Abstract Windowing Toolkit:

Modello Component - Container

Layout

Menu

Marco Ronchetti - ronchet@altavista.it

AWT

Marco Ronchetti - ronchet@altavista.it

AWT

Marco Ronchetti - ronchet@altavista.it

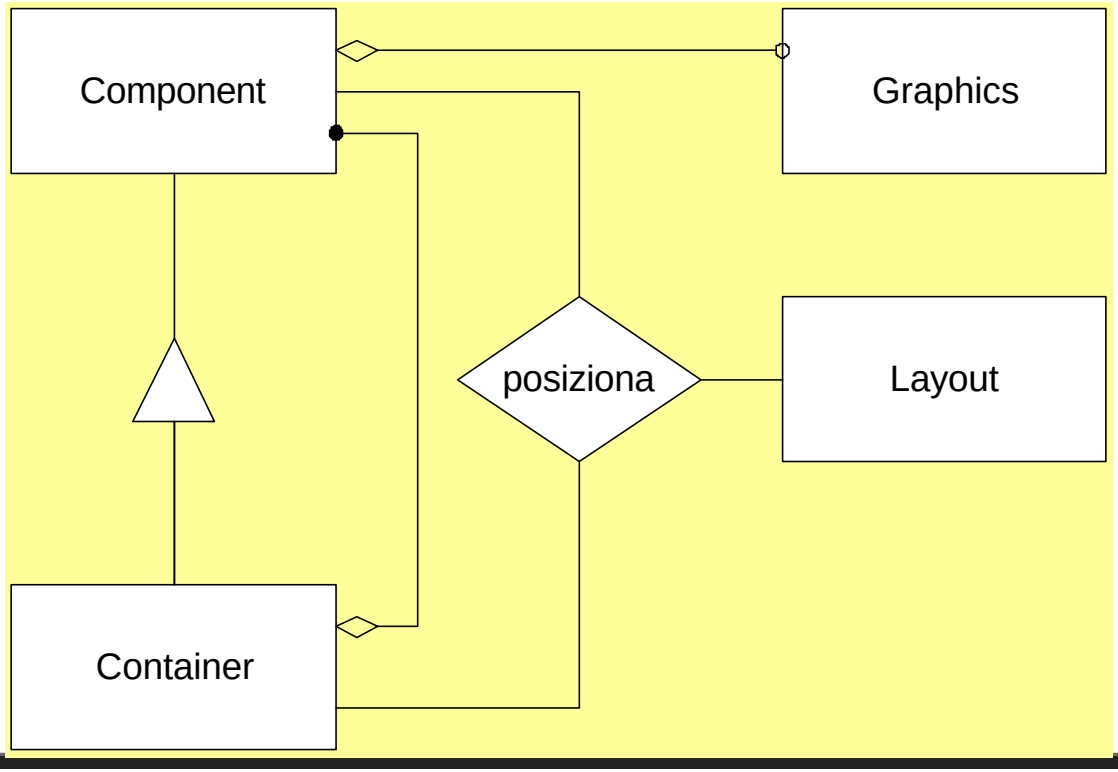
component - container - layout

Un Container contiene [0 o +] Components

Il Layout specifica come i Components sono disposti nel Container

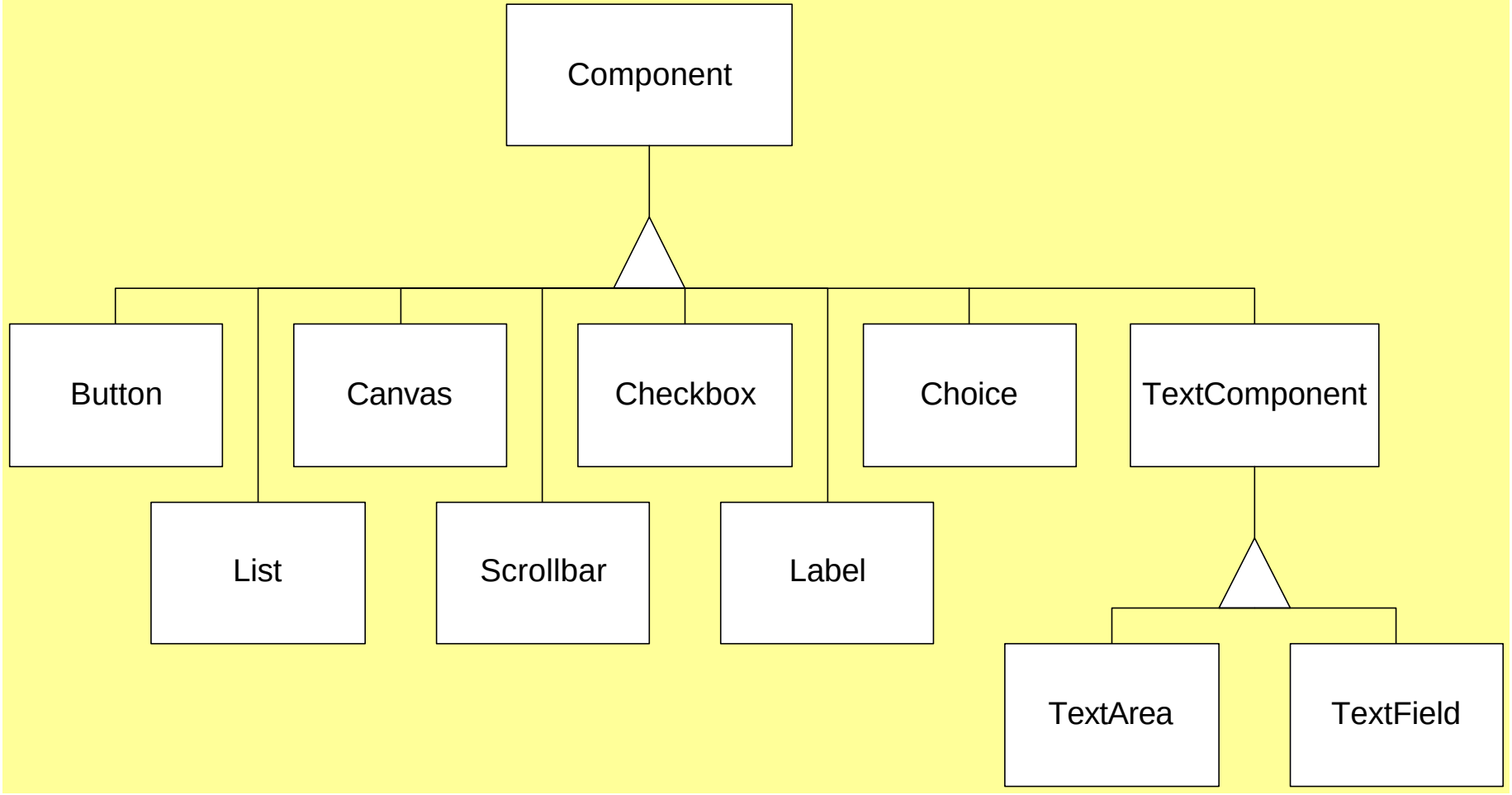
Un Container è un Component (quindi il contenimento è ricorsivo)

