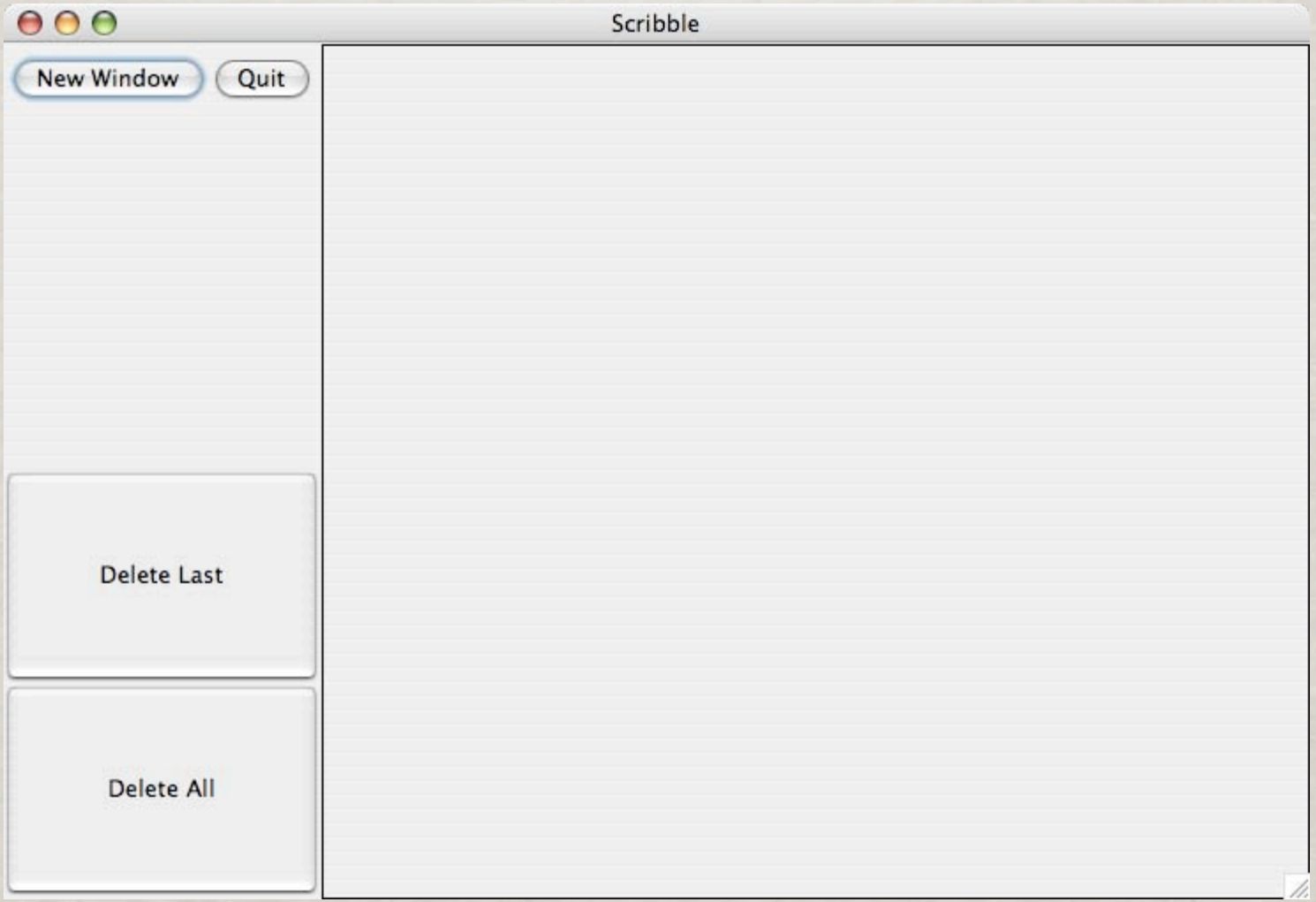
ScribblePoint



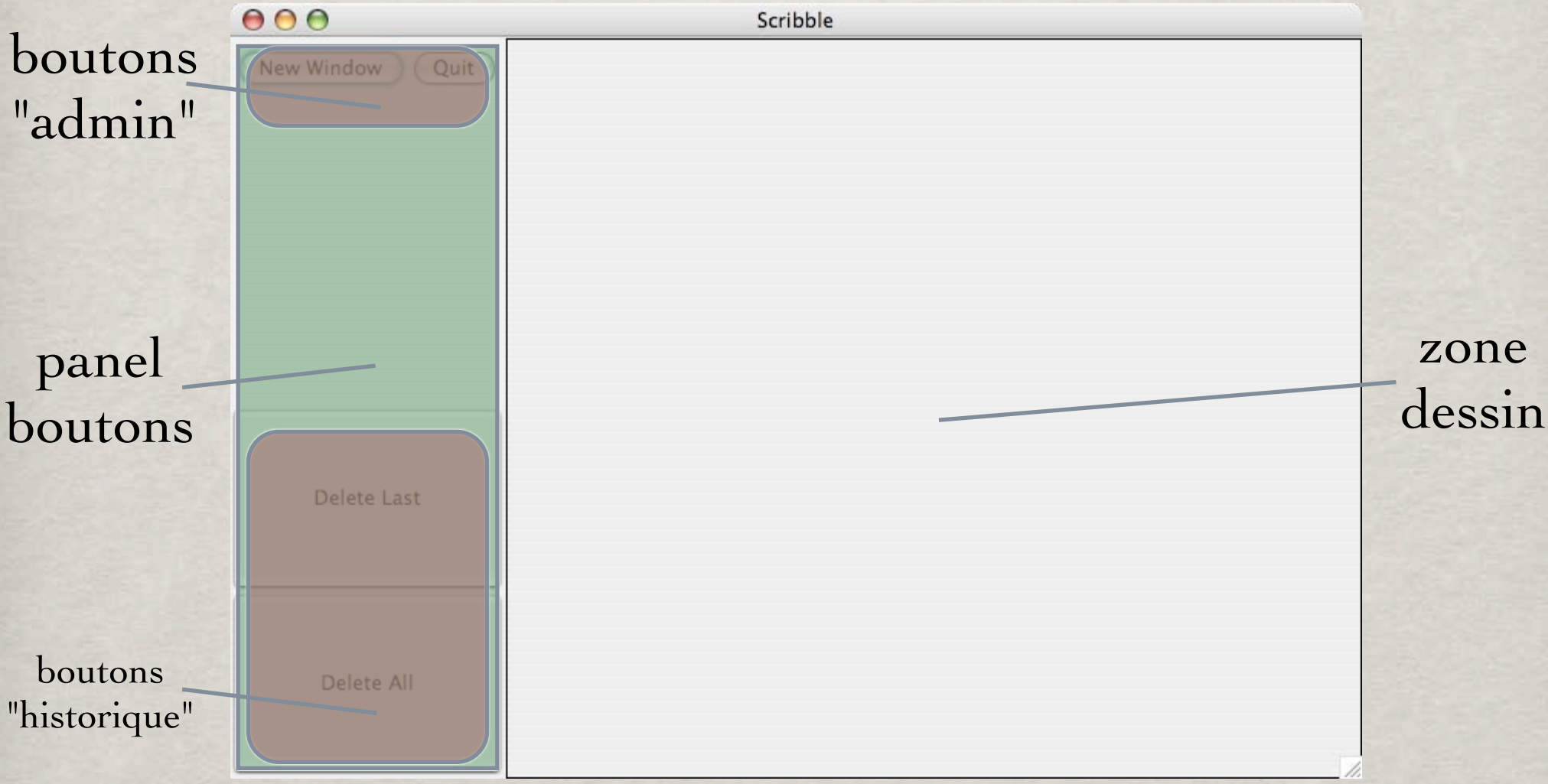
Forme

- Une forme est une succession de points.
- On peut donc résumer une forme à un Vector

