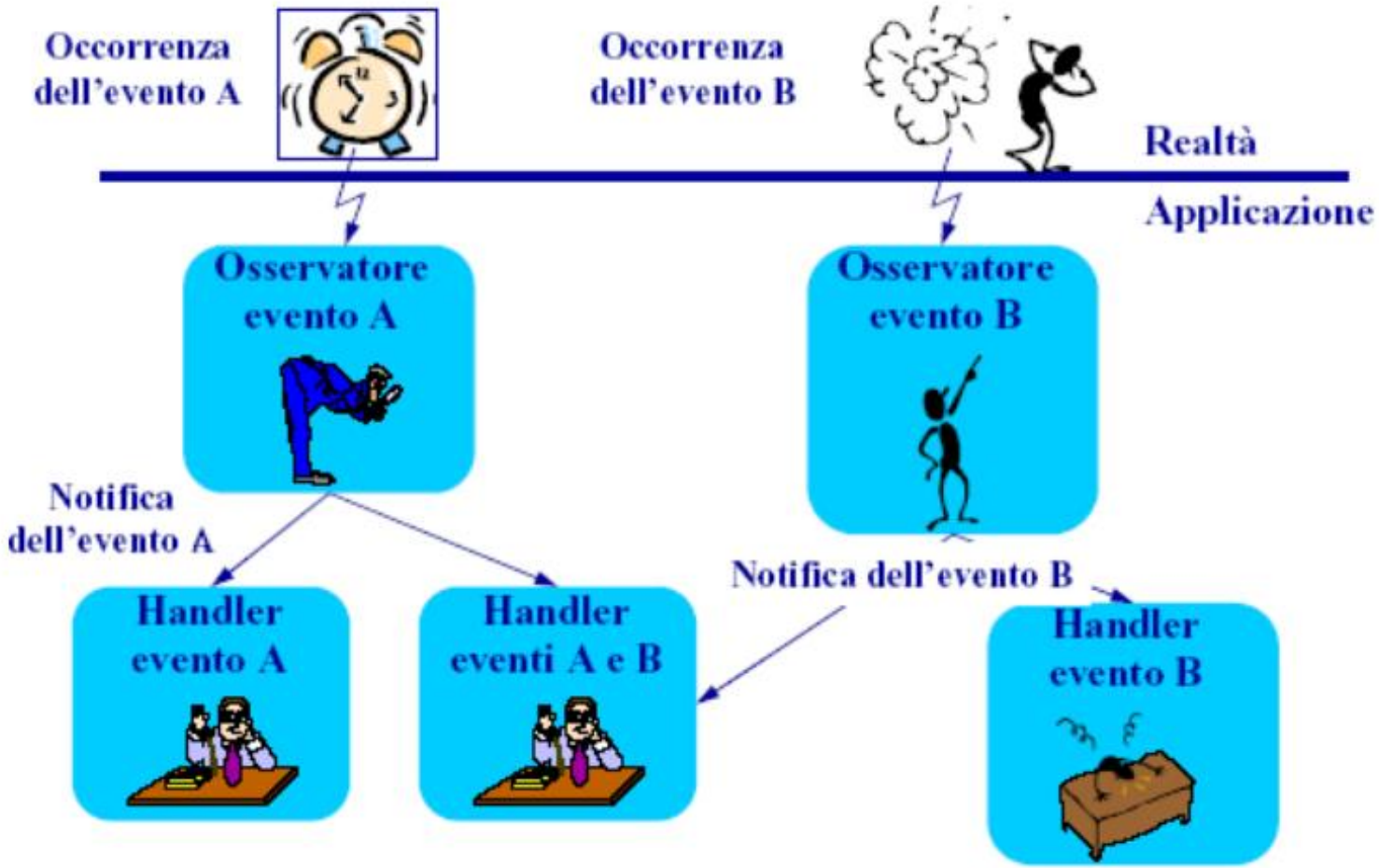
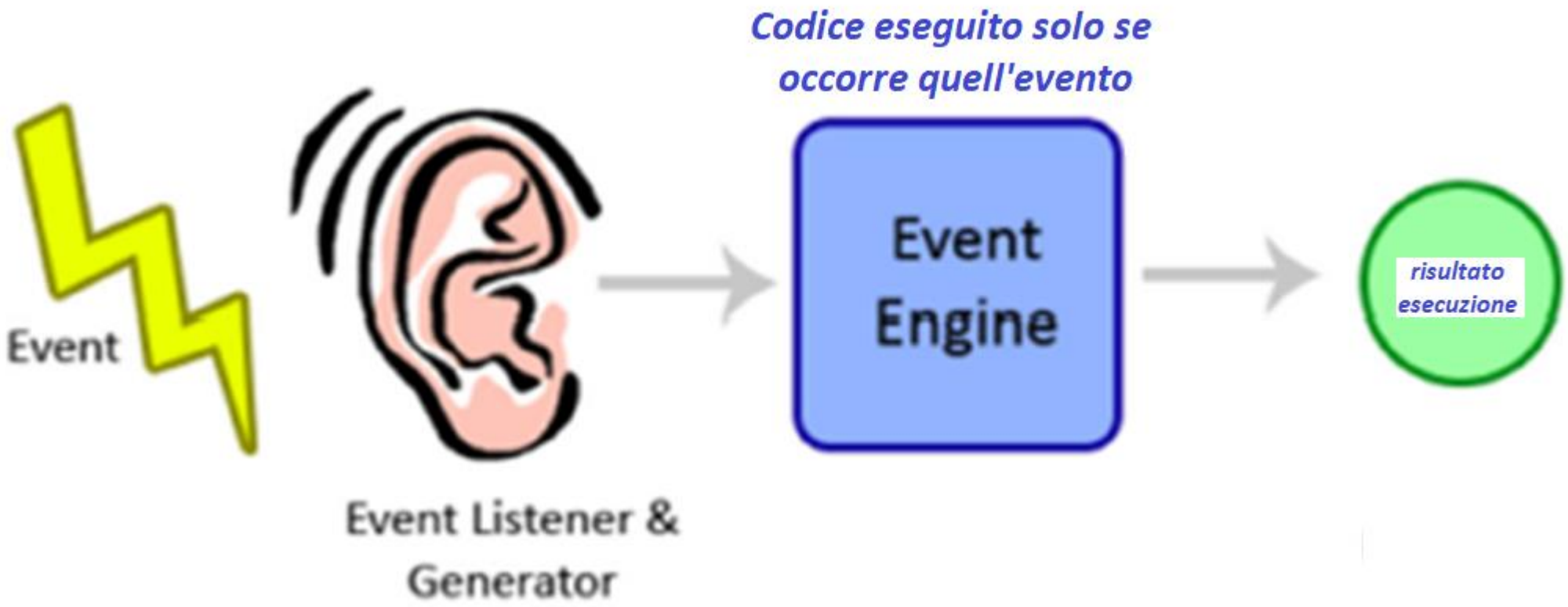
Paradigma di programmazione event-driven