Un Component ha una Graphics associata



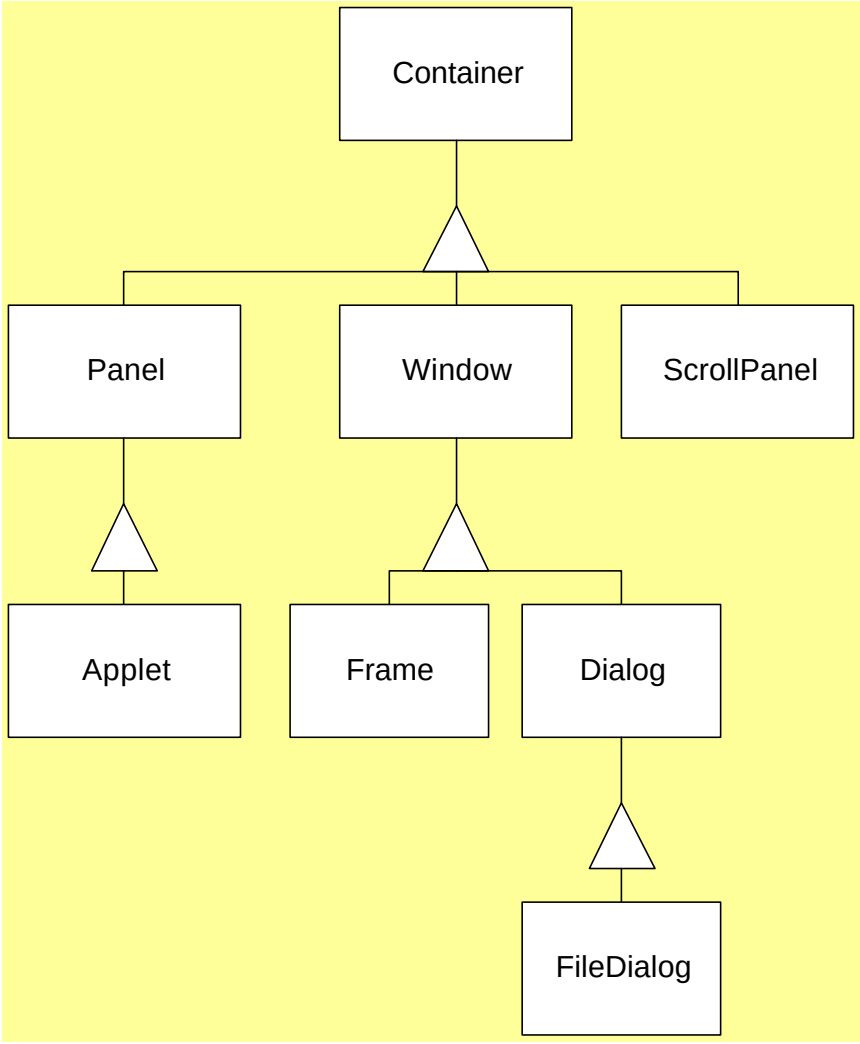
Marco Ronchetti - ronchet@altavista.it

component hierarchy



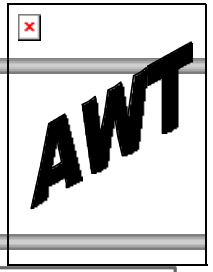
Marco Ronchetti - ronchet@altavista.it

container hierarchy



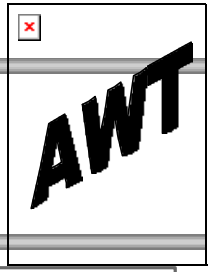
Marco Ronchetti - ronchet@altavista.it

Una finestra animata




```
import java.awt.*;
public class Applicazione {
   public static void main(String s[]) {
      Applicazione a=new Applicazione();
}
   Applicazione() {
    Frame f= new Frame ("Finestra crescente...");
    f.setSize(200,200);    // Dimensione della finestra
    f.setLocation(50,100); // Posizione della finestra
    f.show();
    for (int k=10;k<100;k+=10){
          //cicla su k incrementandolo di 1 al colpo
       // dormi per 500 millisecondi
       try {Thread.sleep(500);} catch (Exception e) {}
       f.setSize(200+k,200+k);
    }
    f.setTitle ("Finestra cresciuta");
   }
}
```

Una finestra animata – versione 2




```
import java.awt.*;
public class Applicazione extends Frame{
  public static void main(String s[]) {
        Applicazione a=new Applicazione();}
   Applicazione() {
       super("Finestra crescente...");
       this.setSize(200,200);    // Dimensione della finestra
       setLocation(50,100); // Posizione della finestra
        show();
       for (int k=10;k<100;k+=10){
         //cicla su k incrementandolo di 1 al colpo
        // dormi per 500 millisecondi
          try {Thread.sleep(500);} catch (Exception e) {}
          setSize(200+k,200+k);
       }
       setTitle ("Finestra cresciuta");
   }
}
```

Marco Ronchetti - ronchet@altavista.it

BorderLayout


```
import java.awt.*;
public class Applicazione {
   public static void main(String s[]) {Applicazione a=new Applicazione();}
   Applicazione() {
      Frame f= new Frame ("Finestra controllata da BORDER Layout");
      f.setSize(200,200);    // Dimensione della finestra
      f.setLocation(50,100); // Posizione della finestra
       Button north, south, east, west, center;
       north  = new Button("North"); east   = new Button("East");
       west   = new Button("West");  center = new Button("Center");
       south  = new Button("South");
       f.setLayout(new BorderLayout(2,2));
       f.add(north,BorderLayout.NORTH);
       f.add(south ,BorderLayout.SOUTH);
       f.add(east ,BorderLayout.EAST);
       f.add(west ,BorderLayout.WEST);
       f.add(center ,BorderLayout.CENTER);
       f.show();
   }
}
```

Marco Ronchetti - ronchet@altavista.it

FlowLayout


```
import java.awt.*;
public class Applicazione {
   public static void main(String s[]) {Applicazione a=new Applicazione();}
   Applicazione() {
       Frame f= new Frame ("Finestra controllata da FLOW Layout");
       f.setSize(200,200);    // Dimensione della finestra
       f.setLocation(50,100); // Posizione della finestra
      Button one, two, three, four, five, six;
      one   = new Button("one");       two   = new Button("two");
      three = new Button("three");     four  = new Button("four");
      five  = new Button("five");      six   = new Button("six");
      f.setLayout(new FlowLayout());
// attenzione all'ordine!
      f.add(one);       f.add(six);      f.add(five);
      f.add(two);       f.add(three);        f.add(four);
// aggiungo five per la seconda volta:
// la prima viene eliminata dalla sequenza!
      f.add(five);
      f.show();
   }
}
```

Marco Ronchetti - ronchet@altavista.it

GridLayout

```
import java.awt.*;
public class Applicazione {
  public static void main(String s[]) {
        Applicazione a=new Applicazione();}
   Applicazione() {
     Frame f= new Frame ("Finestra controllata da Border Layout");
     f.setSize(200,200);    // Dimensione della finestra
     f.setLocation(50,100); // Posizione della finestra
      f.setLayout(new GridLayout(3,4));
      Button b[]=new Button[10]; // VETTORE DI 10 BOTTONI
      for (int k=0; k<10; k++){
        b[k]=new Button();
        b[k].setLabel(Integer.toString(k));
        f.add(b[k]);
      }
      f.show();
   }
}
```

Integer.toString(int a) traduce l'intero a in una String

Marco Ronchetti - ronchet@altavista.it

CardLayout

Esercizio:
Esplorate i metodi della classe CardLayout

```
import java.awt.*;
public class Applicazione {
   public static void main(String s[]) {
          Applicazione a=new Applicazione();}
   Applicazione() {
     Frame f= new Frame ("Finestra controllata da BORDER Layout");
     f.setSize(200,200);    f.setLocation(50,100);
      CardLayout cl=new CardLayout();     f.setLayout(cl);
      Panel p[]=new Panel(5);
      Color c[]={Color.red,Color.orange,Color.green,Color.blue,Color.pink};
      for (int k=0;k<5; k++){
          Panel p[k]=new Panel();           p[k].setBackground(c[k]);
          f.add(Integer.toString(k),p[k]);
          f.show();
          while (true) { // ciclo infinito
              try {
                  Thread.sleep(500); // dormi per 500 millisecondi
              } catch (Exception e) {}
              cl.next(cardPanel);    // richiama il prossimo componente
          }
      }
}
```

il primo parametro è una stringa con la quale riferirsi al componente aggiunto

Marco Ronchetti - ronchet@altavista.it

Posizionamento assoluto: Null Layout


```
import java.awt.*;
public class Applicazione {
  public static void main(String s[]) {
     Applicazione a=new Applicazione()}
  Applicazione() {
     Frame f= new Frame (
        "Finestra controllata da BORDER Layout");
     f.setSize(200,200);
      f.setLocation(50,100);
      Button b=new Button(("Push me");
      f.setLayout(null);
      f.add(b);
// Dimensiono il Bottone
      b.setSize(25,75);
// Posiziono il bottone nella finestra
      b.setLocation(10,100);
      f.show();
  }
}
```

It's possible to set the layout manager to null: no layout control.

You might do this to position an object on the display at some absolute coordinates.

This is almost never the right approach.

Components might have different minimum sizes on different platforms, and your interface would not be very portable.

Marco Ronchetti - ronchet@altavista.it

menu hierarchy

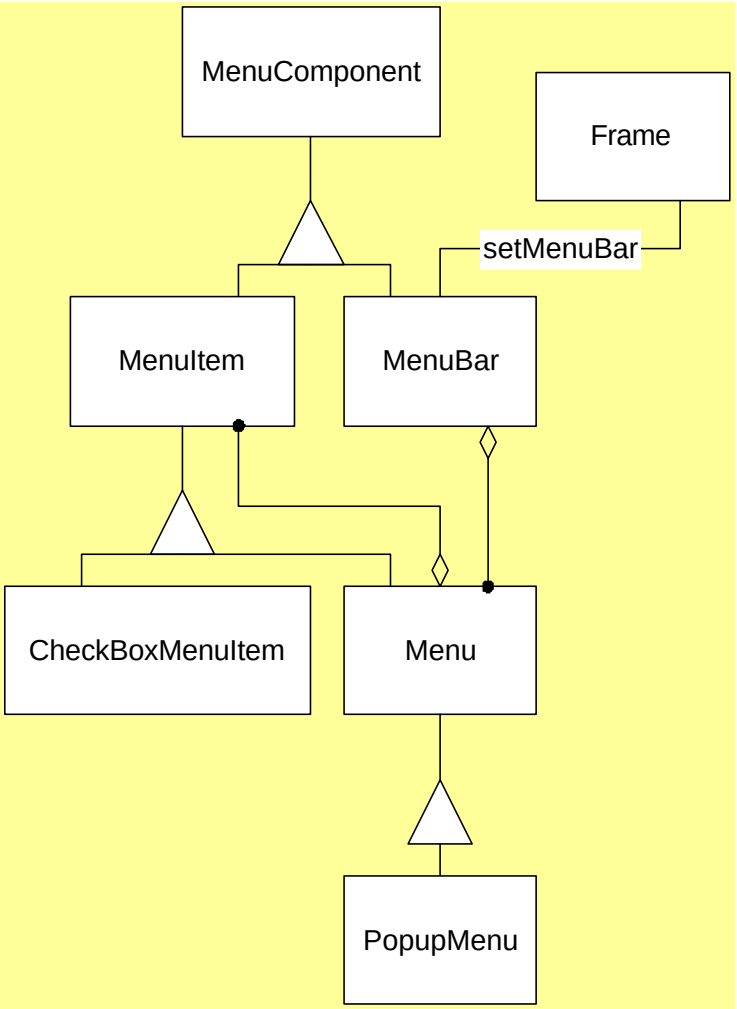
MenuComponent è una classe astratta

Ad un Frame può essere associato un Menubar

Un MenuBar contiene dei Menu

Un Menu contiene dei MenuItem

Un Menu è un MenuItem (risorsività del contenimento)