Interface Graphique



Interface Graphique



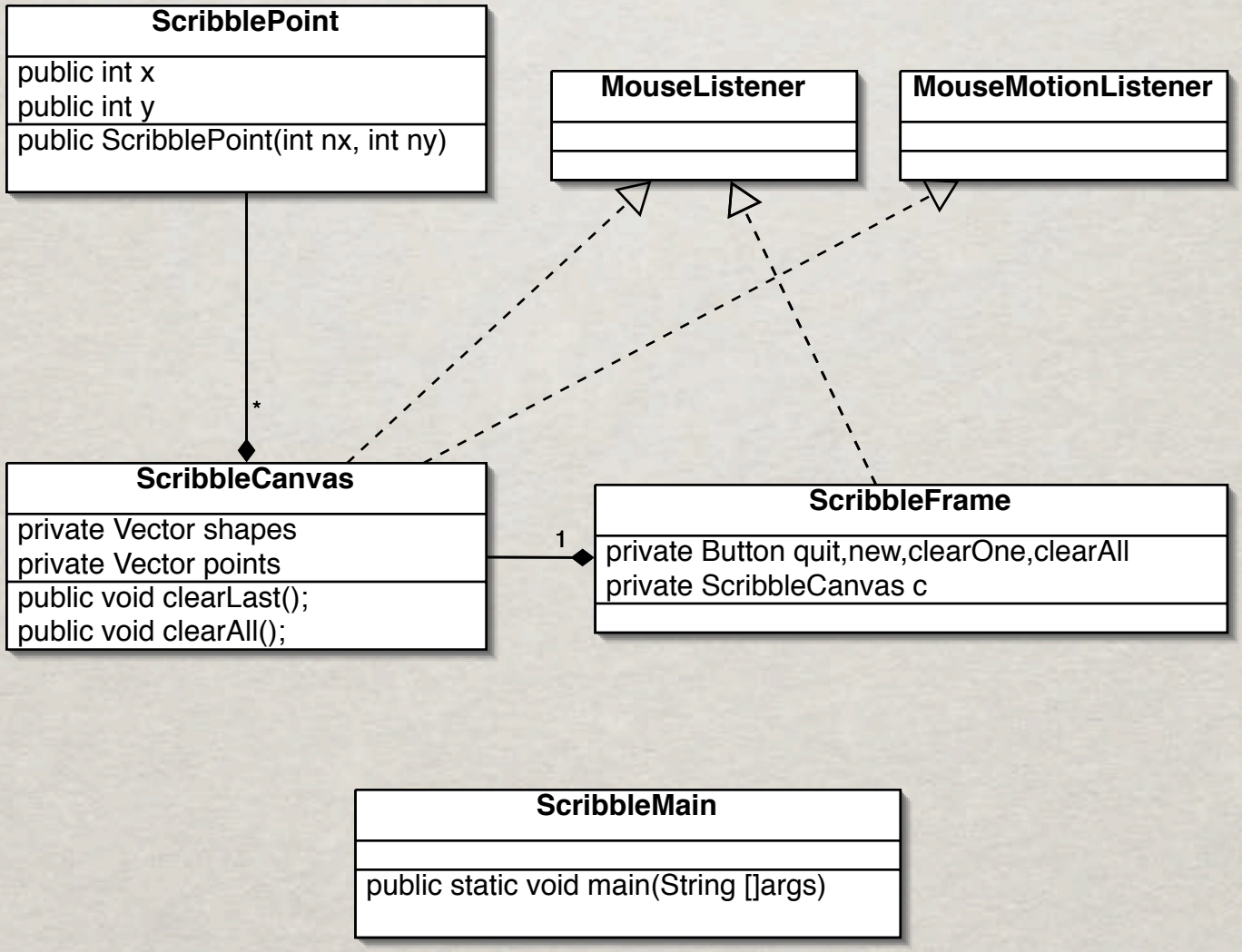
Classes nécessaires

- ScribblePoint
- Vector
- ScribbleCanvas (zone de dessin)
- ScribbleFrame
- ScribbleMain

Actions

- Gestion des boutons : MouseListener (quitter, nouveau, gestion de l'historique)
- Gestion du dessin : MouseListener (pour le clic initial et la fin de clic) et MouseMotionListener (pour le drag)

Diagramme de classes



Code

- Toujours commencer par les classes les plus petites et les moins dépendantes (les feuilles)
- Faire le plus simple possible, et complexifier dans un second temps

ScribblePoint

```
public class ScribblePoint {
    public int x, y;

    public ScribblePoint(int nx,
int ny) {
        x = nx; y = ny;
    }
}
```

ScribblePoint

- Se comporte comme une simple structure
- Pas de fioriture : on a besoin d'un constructeur, et c'est tout

Forme

- Une forme est un Vector de ScribblePoints
- points = new Vector();
- points.add(new ScribblePoint(x,y));

Canvas

- Une zone de dessin, en Java, est un Canvas (ou un JPanel en Swing)
- Par défaut, elle ne fait rien, donc on sous-classe
- On redéfinit la méthode "paint"
- On doit gérer les événements souris

ScribbleCanvas

```
public class ScribbleCanvas
extends Canvas
implements MouseListener,
MouseMotionListener {
    // les formes déjà dessinées
    private Vector shapes;
    // la forme en cours
    private Vector points;
```

LA MÉTHODE PAINT

- public void paint(Graphics g)
- Appelée à chaque fois que la zone change (couverte, redimensionnée, ou repaint())
- g est le "pinceau"