Evento



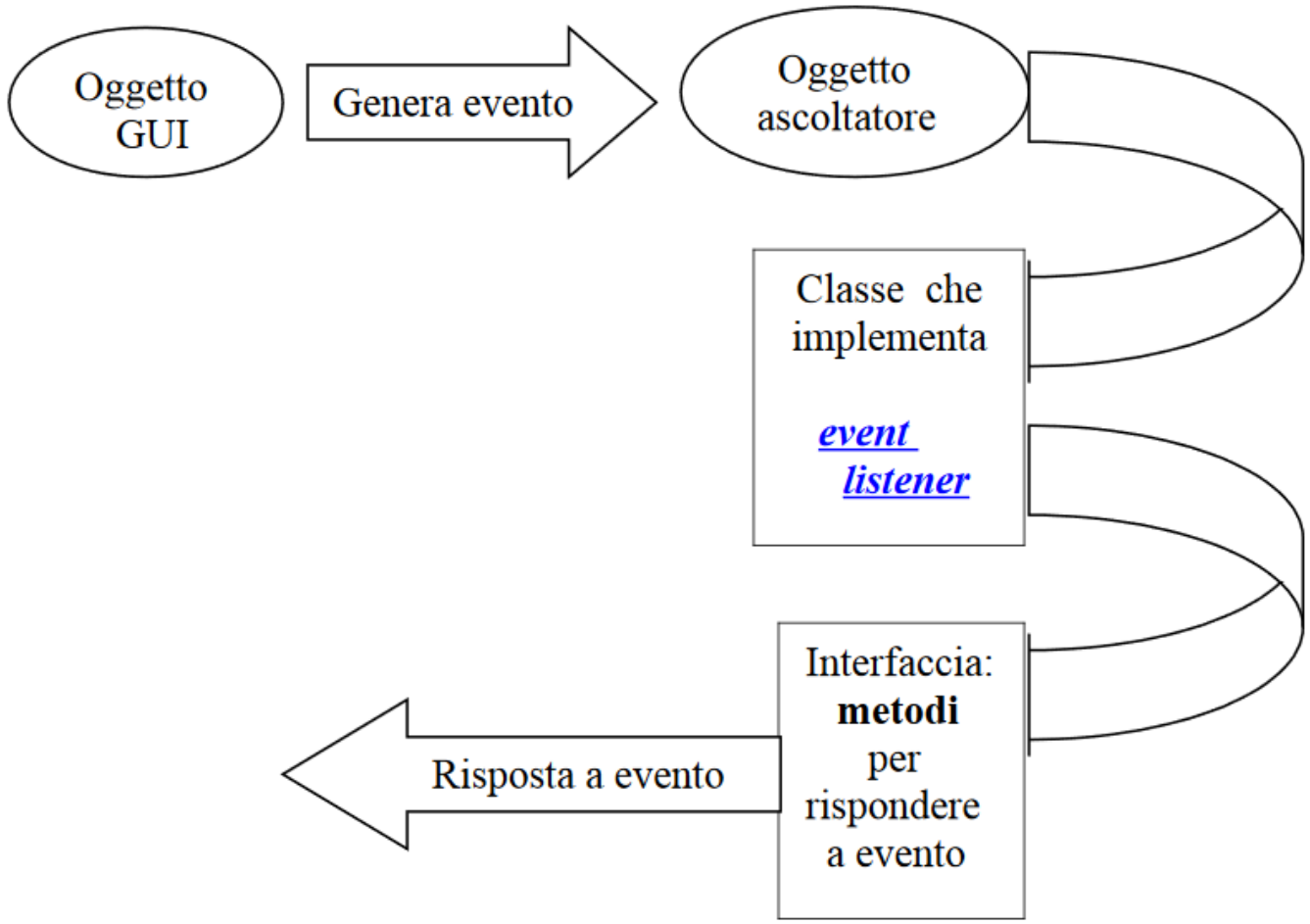
Un ***evento*** è un messaggio che il sistema genera in risposta a un'azione effettuata generalmente dall'utente quando interagisce con l'applicazione.

