

Programmazione ad Oggetti

Interfacce grafiche

V 1.2 © Marco Torchiano 2005

JFrame

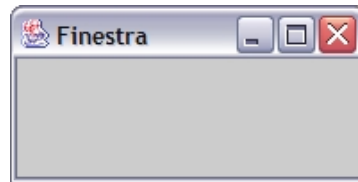
- JFrame è la classe di base per le finestre
- Fornisce tutte le caratteristiche di una finestra vuota
 - ◆ Barra del titolo
 - ◆ Pulsanti standard
 - ◆ Bordo ridimensionabile
 - ◆ Etc.

2

JFrame – Esempio

```
public class FrameSemplice extends JFrame {
    public static void main(String[] args) {
        JFrame f = new FrameSemplice();
        f.show();
    }
    public FrameSemplice(){
        super("Finestra");
        setSize(200,100);
    }
}
```

Che succede se non faccio setSize() ?



3

Linee guida

- Nella realizzazione di interfacce grafiche occorre distinguere due aspetti:
 - ♦ Layout: come disporre gli elementi grafici per ottenere un certo aspetto grafico
 - ♦ Eventi: come legare il comportamento agli elementi grafici
- La logica del programma deve restare, per quanto possibile, separata dall'interfaccia grafica.

4

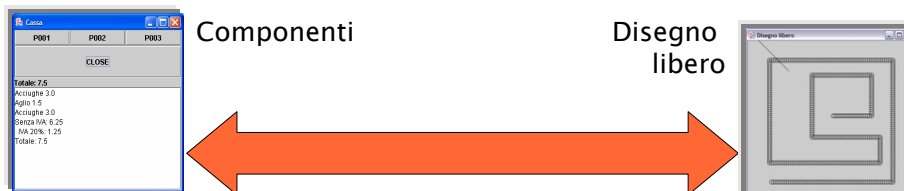
Flusso di esecuzione

- Il metodo main nelle GUI serve solo a creare un'istanza della finestra
- Nelle GUI **NON** esiste un ordine di esecuzione predefinito
 - ♦ Le operazioni vengono eseguite in risposta ad eventi esterni (es. mouse)
 - ♦ La gestione degli eventi è serializzata
 - ♦ Per eseguire operazioni in parallelo occorre utilizzare dei thread

5

Tipologia di GUI

- Possiamo distinguere due tipologie estreme di interfacce grafiche:
 - ♦ composizione di componenti
 - ♦ “disegno libero”
- Nella realtà si usano interfacce intermedie tra questi tipi.



6

GUI con Componenti

- Interfacce come composizione di componenti
 - ♦ Utilizzano i componenti grafici predefiniti (es. pulsanti, caselle di testo, etichette)
 - ♦ Sono adatte per creare un'interfaccia grafica di un programma che “in teoria” potrebbe avere un'interfaccia a carattere
 - ♦ Permettono di trattare informazioni per lo più testuali

7

GUI con disegno

- Interfacce a “disegno libero”
 - ♦ Scrivono direttamente sullo schermo utilizzando l'interfaccia Graphics
 - ♦ Permettono di trattare informazioni grafiche (diagrammi etc.)
 - ♦ Di solito usano un (J)Panel per definire esattamente l'area in cui disegnare