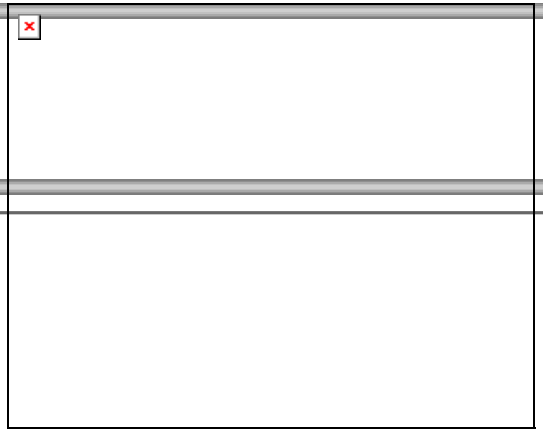
Marco Ronchetti - ronchet@altavista.it

menu hierarchy


```
import java.awt.*;

public class Applicazione extends Frame {

        MenuItem fileNew = new MenuItem("New");
        MenuItem fileOpen = new MenuItem("Open...");
        MenuItem fileSave = new MenuItem("Save");
        MenuItem fileExit = new MenuItem("Exit");
        MenuItem editUndo = new MenuItem("Undo");
        MenuItem editCut = new MenuItem("Cut");
        MenuItem editCopy = new MenuItem("Copy");
        MenuItem editPaste = new MenuItem("Paste");
        MenuItem helpContents = new MenuItem("Contents");
        MenuItem helpAbout = new MenuItem("About MenuFrame...");

        public Applicazione () {
                super("Menu example");

                MenuBar menubar = new MenuBar();
                Menu fileMenu = new Menu("File");
                Menu editMenu = new Menu("Edit");
                Menu helpMenu = new Menu("Help");
```

Marco Ronchetti - ronchet@altavista.it

menu hierarchy



```
fileMenu.add(fileNew);
fileMenu.add(fileOpen);
fileSave.setEnabled(false);
fileMenu.add(fileSave);
fileMenu.addSeparator();
fileMenu.add(fileExit);

editMenu.add(editUndo);
editMenu.addSeparator();
editMenu.add(editCut);
editMenu.add(editCopy);
editMenu.add(editPaste);

helpMenu.add(helpContents);
helpMenu.addSeparator();
helpMenu.add(helpAbout);

menubar.add(fileMenu);
menubar.add(editMenu);
menubar.add(helpMenu);
menubar.setHelpMenu(helpMenu);

setMenuBar(menubar);
```

```
    setSize(new Dimension(400, 300));

    Dimension screenSize =
    Toolkit.getDefaultToolkit().getScreenSize();
    Dimension mySize = getSize();

    setBounds((screenSize.width-mySize.width)/2,
      (screenSize.height-mySize.height)/2,
      mySize.width, mySize.height);

    show();
    }

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

Marco Ronchetti - ronchet@altavista.it

Abstract Windowing Toolkit:

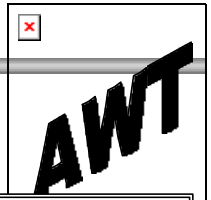
Eventi

Marco Ronchetti - ronchet@altavista.it

Demo events

Marco Ronchetti - ronchet@altavista.it

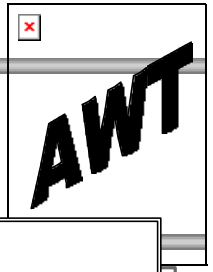
DemoEvents - 1




```
import java.awt.*;
class Main extends Frame {
  Main() {
    super("Event example");

// Menu bar with a menu and menu item.
   MenuBar mb = new MenuBar();
   Menu m = new Menu("Menu");
   m.add("MenuItem");
   m.add(new CheckboxMenuItem(
       "CheckboxMenuItem"));
   mb.add(m);
   setMenuBar(mb);
   setLayout(new GridLayout(0, 10));

// checkbox, grouped checkbox, label,
// text field, scrollbar, button
   add(new Checkbox("Checkbox"));
   add(new Checkbox("Checkbox", new
       CheckboxGroup(),true));
   add(new Label("label"));
   add(new TextField("TextField"));
   add(new Button("Button"));
   add(new MyCanvas());
```

```
// scrollbar
   Scrollbar sb = new Scrollbar(
       Scrollbar.HORIZONTAL);
   sb.setValues(50, 25, 0, 100);
   add(sb);
// choice
   Choice choice = new Choice();
   choice.addItem("Choice");
   choice.addItem("a choice item");
   add(choice);
// list
   List list = new List();
   list.addItem("List");
   list.addItem("a list item");
   add(list);
// text area
   TextArea textArea = new
TextArea("TextArea");
   textArea.resize(50, 25);
   add(textArea);
   pack();
   show();
}
```

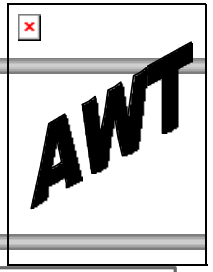
Demo events - 2




```
public boolean handleEvent(Event evt) {
 System.out.print(evt.target.getClass().getName());
 System.out.print(" ");
 switch (evt.id) {
  case Event.ACTION_EVENT:  System.out.print("ACTION_EVENT"); break;
  case Event.GOT_FOCUS:     System.out.print("GOT_FOCUS"); break;
  case Event.KEY_ACTION:    System.out.print("KEY_ACTION"); break;
  case Event.KEY_ACTION_RELEASE: System.out.print("KEY_ACTION_RELEASE"); break;
  case Event.KEY_PRESS:     System.out.print("KEY_PRESS"); break;
  case Event.KEY_RELEASE:   System.out.print("KEY_RELEASE"); break;
  case Event.LIST_DESELECT: System.out.print("LIST_DESELECT"); break;
  case Event.LIST_SELECT:   System.out.print("LIST_SELECT"); break;
  case Event.LOAD_FILE:      System.out.print("LOAD_FILE"); break;
  case Event.LOST_FOCUS:     System.out.print("LOST_FOCUS"); break;
  case Event.MOUSE_DOWN:     System.out.print("MOUSE_DOWN"); break;
  case Event.MOUSE_DRAG:     System.out.print("MOUSE_DRAG"); break;
  case Event.MOUSE_ENTER:    System.out.print("MOUSE_ENTER"); break;
  case Event.MOUSE_EXIT:     System.out.print("MOUSE_EXIT"); break;
  case Event.MOUSE_MOVE:     System.out.print("MOUSE_MOVE"); break;
  case Event.MOUSE_UP:       System.out.print("MOUSE_UP"); break;
  case Event.SAVE_FILE:      System.out.print("SAVE_FILE"); break;
```

Marco Ronchetti - ronchet@altavista.it

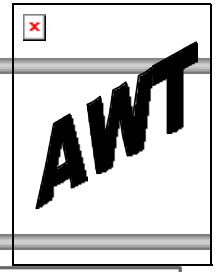
Demo events - 3