Programmazione guidata da evento

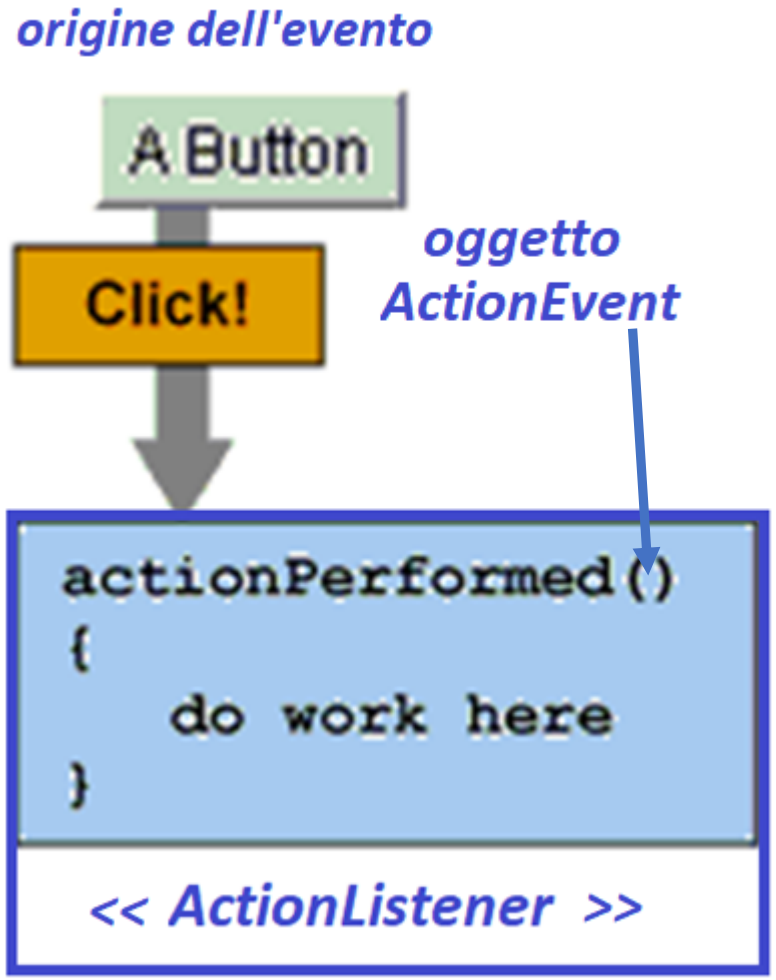


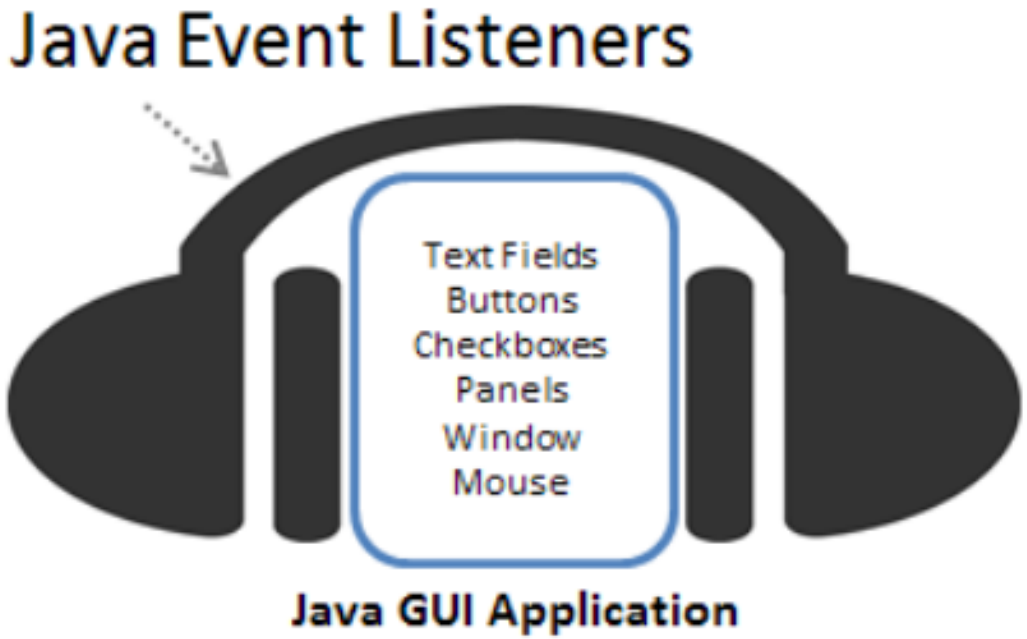
Programmazione event-driven in Java

Modello a delega in Java

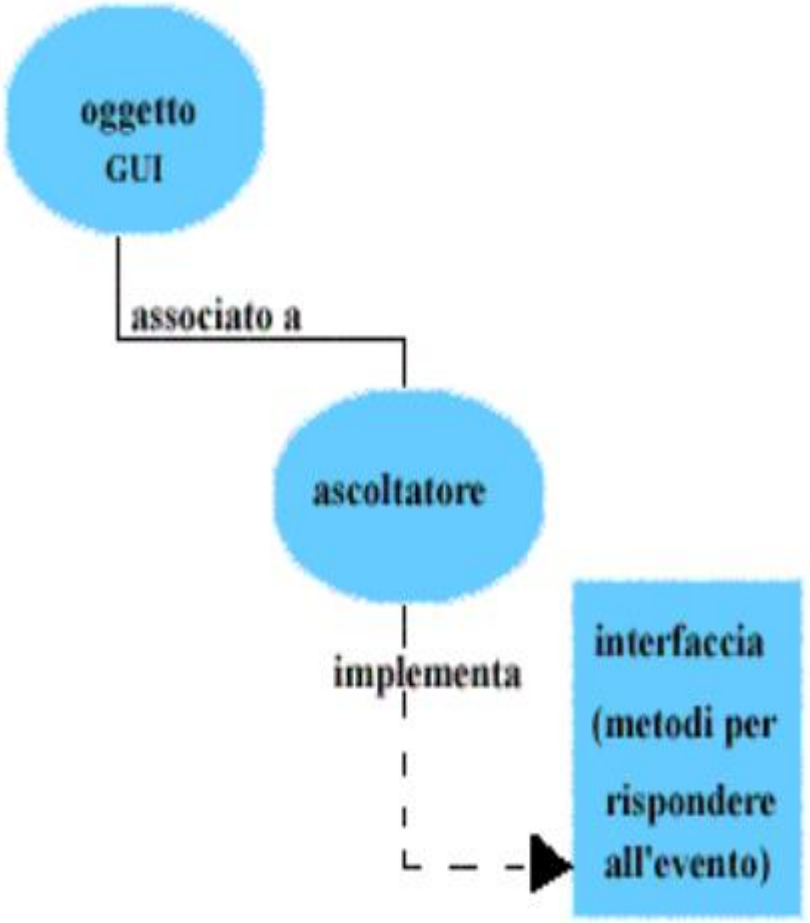


Acoltatori in Java: modello a delega





Passi per gestire gli eventi in Java



- Decidere **quali eventi interessano** l'applicazione. Tale fase si traduce nella scelta dell'***interfaccia*** di ascolto da implementare.
- Decidere qual è l'**origine** dell'evento: il componente o contenitore che può generare (***notificare***)l'evento e creare tale **oggetto GUI** (ogni componente ad eccezione di LABEL o un contenitore ad esempio tutta l'applicazione)
- Creare un **oggetto** della **classe di ascolto**
  - L'oggetto ascoltatore può essere ad esempio **tutta l'applicazione** o una sua parte che implementa (**implements**) la classe predefinita d'ascolto di quell'evento che ne definisce l'interfaccia (metodi di risposta a quell'evento).
  - L'oggetto ascoltatore può essere creato come istanza di una **classe di ascolto** definita dall'utente che implementa (**implements**) la classe predefinita d'ascolto di quell'evento che ne definisce l'interfaccia (metodi di risposta a quell'evento)
- **Collegare** all'**origine** dell'evento **l'oggetto della classe d'ascolto** di quell'evento cioè **registrare l'ascoltatore**
- **Implementare** almeno un metodo dell'interfaccia per rispondere a quell'evento creando il codice di gestione che serve.

Esempio: evento di azione

- **Origine** dell'evento di azione: un JButton

- Interfaccia da implementare **AcionListener: si sceglie tutta l'applicazione che implementi l'unico** metodo astratto

