23 - Graphical User Interface – GUI (Cenni)

Programmazione e analisi di dati
Modulo A: Programmazione in Java

Paolo Milazzo

Dipartimento di Informatica, Università di Pisa
http://pages.di.unipi.it/milazzo
milazzo@di.unipi.it

Corso di Laurea Magistrale in Informatica Umanistica
A.A. 2017/2018

Premessa

In questa lezione vedremo gli aspetti relativi alla realizzazione di interfacce grafiche in maniera molto superficiale.

Nella pagina web del corso potete trovare una DISPENSA introduttiva in italiano sullo sviluppo di interfacce grafiche

- Autori della dispensa: M. de Leoni, M. Mecella e S. Saltarelli – Università di Roma

Inoltre, trovate tutte le informazioni sulle varie classi usate in questa lezione nella documentazione della Libreria Standard di Java (Java API)

- link sulla pagina web del corso

Interfaccia utente testuale

Nei programmi realizzati fino ad ora abbiamo sempre usato una interfaccia testuale per interagire con l'utente

Nei programmi più complessi l'interfaccia era tipicamente basata su un menù principale:

```
Buongiorno, Paolo Milazzo

SPORTELLO N. 1
OPERAZIONI DISPONIBILI
    [A]pri nuovo conto
    [S]aldo
    [V]ersamento
    [P]relievo
    [M]atura interessi (tutti i conti)
    [R]iepilogo conti
    [C]ambia tassi di interesse
    [U]scita
scelta: A

Inserisci tipo cliente [F/B]: F
Inserisci saldo iniziale: 100
Creato conto numero 1001
```

GUI (1)

La grande maggioranza della applicazioni moderne non prevede una interfaccia testuale

Le applicazioni normalmente prevedono una interfaccia utente grafica (o Graphical User Interface – GUI)

L'interfaccia grafica può essere realizzata:

- tramite opportune librerie del linguaggio di programmazione (applicazioni tradizionali)
- come pagina web (interfaccia web-based) da visualizzare tramite un browser

GUI (2)

Lo sviluppo di un'interfaccia grafica è un'attività piuttosto complessa

- A volte implementare un interfaccia grafica richiede più tempo che realizzare il "motore" dell'applicazione

Esistono molte librerie (GUI frameworks) per creare interfacce grafiche in Java:

- Alcuni nomi: AWT/Swing, SWT, JavaFX, ...

Scopo di questa lezione

Gli scopi di questa lezione sulle interfacce grafiche:

- mostrare esempi che potrebbero essere utili per la realizzazione del progetto d'esame
- illustrare i concetti essenziali che possano consentire un approfondimento individuale dell'argomento

Vedremo alcuni esempi pratici basati su AWT/Swing (il GUI framework "classico" di Java).

Esempi

Gli esempi che vedremo:

1. Finestre pop-up per interagire con l'utente
   - interfaccia grafica un po' brutale...
2. Finestra menù principale + finestre pop-up
   - un po' rozza, ma potrebbe andare più o meno bene per un progetto d'esame
3. Piccolo esempio di interfaccia a singola finestra
   - iniziano le complicazioni...
4. Piccolo esempio di interfaccia a finestre multiple
   - sempre più complicato...

Inoltre, vedremo un piccolo esempio di uso di una area di testo con barre di scorrimento (utile quando si devono visualizzare molte righe di testo)

Packages

Tutte le classi che useremo per realizzare le interfacce grafiche apparterrano a uno diquesti packages:

- `java.awt`
- `javax.swing`

Per semplicità in tutti gli esempi importeremo tutte le classi di questi due pacchetti:

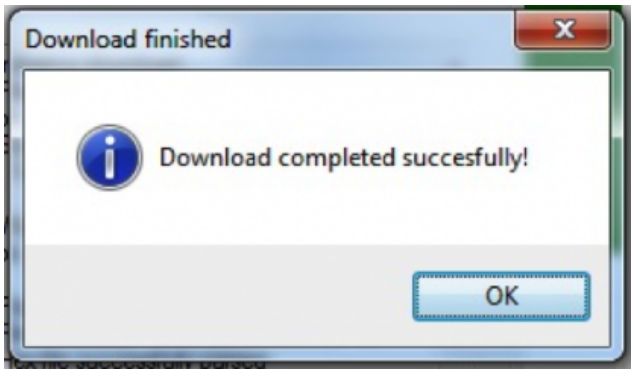
```
import java.awt.*;
import javax.swing.*;
```