Dessiner les formes de l'historique

```
// shapes
for(i = 0 ; i < shapes.size() ; i++) {
    Vector currentShape = (Vector) shapes.get(i);
    for(j = 1 ; j < currentShape.size() ; j++) {
        ScribblePoint a = (ScribblePoint)
currentShape.get(j-1);
        ScribblePoint b = (ScribblePoint)
currentShape.get(j);
        g.drawLine(a.x, a.y, b.x, b.y);
    }
}
```

Dessiner la forme courante

```
if(points.size() > 1) {
    for(j = 1 ; j < points.size() ; j++) {
        ScribblePoint a = (ScribblePoint)
points.get(j-1);
        ScribblePoint b = (ScribblePoint)
points.get(j);
        g.drawLine(a.x, a.y, b.x, b.y);
    }
}
```

Création des formes

- Au premier clic (mousePressed), on met un premier point dans "points" (forme courante)
- Tant qu'on bouge la souris (mouseDragged), on ajoute les points au fur et à mesure dans "points"
- Quand on lâche la souris (mouseReleased), on ajoute la forme courante à la liste des formes

mousePressed

```
public void mousePressed(MouseEvent arg0)
{
	int curX = arg0.getX();
	int curY = arg0.getY();

	// histoire d'être sûrs
	points.removeAllElements();
	points.add(new ScribblePoint(curX,
curY));
}
```

mouseReleased

```
public void mouseReleased(MouseEvent arg0) {
    int curX = arg0.getX();
    int curY = arg0.getY();

    points.add(new ScribblePoint(curX, curY));
    // copy
    if(points.size() > 1)
        shapes.add(new Vector(points));
    points.removeAllElements();

    this.repaint();
}
```

mouseDragged

```
public void mouseDragged(MouseEvent arg0) {
    int curX = arg0.getX();
    int curY = arg0.getY();
    points.add(new ScribblePoint(curX, curY));

    this.repaint();
}
```

Historique

```
public void clearLastShape() {
    shapes.remove(shapes.size() - 1);
    this.repaint();
}

public void clearAllShapes() {
    shapes.removeAllElements();
    this.repaint();
}
```

Constructeur

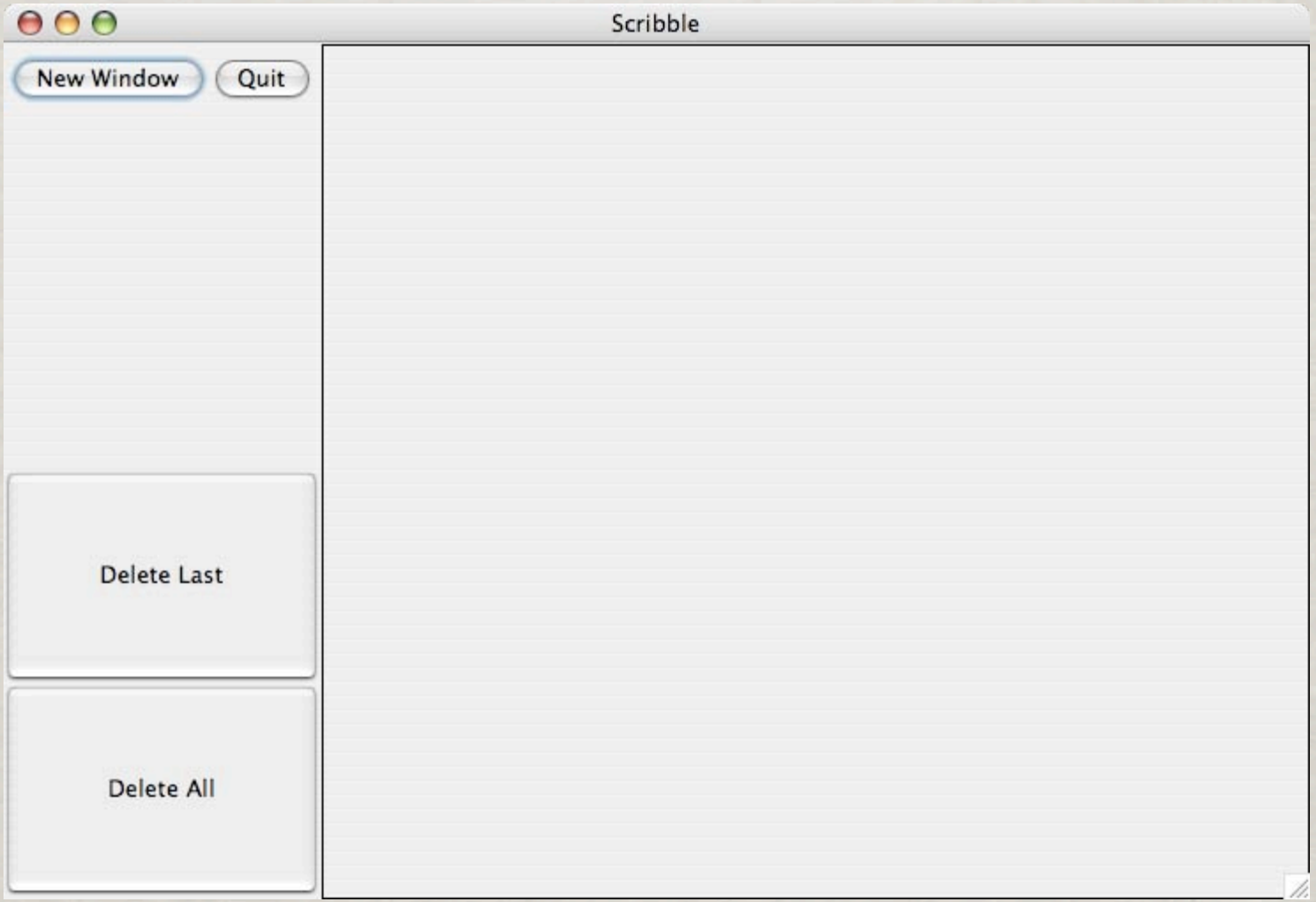
```
public ScribbleCanvas() {
    super();
    this.addMouseListener(this);
    this.addMouseMotionListener(this);

    shapes = new Vector();
    points = new Vector();
}
```