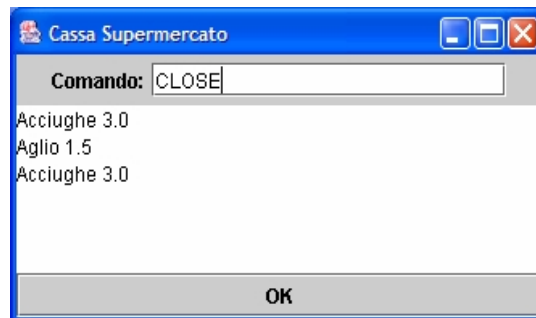
8

Esercizio

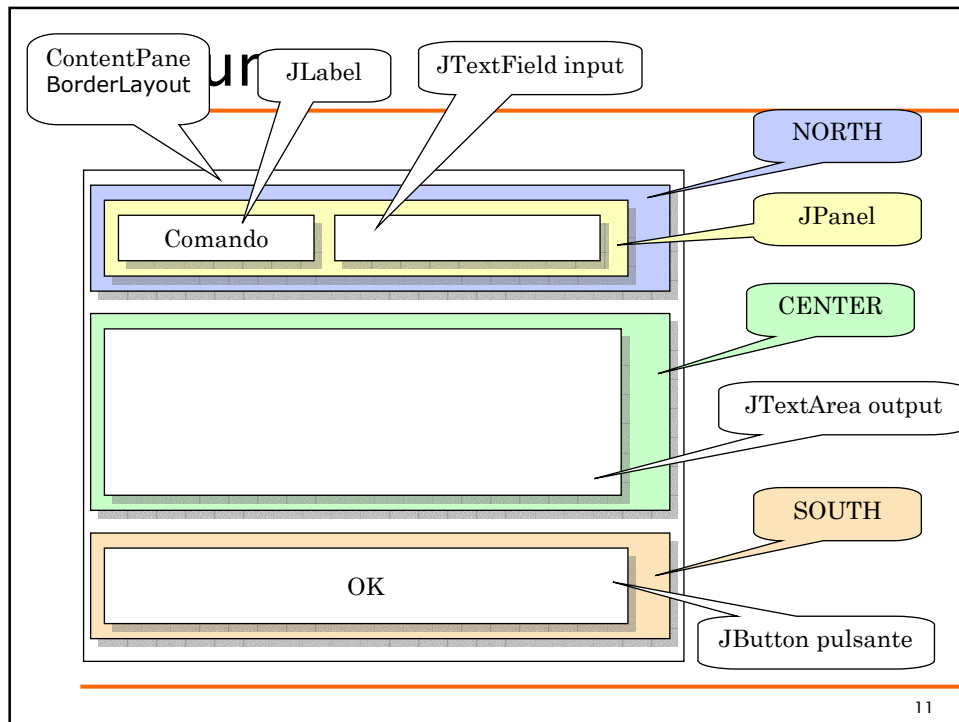
- Definire un'interfaccia grafica per l'applicazione cassa del supermercato.
- Due possibilità:
 - ♦ Il più simile possibile alla versione testuale
 - ♦ Versione più user friendly che sfrutta le possibilità delle GUI

9

Simile alla versione testuale



10



11

Layout

```

public class SimpleCassa extends JFrame
    implements ActionListener{
    JTextField input=new JTextField(20);
    JTextArea output=new JTextArea();
    JButton pulsante=new JButton("OK");
    public SimpleCassa(){
        Container c = this.getContentPane();
        c.setLayout(new BorderLayout());
        JPanel panel = new JPanel();
        panel.add(new JLabel("Comando:"));
        panel.add(input);
        c.add(panel, BorderLayout.NORTH);
        c.add(output, BorderLayout.CENTER);
        c.add(pulsante, BorderLayout.SOUTH);
        setSize(500,400);
    }
}
    
```

Componenti

Creazione della GUI

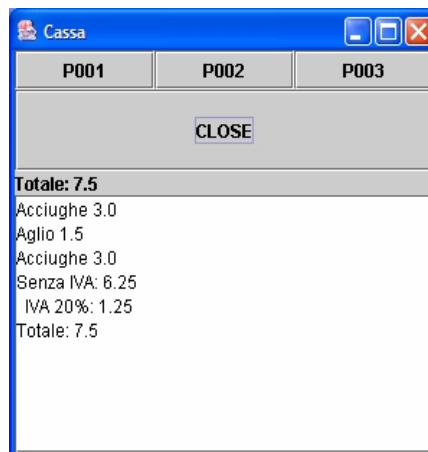
12

Eventi

```
pulsante.addActionListener(this);
}
public void actionPerformed(ActionEvent e){
    String cmd = input.getText();
    if(cmd.equals("CLOSE")){
        output.append("---- Scontrino ----\n");
        //...
    }
}
}
```

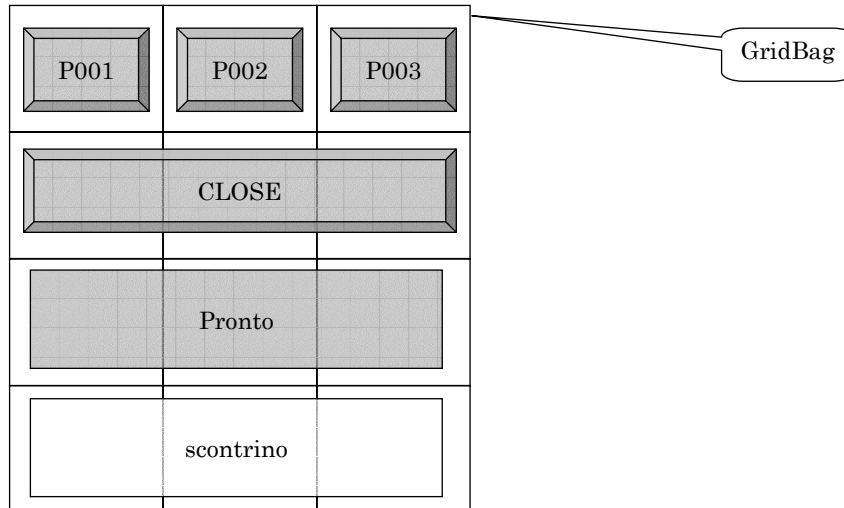
13

Versione User-friendly



14

Struttura



15

Test di GUI

- Per eseguire il test di GUI costruite come composizione di componenti si possono adottare due approcci:
 - ♦ Test dall'esterno
 - ♦ Test dall'interno