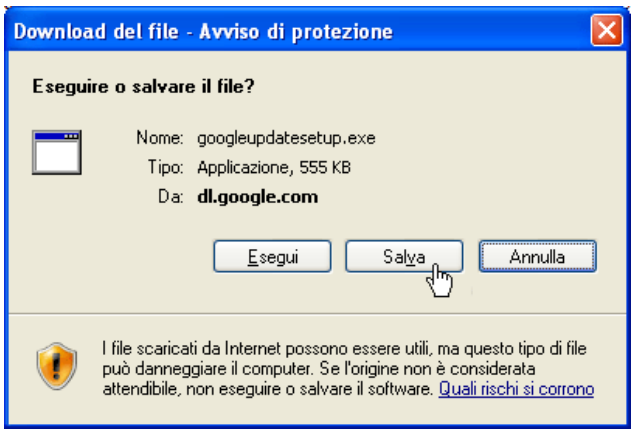
Finestre pop-up (1)

Le finestre pop-up sono piccole finestre per l'interazione con l'utente

Tipicamente possono contenere:

- brevi messaggi
- richieste di inserire un valore
- richieste di conferma





Finestre pop-up (2)

Le finestre pop-up tipicamente "appaiono" sopra alle finistre di una applicazione

- attirano l'attenzione dell'utente su un piccolo particolare (un messaggio, una scelta, ...)

Come primo approccio all'interfacciamento grafico possiamo usarle per sostituire gli input/output testuali di un nostro programma

Finestre pop-up (3)

In Java le finestre di pop-up possono essere create usando alcuni metodi (statici) della classe `JOptionPane`

- `JOptionPane.showMessageDialog(...)` - mostra un pop-up con un messaggio
- `JOptionPane.showInputDialog(...)` - mostra un pop-up con una casella di input
- `JOptionPane.showConfirmDialog(...)` - mostra un pop-up che offre una scelta

Esempio 1: calcolo della somma (1)

Come esempio di uso dei pop-up consideriamo un programma che calcola la somma di una sequenza di numeri inseriti dall'utente

Variante: dopo ogni inserimento si chiede all'utente se vuole continuare

Esempio 1: calcolo della somma (2)

Con l'interfaccia testuale:

```
import java.util.Scanner;

public class Somma {
  public static void main(String[] args) {
    Scanner input = new Scanner(System.in);

    int somma=0;
    char scelta;
    do {
      // chiede in input un numero
      System.out.println("Inserisci un numero:");
      somma+=input.nextInt(); //eccezione se l'utente sbaglia tipo

      do {
        // chiede se si vuole continuare
        // (si rende necessario un altro ciclo)
        System.out.println("Vuoi continuare (S/N)?");
        scelta = input.next().charAt(0);
        input.nextLine(); // consuma il resto della riga
      } while (scelta!='S' && scelta!='N');

    } while (scelta!='N');

    // stampa il risultato
    System.out.println("La somma e': " + somma);
  }
}
```

Esempio 1: calcolo della somma (3)

Con l'interfaccia grafica (basata su finestre pop-up):

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

public class SommaPopUp {
  public static void main(String[] args) {

    int somma=0;
    int scelta;
    do {
      // chiede in input un numero (il popup lo legge come stringa)
      String s = JOptionPane.showInputDialog("Inserisci un numero");
      somma+=Integer.parseInt(s);//eccezione se utente sbaglia tipo

      // chiede se si vuole continuare. Risposte possibili Si/No.
      // titolo finestra pop-up: Scelta.
      scelta=JOptionPane.showConfirmDialog(null,"Vuoi continuare?",
                                  "Scelta",JOptionPane.YES_NO_OPTION);