ScribbleFrame

- One window to rule them all
- Un panel à gauche qui contient deux panels
  - Un panel avec les boutons "admin" (new & quit)
  - Un panel avec les boutons "historique" (clearOne & clearAll)
- Fenêtre qui gère les clics

Interface Graphique



ScribbleWindow

```
public class ScribbleFrame extends
Frame implements MouseListener {

    private ScribbleCanvas c;
    private Button quit,
            newWindow,
            clear,
            clearAll;
```

Initialisation

```
public ScribbleFrame() {
    super("Scribble");
    this.setSize(300,300);
    this.setLayout(new BorderLayout());

    // canvas
    c = new ScribbleCanvas();
    this.add(c, BorderLayout.CENTER);
```

Initialisation (II)

```
// firstPanel : new & quit
Panel p1 = new Panel();
p1.setLayout(new FlowLayout());
newWindow =
    new Button("New Window");
newWindow.addMouseListener(this);
p1.add(newWindow);
quit = new Button("Quit");
quit.addMouseListener(this);
p1.add(quit);
```

Initialisation (III)

```
// secondPanel : back & clear all
Panel p2 = new Panel();
p2.setLayout(new GridLayout(2,1));

clear = new Button("Delete Last");
clear.addMouseListener(this);
p2.add(clear);

clearAll = new Button("Delete All");
clearAll.addMouseListener(this);
p2.add(clearAll);
```

Initialisation (IV)

```
        // bigger panel
        Panel gPane = new Panel();
        gPane.setLayout(new GridLayout(2,1));
        gPane.add(p1);
        gPane.add(p2);

        this.add(gPane, BorderLayout.WEST);

        // finally
        this.setVisible(true);
}
```

Gestion du clic

- quit -> fermer la fenêtre
- new -> créer une nouvelle fenêtre
- clear one -> effacer la dernière forme
- clear all -> effacer toutes les formes

mouseClicked

- getSource() vous donne l'objet sur lequel on a cliqué. Ça peut être n'importe quel objet pour lequel vous êtes un Listener
- instanceof permet de tester le type (la classe) de l'objet
- Une comparaison == ne peut se faire qu'entre objets du même type

mouseClicked

```
public void
mouseClicked(MouseEvent arg0) {
  Object clicked = arg0.getSource();
  if(clicked instanceof Button) {
    Button clBut = (Button) clicked;
```

mouseClicked

```
      if(clBut == quit) {
        this.dispose();
      } else if(clBut == newWindow) {
        new ScribbleFrame();
      } else if(clBut == clear) {
        c.clearLastShape();
      } else if(clBut == clearAll) {
        c.clearAllShapes();
      }
    } // end if (instance)
} // end mouseClicked
```

ScribbleMain

- La classe principale
- Son seul rôle est de créer le premier objet

ScribbleMain

```
public class ScribbleMain {
    public static void main(String[] args) {
        new ScribbleFrame();
    }
}
```

En vrai