```
case Event.SCROLL_ABSOLUTE:  System.out.print("SCROLL_ABSOLUTE"); break;
  case Event.SCROLL_LINE_DOWN: System.out.print("SCROLL_LINE_DOWN"); break;
  case Event.SCROLL_LINE_UP:   System.out.print("SCROLL_LINE_UP"); break;
  case Event.SCROLL_PAGE_DOWN: System.out.print("SCROLL_PAGE_DOWN"); break;
  case Event.SCROLL_PAGE_UP:   System.out.print("SCROLL_PAGE_UP"); break;
  case Event.WINDOW_DEICONIFY: System.out.print("WINDOW_DEICONIFY"); break;
  case Event.WINDOW_DESTROY: System.out.print("WINDOW_DESTROY"); break;
  case Event.WINDOW_EXPOSE:  System.out.print("WINDOW_EXPOSE"); break;
  case Event.WINDOW_ICONIFY: System.out.print("WINDOW_ICONIFY"); break;
  case Event.WINDOW_MOVED:   System.out.print("WINDOW_MOVED"); break;
  }
System.out.print(" ("+evt.x+" "+evt.y+")  w("+evt.when+")");
System.out.print(" k("+(char)evt.key+") m("+evt.modifiers+")");
System.out.print(" c("+evt.clickCount+") a("+evt.arg+")");
System.out.println();
return false;
}
```

Marco Ronchetti - ronchet@altavista.it

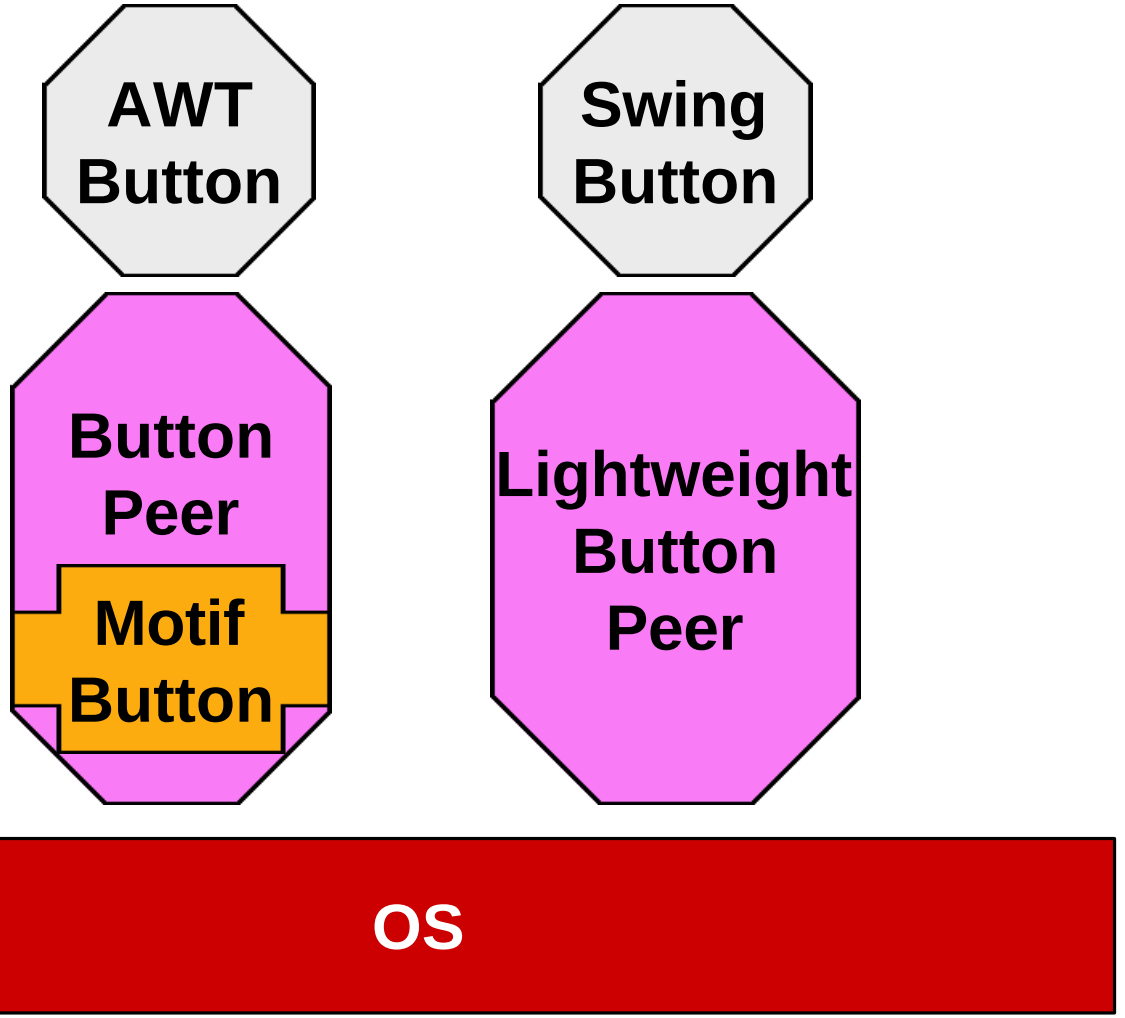
Demo events - 4




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

class MyCanvas extends Canvas {
    public void paint(Graphics g) {
        FontMetrics fm = g.getFontMetrics();
        g.drawString("Canvas",
              (size().width-fm.stringWidth("Canvas"))/2,
              (size().height-fm.getAscent())/2);
    }
}
```

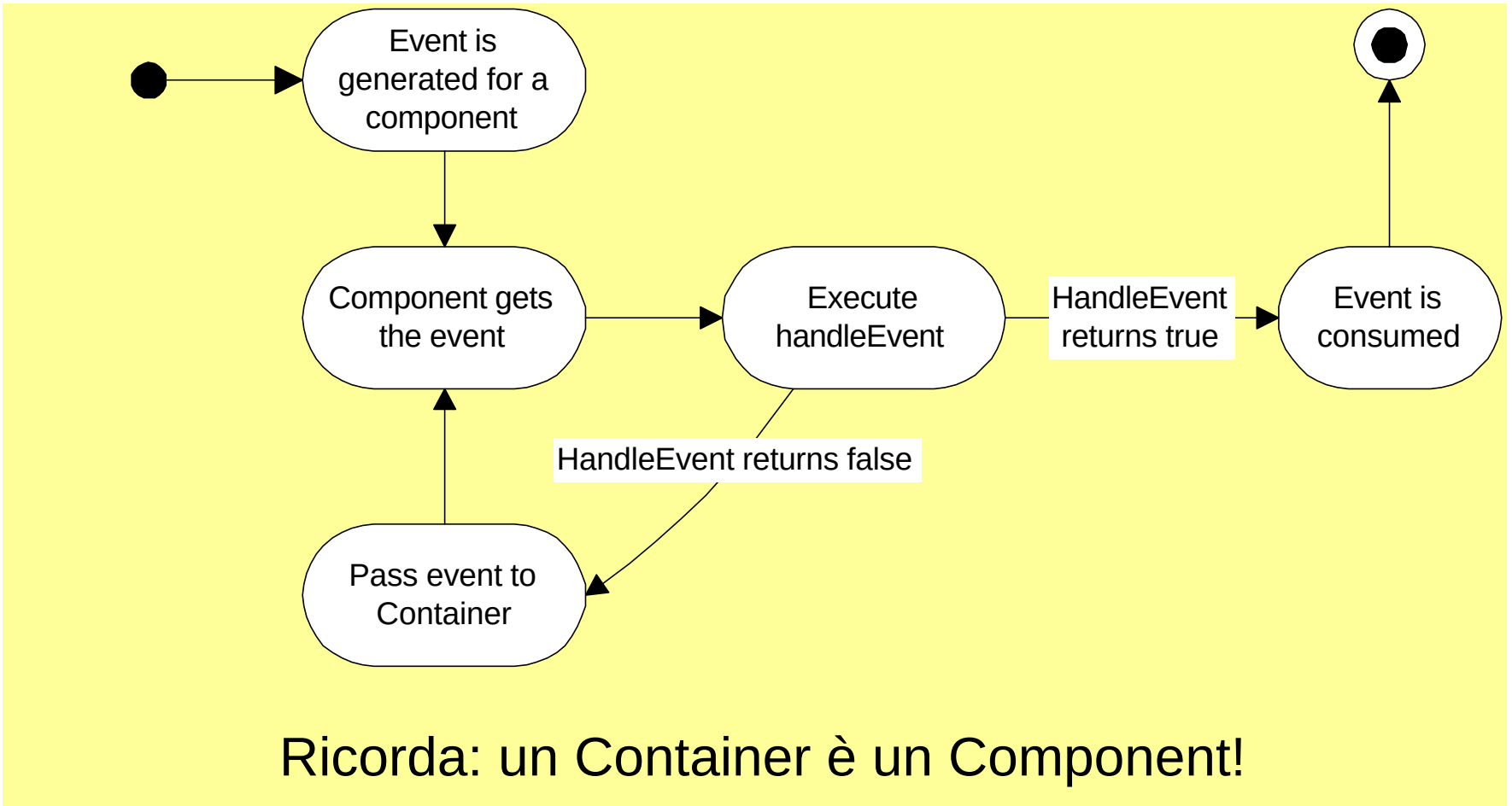
Marco Ronchetti - ronchet@altavista.it

Achieving OS Independence



Marco Ronchetti - ronchet@altavista.it

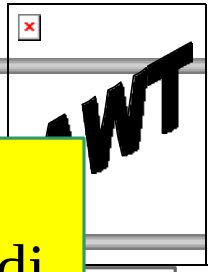
Modello 1.0



Ricorda: un Container è un Component!