    } while (scelta!=JOptionPane.NO_OPTION); // notare

    // stampa il risultato
    JOptionPane.showMessageDialog(null, "La somma e': " + somma);
    }
}
```

Creare finestre pop-up (1)

Riguardo in particolare le finestre pop-up di scelta, è utile vedere i vari modi in cui possono essere create:

- crea un pop-up di scelta con le opzioni “Si”, “No” e “Annulla” e domanda `Cosa scegli"`

```
scelta = JOptionPane.showConfirmDialog(null,"Cosa scegli?");
```

- come prima, ma la finestra ha titolo `"Scelta"` e le opzioni sono solo “Si” e “No”

```
scelta = JOptionPane.showConfirmDialog(null,"Cosa scegli?",
                                       "Scelta",JOptionPane.YES_NO_OPTION);
```

La costante (di tipo `int`) `JOptionPane.YES_NO_OPTION` fa capire al metodo che le opzioni devono essere solo “Si” e “No”.

- Alternative: `JOptionPane.YES_NO_CANCEL_OPTION` e `JOptionPane.OK_CANCEL_OPTION`

Creare finestre pop-up (2)

Il valore restituito da `JOptionPane.showConfirmDialog(...)` è un valore intero che corrisponde al bottone premuto dall'utente

Per capire che bottone è stato premuto bisogna confrontare il valore restituito con delle costanti fornite dalla classe `JOptionPane`

