Fondamenti di Java

Il linguaggio cult per la programmazione a oggetti

Appendice: la creazione di interfacce grafiche

Interfacce grafiche in Java

- Java è un linguaggio nato per scopi matematici (in particolare per il calcolo matriciale), ma si è poi diffuso principalmente in altri campi
- In particolare, sicurezza dei dati e portabilità ne hanno fatto un linguaggio molto usato in rete
- Java è largamente impiegato oggi nello sviluppo di applicazioni grafiche

Le GUI in Java

- Java è ampiamente utilizzato per la creazione di interfacce utente o GUI
- Per GUI (Graphic User Interface) si intendono quelle finestre di dialogo che permettono all'utente lo scambio di dati con il programma
- Elementi tipici delle GUI sono dunque finestre, etichette, campi di testo, aree di testo, checkbox, radio button, bottoni, menù a tendina e così via.
- I package java e javax contengono pacchetti di classi dedicate alla creazione di questi elementi grafici.

Contenitori e componenti

- I componenti grafici (caselle di testo, pulsanti, label etc.), per essere visualizzati, devono essere collocati all'interno di contenitori, cioè aree rettangolari delimitate sullo schermo
- Il principale contenitore grafico è la finestra, un oggetto di classe Frame
- La classe predefinita Frame implementa una serie di metodi che consentono di personalizzare il contenitore
- Una finestra può essere a sua volta divisa in zone rettangolari interne (oggetti Panel) e i componenti possono essere disposti secondo vari modelli o layout (oggetti di classe FlowLayout, GridLayout, BorderLayout)

Il package grafico awt di Java

- La prima generazione di Java (le cui classi costituiscono il package java) conteneva come pacchetto per le applicazioni grafiche il package awt (Abstract Window Toolkit)
- Il package awt contiene classi specifiche (Label, TextField, Button...) per la rappresentazione dei vari elementi che compaiono nelle finestre di dialogo
- Tutte queste classi sono derivate da una superclasse Component che descrive il generico elemento grafico

Il package swing

- Nella seconda release di Java (le cui classi sono contenute nel package javax) un nuovo package grafico è stato affiancato al preesistente package awt.
- Il nuovo package, denominato swing, è introdotto principalmente per due scopi:
- alleggerire la sintassi per la programmazione degli eventi associati ai componenti grafici
- migliorare il layout (cioè la disposizione degli elementi) all'interno delle finestre

La programmazione a eventi

- Gli elementi grafici creati con questi due package sono infatti elementi inizialmente non interattivi
- Per ottenere che un elemento grafico non sia un semplice disegno, ma sia in grado di compiere azioni (inserimento dati in un textfield, esecuzione di un calcolo alla pressione di un bottone, cambio aspetto al passaggio del mouse, ecc.) è necessario associare al componente grafico un ascoltatore che lo renda attivo

Gli ascoltatori

- Un ascoltatore è un metodo che può essere implementato nella classe in cui si disegnano gli elementi grafici e nel quale vengono descritte le azioni che il programma deve compiere quando un certo elemento grafico viene attivato
- Ad esempio è necessario aggiungere un ascoltatore a un textfield per poter acquisire il valore in esso digitato, o per ottenere l'esecuzione di un comando alla pressione di un bottone.

Le classi per gli ascoltatori

- Gli ascoltatori sono implementati nelle singole classi, ma i loro prototipi sono dichiarati in classi specifiche, denominate interfacce
- Le interfacce sono particolari classi nelle quali non si trovano attributi e in cui i metodi sono dichiarati, ma non implementati
- Una classe può dunque implementare più interfacce e ridefinire i metodi in esse contenute adattandoli ai sui obiettivi

WindowListener ed ActionListener

- Le principali interfacce per gli ascoltatori sono:
- la classe WindowListener, che serve a rendere attive le finestre, permettendo tra l'altro la chiusura delle stesse e dei processi associati
- La classe WindowAdapter, che consente di creare un semplice acoltatore che può essere associato a ogni finestra, ne gestisce la chiusura e termina i processi associati
- La classe ActionListener, che permette di rendere attivi gli elementi contenuti all'interno della finestra stessa

Gli eventi legati alle GUI

- Quando una finestra o un suo componente grafico riceve un comando, attraverso mouse o tastiera, dall'utente, si dice che viene sollevato un evento
- Lo scopo dei metodi ereditati dalle interfacce è quello di gestire questi eventi, cioè di eseguire i comportamenti previsti dall'attivazione dell'elemento grafico

Gli eventi legati alle finestre

- Gli eventi legati alle finestre (apertura, chiusura, riduzione a icona, chiusura dei processi associati) sono gestiti dall'interfaccia WindowListener e sono di tipo WindowEvent
- Gli eventi associati ai componenti sono gestiti dall'interfaccia ActionListener e sono di tipo ActionEvent
- Il metodo actionPerformed(ActionEvent e) descrive le azioni da intraprendere per gestire l'evento.