16

Test esterno

- Test dall'esterno
 - ♦ Tramite il sistema operativo si inviano degli eventi all'applicazione simulando le operazioni eseguite da un utente
 - ♦ Pro: approccio realistico
 - ♦ Contro: macchinoso e dipendente dal S.O.
 - ♦ Di solito si usano degli strumenti che registrano le operazioni fatte da un utente e le replicano

17

Test interno

- Test dall'interno
 - ♦ Si chiamano direttamente i metodi dei componenti grafici che hanno (circa) gli stessi effetti di un uso reale (ad es. doClick)
 - ♦ Pro: semplice e indipendente dal S.O.
 - ♦ Contro: poco realistico perchè "salta" alcuni passi
 - ♦ Contro: bisogna rendere "testabili" le classi
 - Es. rendere visibili gli attributi

18

Test di interfacce grafiche

```
public void testGUI(){
    SimpleCassa gui = new SimpleCassa();
    gui.setVisible(true);
    gui.input.setText("P001");
    gui.pulsante.doClick();
    gui.input.setText("P002");
    gui.pulsante.doClick();
    gui.input.setText("P001");
    gui.pulsante.doClick();
    gui.input.setText("CLOSE");
    gui.pulsante.doClick();
    String output = gui.output.getText();
    assertTrue("output sbagliato",
        output.indexOf("Totale: 7.5")>-1);
}
```

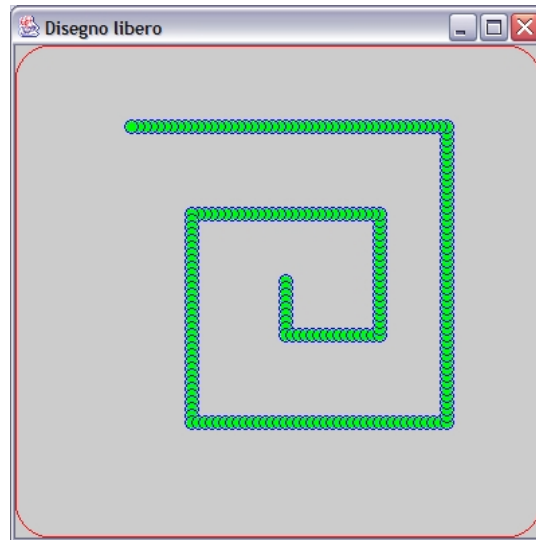
19

Disegno libero

- Per disegnare in maniera esplicita l'interfaccia grafica si usano due elementi:
 - ◆ Il metodo `void paint(Graphics g)`
 - Deve essere ridefinito
 - Viene chiamato dal S.O.
 - ◆ L'interfaccia `Graphics`
 - Fornisce i metodi per disegnare

20

Disegno Libero – Esempio



21

Esempio

```
public class Disegno extends JFrame{
    int x;
    int y;
    public void paint(Graphics g){
        Rectangle b = getBounds();
        g.setColor(Color.RED);
        g.drawRoundRect(4,30,
            b.width-9,b.height-35,50,50);
        g.setColor(Color.BLUE);
        g.drawOval(x,y,10,10);
        g.setColor(Color.GREEN);
        g.fillOval(x+1,y+1,9,9);
    }
}
```

g copre tutta l'area della finestra, titolo e bordi inclusi


22

Events

```
public class Disegno implements KeyListener{
    public void keyPressed(KeyEvent e) {
        if(e.getKeyCode()==KeyEvent.VK_DOWN){
            moveXY(0,5);
        }
        // ...
    }
    void moveXY(int deltaX, int deltaY){
        x+=deltaX;
        y+=deltaY;
        this.repaint();
    }
}
```

23

Osservazioni

- L'operazione di repaint NON cancella il contenuto della finestra
 - ♦ Quindi si ha l'effetto scia 
- Occorre cancellare esplicitamente il contenuto della finestra:


```
Rectangle bounds = getBounds();
g.clearRect(0,0,bds.width,bounds.height);
```

24

Consigli

- Consigli:
 - ♦ Definire il metodo `paint` su un componente che non contiene nulla
 - ♦ **NON** ridefinire il metodo `paint()` su una finestra che contiene componenti
 - ♦ Spesso si definisce una classe che estende `JPanel` e si ridefinisce il metodo `paint()`