Marco Ronchetti - ronchet@altavista.it

handleEvent()



Questo codice fa parte del sorgente di AWT

```
public boolean handleEvent(Event evt) {
    switch (evt.id) {
        case Event.MOUSE_ENTER:  return mouseEnter(evt, evt.x, evt.y);
        case Event.MOUSE_EXIT:   return mouseExit(evt, evt.x, evt.y);
        case Event.MOUSE_MOVE:   return mouseMove(evt, evt.x, evt.y);
        case Event.MOUSE_DOWN:   return mouseDown(evt, evt.x, evt.y);
        case Event.MOUSE_DRAG:   return mouseDrag(evt, evt.x, evt.y);
        case Event.MOUSE_UP:     return mouseUp(evt, evt.x, evt.y);
        case Event.KEY_PRESS:
        case Event.KEY_ACTION:   return keyDown(evt, evt.key);
        case Event.KEY_RELEASE:
        case Event.KEY_ACTION_RELEASE: return keyUp(evt, evt.key);
        case Event.ACTION_EVENT: return action(evt, evt.arg);
        case Event.GOT_FOCUS:    return gotFocus(evt, evt.arg);
        case Event.LOST_FOCUS:   return lostFocus(evt, evt.arg);
    }
    return false;
}
```

Event (1.0)

Questo codice fa parte del sorgente di AWT

```
/* Modifier constants */
public static final int SHIFT_MASK      = 1 << 0;
public static final int CTRL_MASK       = 1 << 1;
public static final int META_MASK       = 1 << 2;
ublic static final int ALT_MASK     = 1 << 3;

/* Action keys */
public static final int HOME     = 1000;
ublic static final int END      = 1001;
public static final int PGUP     = 1002; ...
public static final int PAUSE       = 1024;
public static final int INSERT      = 1025;

/* Non-action keys */
public static final int ENTER       = '\n';
public static final int BACK_SPACE  = '\b';
public static final int TAB         = '\t';
public static final int ESCAPE      = 27;
public static final int DELETE      = 127;
```

Marco Ronchetti - ronchet@altavista.it

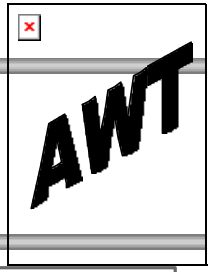
Event (1.0)

Questo codice fa parte del sorgente di AWT

```
/* Base for all window events. */
private static final int WINDOW_EVENT    = 200;
public static final int WINDOW_DESTROY   = 1 + WINDOW_EVENT;
public static final int WINDOW_EXPOSE    = 2 + WINDOW_EVENT;
public static final int WINDOW_ICONIFY   = 3 + WINDOW_EVENT;
 public static final int WINDOW_DEICONIFY    = 4 + WINDOW_EVENT;
public static final int WINDOW_MOVED = 5 + WINDOW_EVENT;

/* Base for all keyboard events. */
private static final int KEY_EVENT       = 400;
public static final int KEY_PRESS        = 1 + KEY_EVENT;
public static final int KEY_RELEASE      = 2 + KEY_EVENT;
public static final int KEY_ACTION       = 3 + KEY_EVENT;
public static final int KEY_ACTION_RELEASE   = 4 + KEY_EVENT;

 /* Base for all mouse events. */
private static final int MOUSE_EVENT     = 500;
public static final int MOUSE_DOWN       = 1 + MOUSE_EVENT;
public static final int MOUSE_UP         = 2 + MOUSE_EVENT;
public static final int MOUSE_MOVE       = 3 + MOUSE_EVENT;
public static final int MOUSE_ENTER      = 4 + MOUSE_EVENT;
public static final int MOUSE_EXIT       = 5 + MOUSE_EVENT;
public static final int MOUSE_DRAG       = 6 + MOUSE_EVENT;
```

Marco Ronchetti - ronchet@altavista.it

Event (1.0)

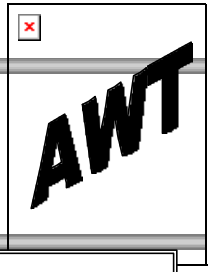



```
/* Scrolling events */   private static final int SCROLL_EVENT = 600;
public static final int SCROLL_LINE_UP   = 1 + SCROLL_EVENT;
public static final int SCROLL_LINE_DOWN = 2 + SCROLL_EVENT;
public static final int SCROLL_PAGE_UP   = 3 + SCROLL_EVENT;
public static final int SCROLL_PAGE_DOWN = 4 + SCROLL_EVENT;
public static final int SCROLL_ABSOLUTE  = 5 + SCROLL_EVENT;
public static final int SCROLL_BEGIN     = 6 + SCROLL_EVENT;
public static final int SCROLL_END       = 7 + SCROLL_EVENT;

/* List Events */   private static final int LIST_EVENT = 700;
public static final int LIST_SELECT      = 1 + LIST_EVENT;
public static final int LIST_DESELECT        = 2 + LIST_EVENT;

/* Misc Event */   private static final int MISC_EVENT       = 1000;
//This event indicates that the user wants some action to occur public static final int
 ACTION_EVENT    = 1 + MISC_EVENT;
public static final int LOAD_FILE        = 2 + MISC_EVENT;
public static final int SAVE_FILE        = 3 + MISC_EVENT;
public static final int GOT_FOCUS        = 4 + MISC_EVENT;
public static final int LOST_FOCUS   = 5 + MISC_EVENT;
```

Questo codice fa parte del sorgente di AWT

Marco Ronchetti - ronchet@altavista.it

Dealing with events-1.0




```
import java.awt.*;
import java.awt.event.*;
class Applicazione extends Frame  {
 int lastx, lasty;
 Graphics g;
 Button clear_button;
 public static void main(String s[]) {
       new Applicazione();}
 Applicazione() {
      setSize(200,200);
      clear_button = new Button("Clear");
      add(clear_button,BorderLayout.NORTH);
      show();
      g=getGraphics(); //dopo la show!
  }
/** Respond to mouse clicks */
 public boolean mouseDown(
        Event e, int x, int y) {
     lastx = x; lasty = y;
     return true;  }
/** Respond to mouse drags */
 public boolean mouseDrag(
        Event e, int x, int y) {
   g.setColor(Color.black);
    g.drawLine(lastx, lasty, x, y);
    lastx = x; lasty = y;
    return true;  }
```

```
/** Respond to key presses */
public boolean keyDown(
        Event e, int key) {
  if ((e.id == Event.KEY_PRESS) &&
            (key == 'c')) {
        clear();
        return true;
  }  else return false;
}
/** Respond to Button clicks */
public boolean action(
        Event e, Object arg) {
  if (e.target == clear_button) {
        clear();
        return true;
  } else return false;
}
/** convenience method to erase the
    scribble */
public void clear() {
  g.setColor(this.getBackground());
  g.fillRect(0, 0, bounds().width,
        bounds().height);
}
}
```

Marco Ronchetti - ronchet@altavista.it

Eventi 1.1

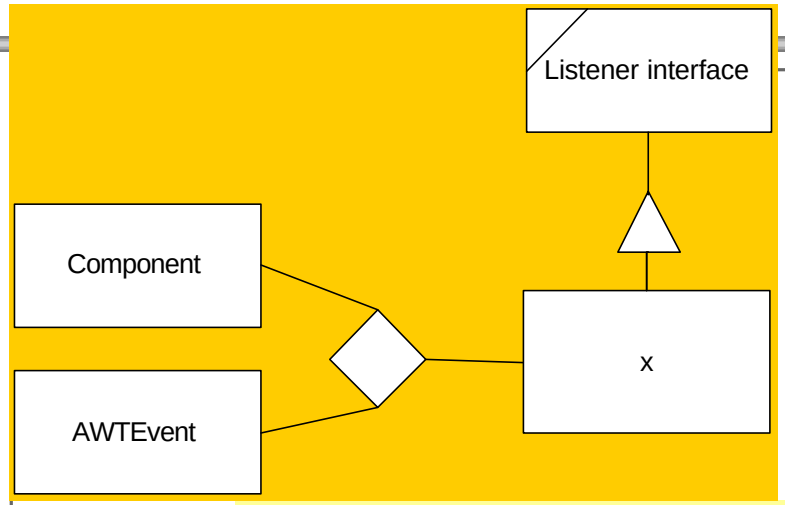
Nel modello di Java 1.0 a ciascuna componente vengono notificati gli eventi che la riguardano, ed e' poi sua responsabilità gestirli (eventualmente scalandoli nella gerarchia di contenimento).

Il modello di Java 1.1 invece "centralizza" la gestione degli eventi facendo riferimento a delle classi "Listener", presso le quali chi genera eventi deve registrarsi. La gestione dell'evento e' responsabilità del listener.