Esempi di semplici GUI

In questo esempio si crea una semplice finestra con messaggio

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

public class Finestra {

  Finestra(){
  Handler h=new Handler();
  Frame f=new Frame("Titolo della finestra");
  Label l=new Label("Testo del messaggio");
  f.setSize(400,200);
  f.setLocation(100,100);
  f.setBackground(Color.magenta);
  f.add(l);
  f.setLayout(new FlowLayout());
  f.addWindowListener(h);
  f.setVisible(true);
  }}
```

La classe Handler

```
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
public class Handler extends WindowAdapter {
    public void windowClosing(WindowEvent e)
    {
        e.getWindow().dispose();
        System.exit(0);
    }
}
```

Note al primo esempio

- La classe Handler compare in tutti gli esempi del genere e rappresenta l'ascoltatore associato alla finestra
- La classe Handler è una sottoclasse derivata da WindowAdapter e provoca necessaria la chiusura dell'elemento grafico e dei processi associati
- Il metodo addWindowListener() della classe Frame aggiunge l'ascoltatore alla finestra
- I metodi per la gestione degli eventi e le relative interfacce si trovano nel package event

Esempi di semplici GUI

- In questo esempio si crea una finestra con un bottone e si calcola quante vote si preme il bottone

```
import java.awt.*;
import java.awt.event.*;
public class Contatore implements ActionListener{
//attributi della classe
Frame f;
Label l;
Button b;
int x=0;
Handler h;
```

Esempi di semplici GUI

```
Contatore(){
  int x=0;
  h=new Handler();
  f=new Frame("Quante volte si preme il pulsante?");
  l=new Label("      Il pulsante è stato premuto "+x+" volte     ");
  l.setBackground(Color.cyan);
  b=new Button("   Premere il pulsante    ");
  b.setBackground(Color.LIGHT_GRAY);
  b.addActionListener(this);
  f.setSize(300,150);
  f.setLocation(100,100);
  f.setBackground(Color.magenta);
  f.add(b);
  f.add(l);
  f.setLayout(new FlowLayout());
  f.addWindowListener(h);
  f.setVisible(true);
}
```

Esempi di semplici GUI

```
/* la finestra si crea nel costruttore della
   classe. Quando ci sono elementi attivi il
   costruttore è seguito dal gestore
   dell'evento*/
public void actionPerformed(ActionEvent e) {
  x++;
  l.setText("    Il pulsante è stato premuto
    "+x+" volte    ");
}}
```

Esempi di semplici GUI

- Per mandare in esecuzione le finestre è necessario invocare nel main() della classe principale il costruttore della classe

```
public class ProvaFinestra {
public static void main(String[] args) {
   new Finestra();
   new Contatore();}
}
```

Esempi di semplici GUI

- Questo esempio permette il login di un profilo utente

```
import java.awt.*;
import java.awt.event.*;
public class Login implements ActionListener{
  String id;
  String psw;
  Frame f;
  Label l_id,l_psw;
  Button b;
  TextField t_id,t_psw;
```

Esempi di semplici GUI

```
Login(){
        Frame f=new Frame("Login");
        f.setSize(300,200 );
        f.setLocation(200, 150);
        f.setBackground(Color.lightGray);
        Handler h=new Handler();
        Label l_id=new Label("Nome utente:  ");
        l_id.setBackground(Color.lightGray);
        Label l_psw=new Label("Password:      ");
        l_psw.setBackground(Color.lightGray);
        t_id=new TextField(20);
        t_psw=new TextField(20);
        Button b=new Button("      Login      ");
        f.setLayout(new FlowLayout());
        f.add(l_id);
        f.add(t_id);
        f.add(l_psw);
        f.add(t_psw);
        b.addActionListener(this);
        f.add(b);
        f.addWindowListener(h);
        f.setVisible(true);
}
```

Esempi di semplici GUI

```
public void actionPerformed(ActionEvent e) {

 if(t_id.getText().equals("Alex") && t_psw.getText().equals("fma"))
    new Profilo("Alex Louis Armstrong");
 else{
    Frame n=new Frame("Errore");
    n.setSize(400,100 );
    n.setLocation(500, 500);
    n.setBackground(Color.magenta);
    Handler h=new Handler();
    Label l=new Label("Nome utente o  password non corretti");
    l.setBackground(Color.lightGray);
    n.add(l);
    n.addWindowListener(h);
    n.setVisible(true);}}}
```

Esempi di semplici GUI

```
import java.awt.*;
import javax.swing.ImageIcon;
import javax.swing.JLabel;
public class Profilo {
Profilo(String id){
        Frame p=new Frame("Profilo utente");
        p.setSize(600,600 );
        p.setLocation(100, 100);
        p.setBackground(Color.lightGray);
        Handler h=new Handler();
        Label l=new Label("Benvenuto "+id);
        l.setBackground(Color.lightGray);
        String s="armstrong.jpg";
        ImageIcon avatar=new ImageIcon(s);
        JLabel label=new JLabel(avatar, JLabel.CENTER);
        p.setLayout(new FlowLayout());
        p.add(label);
        p.add(l);
        p.addWindowListener(h);
        p.setVisible(true);
        System.out.println("Prova");}}
```