```
scelta = JOptionPane.showConfirmDialog(null,"Premi un bottone...");
switch (scelta) {
  case JOptionPane.YES_OPTION : // premuto "Si"
  case JOptionPane.NO_OPTION : // premuto "No"
  case JOptionPane.CANCEL_OPTION : // premuto "Annulla"
  case JOptionPane.OK_OPTION : // premuto "Ok"
  case JOptionPane.CLOSED_OPTION : // chiuso il popup
}
```

More info...

Maggiori informazioni sulla classe `JOptionPane` sono nella documentazione della Libreria di Java

Link diretto:

`http://docs.oracle.com/javase/7/docs/api/javax/swing/JOptionPane.html`

Finestre per le applicazioni

Finestre più sofisticate dei pop-up richiedono un po' di lavoro di costruzione.

Una finestra è un oggetto della classe `JFrame`

Ogni elemento che deve comparire nella finestra (bottoni, testo, caselle di input, ecc...) è un oggetto diverso

- Esistono diverse classi per i diversi tipi di elementi:
  - `JButton` per i bottoni
  - `JLabel` per il testo
  - `JTextField` per i campi di input
  - ....

Tutti gli oggetti creati devono essere inseriti nell'oggetto della classe `JFrame`

- La disposizione degli oggetti nella finestra può seguire diversi modelli di layout che non vedremo...

Eventi (1)

Supponiamo di aver costruito correttamente la finestra con tutti gli elementi necessari

Il programma ora rimane in attesa che l'utente faccia qualcosa...

- prema un bottone
- inserisca un valore
- ....

Eventi (2)

Quando l'utente interagisce con l'interfaccia grafica fa scaturire un evento

Affinchè ai vari eventi corrispondano gli effetti che vogliamo, dovremo implementare dei gestori (o action listeners)

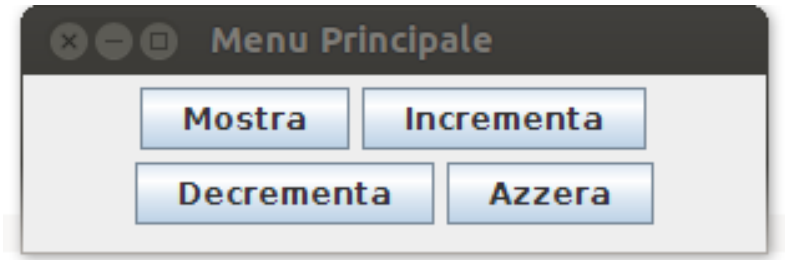
- ogni oggetto avrà il suo gestore, che viene richiamato quando l'utente interagisce con quell'oggetto
- ad esempio: ogni bottone avrà il suo gestore, che contiene i comandi da eseguire quando quel bottone viene premuto

Esempio 2: Un contatore (1)

Esempio: Supponiamo di avere una classe `Contatore` con le seguenti funzionalità:

- crea un nuovo contatore inizializzato a 0
- incrementa il contatore di 1
- decrementa il contatore di 1
- azzera il contatore

Vogliamo costruire la seguente interfaccia grafica:



Esempio 2: Un contatore (2)

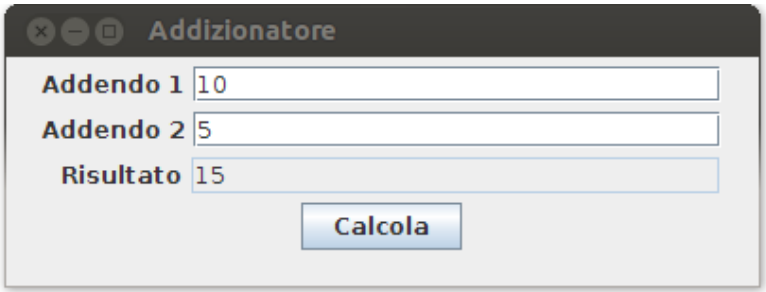
Per ottenere il risultato voluto dobbiamo:

1. Creare la finestra con `JFrame`
2. Creare i quattro bottoni con `JButton`
3. Implementare un gestore per ogni bottone
   - ogni gestore sarà una classe che implementa `ActionListener`
4. Inserire i `JButton` nel `JFrame`
   - in realtà dovremo raccogliere i `JButton` in un contenitore `JPanel`

Vedere il codice dell'applicazione `ContatoreGUI`
nella pagina web del corso

Esempio 3: somma di due numeri

Esempio: Vogliamo realizzare un programma che sommi coppie di numeri con questa interfaccia:



Dobbiamo:

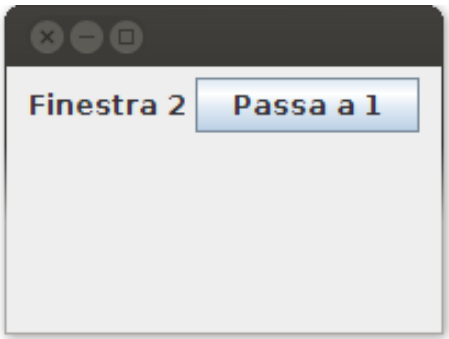
- Usare le varie classi corrispondenti ai vari tipi di elementi inclusi nella finestra
- Prevedere un gestore dell'evento di pressione del bottone che possa accedere ai valori inseriti nelle caselle di testo (per sommarli)

Vedere il codice dell'applicazione `Addizionatore`
nella pagina web del corso

Esempio 4: due finestre

Esempio: Vogliamo realizzare un programma che passi da una finestra a un'altra (e ritorno) premendo un bottone.

Le due finestre devono avere il seguente aspetto:



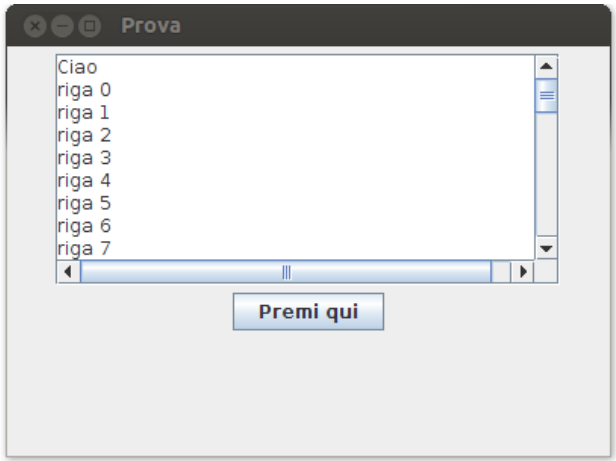
Dobbiamo:

- Usare due oggetti `JFrame`
- Consentire ad ognuno dei due oggetti di poter accedere all'altro
- Il gestore dell'evento di pressione del bottone renderà invisibile la finestra corrente e visibile l'altra

Vedere il codice dell'applicazione `Finestra1`
nella pagina web del corso

Esempio 5: barre di scorrimento

Esempio: Una finestra che include un'area di testo con barre di scorrimento



Vedere il codice dell'applicazione `Scrollabile`
nella pagina web del corso