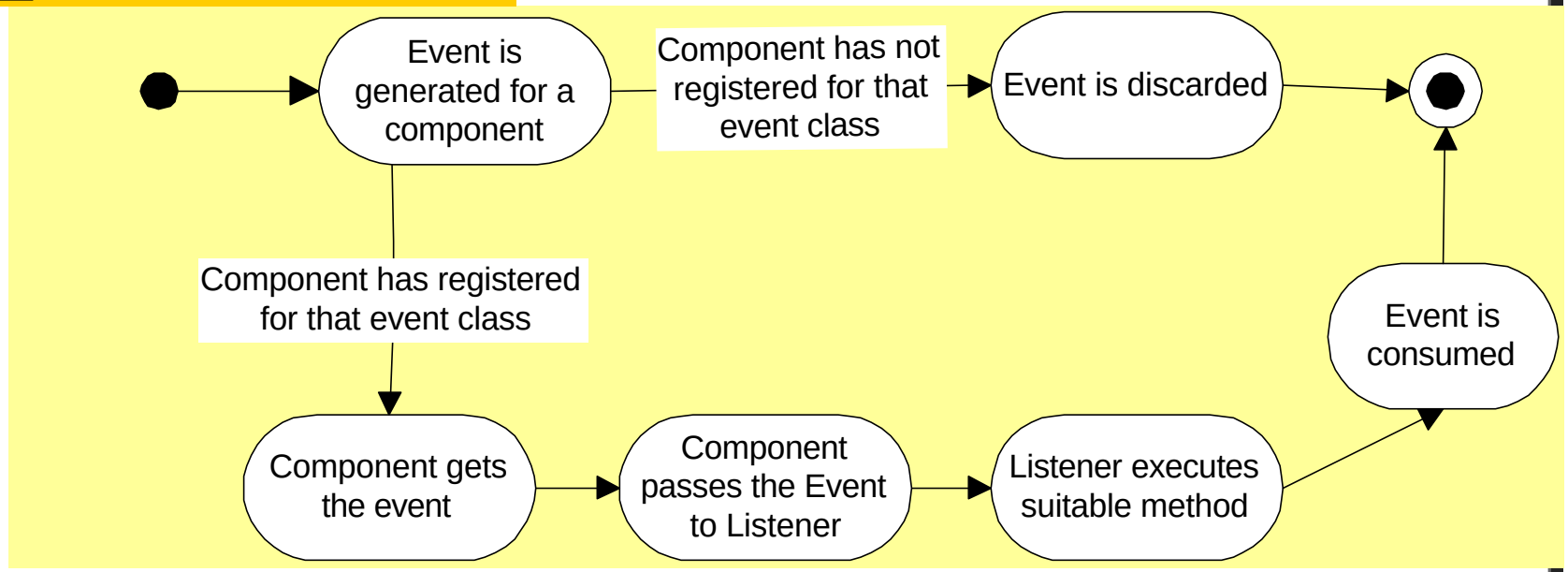
A volte, per semplicità, e' possibile unificare il generatore ed il gestore degli eventi. Anche in questo caso comunque si ha un vantaggio: la classe risulta chiaramente firmata come "implements Xlistener".

Marco Ronchetti - ronchet@altavista.it

Modello 1.1



Container non entra in gioco



Marco Ronchetti - ronchet@altavista.it

AWT Components and the Java 1.1 Events They Generate

Component	Generated Events	Meaning
Button	ActionEvent	User clicked on the button
Checkbox	ItemEvent	User selected or deselected an item
CheckboxMenuItem	ItemEvent	User selected or deselected an item
Choice	ItemEvent	User selected or deselected an item
Component	ComponentEvent	Component moved, resized, hidden, or shown
	FocusEvent	Component gained or lost focus
	KeyEvent	User pressed or released a key
	MouseEvent	User pressed or released mouse button, mouse entered or exited component, or user moved or dragged mouse. Note: MouseEvent has two corresponding listeners, MouseListener and MouseMotionListener.
Container	ContainerEvent	Component added to or removed from container
List	ActionEvent	User double-clicked on list item
	ItemEvent	User selected or deselected an item
MenuItem	ActionEvent	User selected a menu item
Scrollbar	AdjustmentEvent	User moved the scrollbar
TextComponent	TextEvent	User changed text
TextField	ActionEvent	User finished editing text
Window	WindowEvent	Window opened, closed, iconified, deiconified, or close requested

Marco Ronchetti - ronchet@altavista.it

1.1 Event Types, Listeners, and Listener Methods

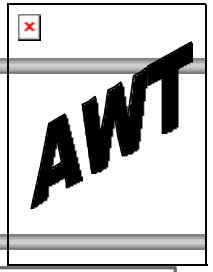
Event Class	Listener Interface	Listener Methods
ActionEvent	ActionListener	actionPerformed()
AdjustmentEvent	AdjustmentListener	adjustmentValueChanged()
ComponentEvent	ComponentListener	componentHidden()
		componentMoved()
		componentResized()
		componentShown()
ContainerEvent	ContainerListener	componentAdded()
		componentRemoved()
FocusEvent	FocusListener	focusGained()
		focusLost()
ItemEvent	ItemListener	itemStateChanged()
KeyEvent	KeyListener	keyPressed()
		keyReleased()
		keyTyped()
MouseEvent	MouseListener	mouseClicked()
		mouseEntered()
		mouseExited()
		mousePressed()
	MouseMotionListener	mouseDragged()
		mouseMoved()
TextEvent	TextListener	textValueChanged()
WindowEvent	WindowListener	windowActivated()
		windowClosed()
		windowClosing()
		windowDeactivated()
		windowDeiconified()
		windowIconified()
		windowOpened()

Marco Ronchetti - ronchet@altavista.it

Una finestra che risponde -1


```
import java.awt.*;
import java.awt.event.*;
public class Applicazione {
   public static void main(String s[]) {
       Applicazione a=new Applicazione();}
    Applicazione() {
      Frame f= new Frame ();
      f.setSize(200,200);    // Dimensione della finestra
      f.setLocation(50,100); // Posizione della finestra
      f.add(new Label("Clicca sul bottone di chiusura"));
      Controllore c=new Controllore();
      f.addWindowListener(c);
       f.show();
    }
}
```

Una finestra che risponde – 2a

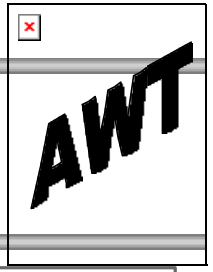



```
import java.awt.event.*;
class Controllore implements WindowListener {
                    public void windowClosing(WindowEvent we) {
                       System.exit(1); // termina il programma
    }
  public void windowIconified(WindowEvent we) {}
  public void windowDeiconified(WindowEvent we) {}
  public void windowClosed(WindowEvent we) {}
  public void windowOpened(WindowEvent we) {}
  // public void windowActivating(WindowEvent we) {}
  public void windowActivated(WindowEvent we) {}
  public void windowDeactivated(WindowEvent we) {}
}
```

La maggior parte dei metodi implementati sono vuoti, ma l'implementazione é obbligatoria per rispettare il contratto dell'Interface

Marco Ronchetti - ronchet@altavista.it

Una finestra che risponde – 2b



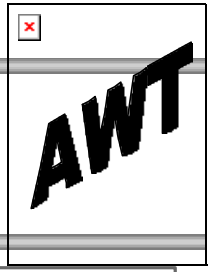

```
import java.awt.event.*;
class Controllore extends WindowAdapter {
     public void windowClosing(WindowEvent we) {
         System.exit(1); // termina il programma
    }
}
```

Per ogni Interfaccia XListener
esiste un classe Xadapter,
che definisce tutti I metodi richiesti
dall'interfaccia come vuoti: basta
quindi sovrascrivere quelli che servono.

!Attenzione all'ereditarietá multipla!

Marco Ronchetti - ronchet@altavista.it

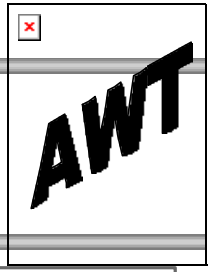
Dealing with events - 1.1a




```
import java.awt.*;
import java.awt.event.*;
class Controller
    implements MouseListener, MouseMotionListener {
    private int last_x, last_y;
    private Frame a;
    Controller(Frame sender) {a=sender;}
// A method from the MouseListener interface.
// Invoked when the user presses a mouse button.
    public void mousePressed(MouseEvent e) {
        last_x = e.getX();
        last_y = e.getY();  }
// A method from the MouseMotionListener interface.
//Invoked when the user drags the mouse with a button pressed.
    public void mouseDragged(MouseEvent e) {
        Graphics g = a.getGraphics();
        int x = e.getX(), y = e.getY();
        g.drawLine(last_x, last_y, x, y);
        last_x = x; last_y = y;  }
// The other, unused methods of the MouseListener interface.
    public void mouseReleased(MouseEvent e) {;}
    public void mouseClicked(MouseEvent e) {;}
    public void mouseEntered(MouseEvent e) {;}
    public void mouseExited(MouseEvent e) {;}
// The other method of the MouseMotionListener interface.
    public void mouseMoved(MouseEvent e) {;}
}
```

Marco Ronchetti - ronchet@altavista.it

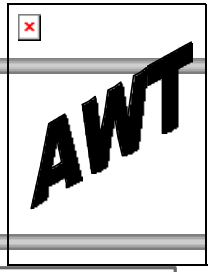
Dealing with events - 1.1a




```
public class Applicazione extends Frame{
  public static void main(String s[]) {
    new Applicazione ();
  }
  Applicazione() {
      Controller c=new Controller(this);
      setSize(200,200);
      show();
      addMouseListener(c);
      addMouseMotionListener(c);
  }
}
```

Marco Ronchetti - ronchet@altavista.it

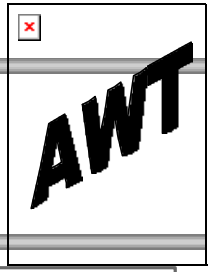
Dealing w.events-1.1b




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

class Applicazione extends Frame implements
    MouseListener, MouseMotionListener {
      int last_x, last_y;
  public static void main(String s[]) { new Applicazione();}
  Applicazione() {
      setSize(200,200);
      show();
      addMouseListener(this);
      addMouseMotionListener(this);
  }
// A method from the MouseListener interface.
// Invoked when the user presses a mouse button.
    public void mousePressed(MouseEvent e) {
        last_x = e.getX();
        last_y = e.getY();  }
```

Marco Ronchetti - ronchet@altavista.it

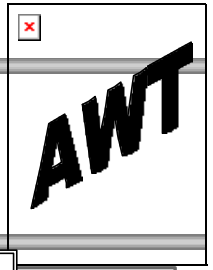
Dealing w.events-1.1b




```
// A method from the MouseMotionListener interface.
//Invoked when the user drags the mouse with a button pressed.
    public void mouseDragged(MouseEvent e) {
        Graphics g = getGraphics();
        int x = e.getX(), y = e.getY();
        g.drawLine(last_x, last_y, x, y);
        last_x = x; last_y = y;  }
// The other, unused methods of the MouseListener interface.
    public void mouseReleased(MouseEvent e) {;}
    public void mouseClicked(MouseEvent e) {;}
    public void mouseEntered(MouseEvent e) {;}
    public void mouseExited(MouseEvent e) {;}
// The other method of the MouseMotionListener interface.
    public void mouseMoved(MouseEvent e) {;}
}
```

Marco Ronchetti - ronchet@altavista.it

Dealing with events-1.1d

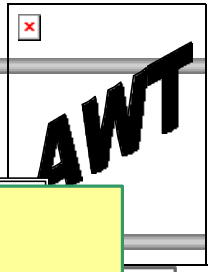



```
import java.awt.*;
import java.awt.event.*;
class Applicazione extends Frame {
  int last_x, last_y;
  public static void main(String a[]) {
       new Applicazione();}
  Applicazione() {
    super("Con le inner classes");
    setSize(200,200);
    // Create a clear button
    Button b = new Button("Clear");
    // Define, instantiate, and register a listener
    //to handle button presses
    b.addActionListener(new ActionListener() {
       public void actionPerformed(ActionEvent e) {
          // clear the scribble
           Graphics g = getGraphics();
           g.setColor(getBackground());
           g.fillRect(0, 0, getSize().width, getSize().height);
       }
    });
```

Using inner anonymous Classes

Marco Ronchetti - ronchet@altavista.it

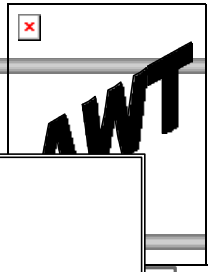
Dealing with events-1.1d




```
// And add the button to the applet
  this.add(b,BorderLayout.NORTH);
  show();
// Define, instantiate and register a
// MouseListener object.
  this.addMouseListener(new MouseAdapter() {
       public void mousePressed(MouseEvent e) {
           last_x = e.getX();
           last_y = e.getY();
       }
    });
// Define, instantiate and register a MouseMotionListener object.
  this.addMouseMotionListener(new MouseMotionAdapter() {
       public void mouseDragged(MouseEvent e) {
           Graphics g = getGraphics();
           int x = e.getX(), y = e.getY();
           g.setColor(Color.black);
           g.drawLine(last_x, last_y, x, y);
           last_x = x; last_y = y;
       }
    });
  }
}
```

Using inner anonymous Classes

1.1a with image persistence




```
import java.awt.*;
import java.awt.event.*;
class Controller
    implements MouseListener, MouseMotionListener {
    private int last_x, last_y;
    private Application a;
     Controller(Application sender) {a=sender;}
// MouseListener interface. -----------------------------------
    public void mousePressed(MouseEvent e) {
        last_x = e.getX();   last_y = e.getY();  }
    public void mouseReleased(MouseEvent e) {;}
    public void mouseClicked(MouseEvent e) {;}
    public void mouseEntered(MouseEvent e) {;}
    public void mouseExited(MouseEvent e) {;}
// MouseMotionListener interface. ------------------------------
    public void mouseDragged(MouseEvent e) {
        Graphics g = a.getOffScreenGraphics();
        int x = e.getX(); y = e.getY();
        g.drawLine(last_x, last_y, x, y);
        last_x = x; last_y = y;
        a.repaint();
    }
    public void mouseMoved(MouseEvent e) {;}
}
```

Marco Ronchetti - ronchet@altavista.it

1.1a with image persistence

```
public class Applicazione extends Frame{
  Image offscreenImg;
  Graphics offscreenG;
  public static void main(String s[]) { new Applicazione ();}
  Applicazione() {
      Controller c=new Controller(this);
       setSize(200,200);
      show();
       offscreenImg = createImage(this.size().width,this.size().height);
      offscreenG = offscreenImg.getGraphics();
      addMouseListener(c);
      addMouseMotionListener(c);
  }
  public void paint(Graphics g) {
    if (offscreenImg!=null) {
        g.drawImage(offscreenImg,0,0,this);
    }
  }
  public Graphic getOffScreenGraphics(){
    return offscreenG ;
  }
}
```

Problema:
Che succede se
Si fa un resize della finestra?
Come si può aggiustare?

Marco Ronchetti - ronchet@altavista.it

Double buffering

Le animazioni vengono effettuate cancellando l'immagine presente e ridisegnando la nuova. Questo da' origine ad un fastidioso sfarfallio.
Per evitare questo effetto, la nuova immagine viene disegnata in memoria anziche' a video: quando l'immagine e' pronta, quella presente a video viene sostituita con la nuova (tecnica di double buffering).

La tecnica si basa sulla ridefinizione del metodo "update()" che viene chiamato dalla repaint(), e che a sua volta schedula la paint(). La versione standard di update e' :

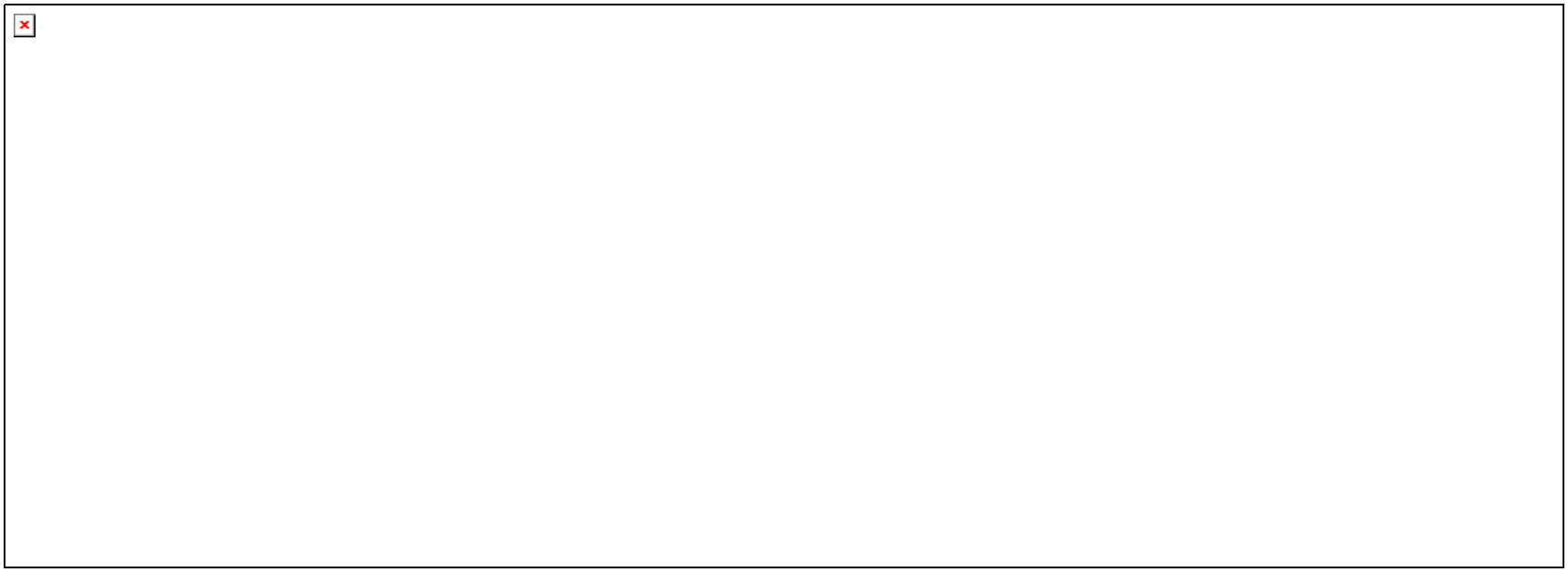
```
public void update(Graphics g) {
  g.setColor(getBackground());
  g.fillRect(0,0,width,height);
  g.setColor(getForeground());
  paint(g);
}
```

Eliminare le righe rosse per evitare il flickering

Marco Ronchetti - ronchet@altavista.it

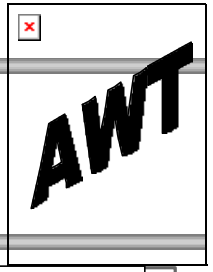
Grafica: Coordinate

Origine in alto a sinistra



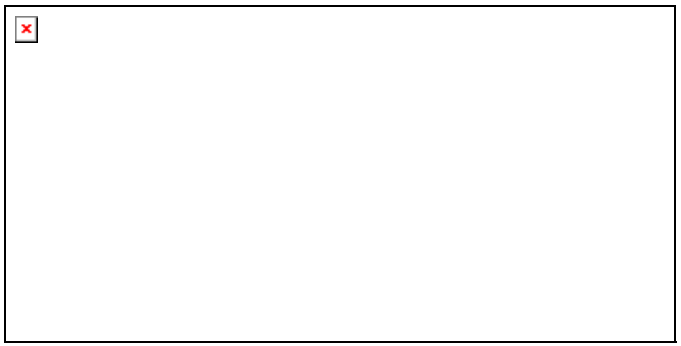
Marco Ronchetti - ronchet@altavista.it

Elementi Grafici



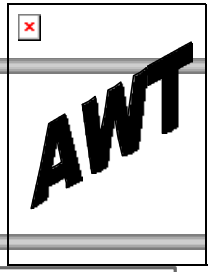

```
import java.awt.*;
import java.awt.event.*;
public class TestPattern extends
                  java.applet.Applet {
    int theta = 45;
    public void paint( Graphics g ) {
        int Width = size().width;
        int Height = size().height;
        int width = Width/2;
        int height = Height/2;
        int x = (Width - width)/2;
        int y = (Height- height)/2;
        int [] polyx =  { 0, Width/2, Width, Width/2 };
        int [] polyy =  { Height/2, 0, Height/2, Height };
        Polygon poly = new Polygon( polyx, polyy, 4 );

        g.setColor( Color.black );
        g.fillRect( 0, 0, size().width, size().height );
        g.setColor( Color.yellow );
        g.fillPolygon( poly );
        g.setColor( Color.red );
        g.fillRect( x, y, width, height );
        g.setColor( Color.green );
        g.fillOval( x, y, width, height );
        g.setColor( Color.blue );
        int delta = 90;
        g.fillArc( x, y, width, height, theta, delta );
        g.setColor( Color.white );
        g.drawLine( x, y, x+width, x+height );
    }
```

```
public void init() {
        addMouseListener( new MouseAdapter() {
            public void mousePressed( MouseEvent e ) {
                theta = (theta + 10) % 360;
                repaint();
            }
        } );
    }
}
```



Marco Ronchetti - ronchet@altavista.it

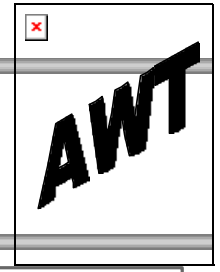
Metodi grafici di Graphics



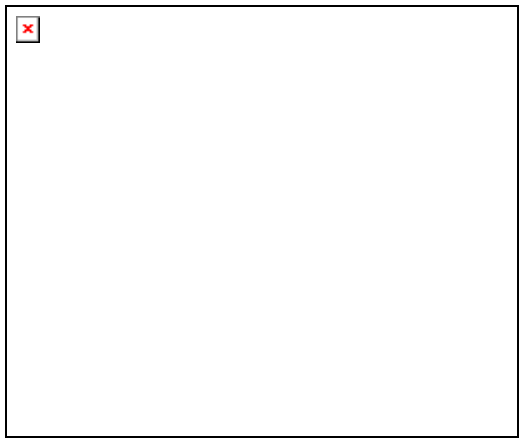
draw3DRect()	Draws a highlighted, 3D rectangle
drawArc()	Draws an arc
drawLine()	Draws a line
drawOval()	Draws an oval
drawPolygon()	Draws a polygon, connecting endpoints
drawPolyline()	Draws a line connecting a polygon's points
drawRect()	Draws a rectangle
drawRoundRect()	Draws a rounded-corner rectangle
fill3DRect()	Draws a filled, highlighted, 3D rectangle
fillArc()	Draws a filled arc
fillOval()	Draws a filled oval
fillPolygon()	Draws a filled polygon
fillRect()	Draws a filled rectangle
fillRoundRect()	Draws a filled, rounded-corner rectangle

Marco Ronchetti - ronchet@altavista.it

Text & Fonts

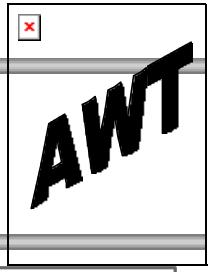



```
import java.awt.Font;  import java.awt.Graphics;  import java.awt.FontMetrics;

public class ManyFonts extends java.applet.Applet {
  public void paint(Graphics g) {
    Font f = new Font("TimesRoman", Font.PLAIN, 18);
    FontMetrics fm = getFontMetrics(f);
    g.setFont(f);
    String s = "This is a plain font";
    int xstart = (size().width - fm.stringWidth(s)) / 2;
    //int ystart = (size().height + fm.getHeight()) / 2;
    g.drawString(s, xstart, 25);

    Font fb = new Font("TimesRoman", Font.BOLD, 18);
    Font fi = new Font("TimesRoman", Font.ITALIC, 18);
    Font fbi = new Font("TimesRoman", Font.BOLD + Font.ITALIC, 18);
    g.setFont(fb);
    g.drawString("This is a bold font", 10, 50);
    g.setFont(fi);
    g.drawString("This is an italic font", 10, 75);
    g.setFont(fbi);
    g.drawString("This is a bold italic font", 10, 100);
  }
}
```

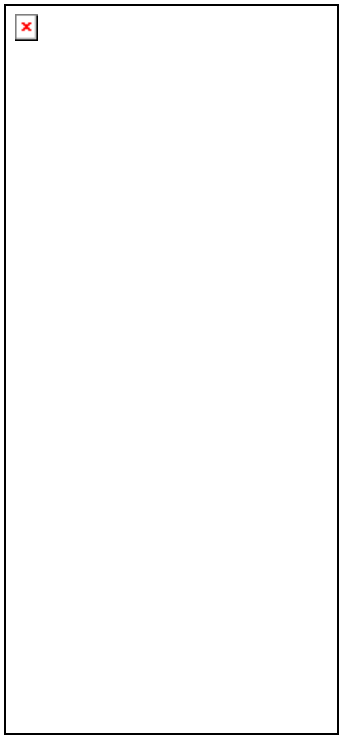


Marco Ronchetti - ronchet@altavista.it

Colore

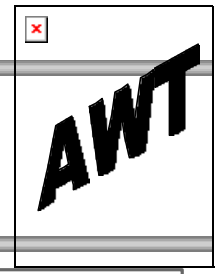



```
import java.awt.Graphics;
import java.awt.Color;

public class ColorBoxes extends java.applet.Applet {

  public void paint(Graphics g) {
    int rval, gval, bval;

    for (int j = 30; j < (this.size().height -25); j += 30)
      for (int i = 5; i < (this.size().width -25); i+= 30) {
  rval = (int)Math.floor(Math.random() * 256);
  gval = (int)Math.floor(Math.random() * 256);
  bval = (int)Math.floor(Math.random() * 256);

  g.setColor(new Color(rval,gval,bval));
  g.fillRect(i,j,25,25);
  g.setColor(Color.black);
  g.drawRect(i-1,j-1,25,25);
      }
  }
}
```

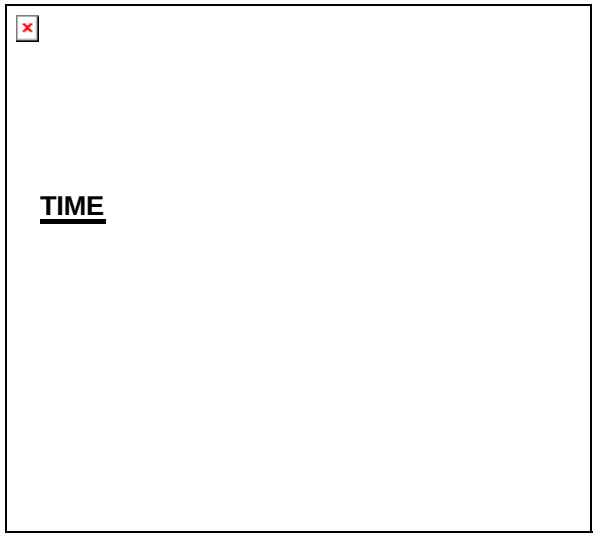


Marco Ronchetti - ronchet@altavista.it

Esercizio



Scrivere una calcolatrice che risponda a menu, mouse e keyboard events.



Marco Ronchetti - ronchet@altavista.it