public void **actionPerformed**(**ActionEvent** ev) ;

codice che cambia colore di sfondo del bottone

Codice (1)

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

public class BottoneRB implements ActionListener { // classe ascoltatore
                                                   // che implementa interfaccia per eventi di azione
private JFrame f;
private Container c;
private JPanel p;
private JButton b;

public BottoneRB(){                                // costruttore
    f = new JFrame ("Cambio di colore");
    c = f.getContentPane();
    p = new JPanel();
    b = new JButton("Premi");
    b.addActionListener (this); // lega la classe di ascolto cioè l'applicazione stessa
                                // all'origine dell'evento cioè il pulsante

    // prosegue …….
```

Codice (2)

```
        b.setBackground (Color.red);
        p.add(b);
        c.add(p);
        f.setLocation(200,200);
        f.setSize(400,100);
        f.setBackground(Color.white);
        f.setVisible(true);
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); // senza esplicita gestione di eventi di finestra
}
public void actionPerformed (ActionEvent ev) {   // implemento unico metodo dell'interfaccia di ascolto
        b.setBackground (Color.cyan); // cambia colore di sfondo del bottone
}
public static void main (String [] args) {
         new BottoneRB ();   // creo oggetto anonimo
}
} // fine applicazione
```

Eventi: primi metodi utili

- Per ***qualsiasi evento***:

  Object ***getSource()*** restituisce la componente su cui si è verificato l'evento e si può scoprire con "*instanceof*" a quale classe appartiene.

- Per **eventi di azione**:

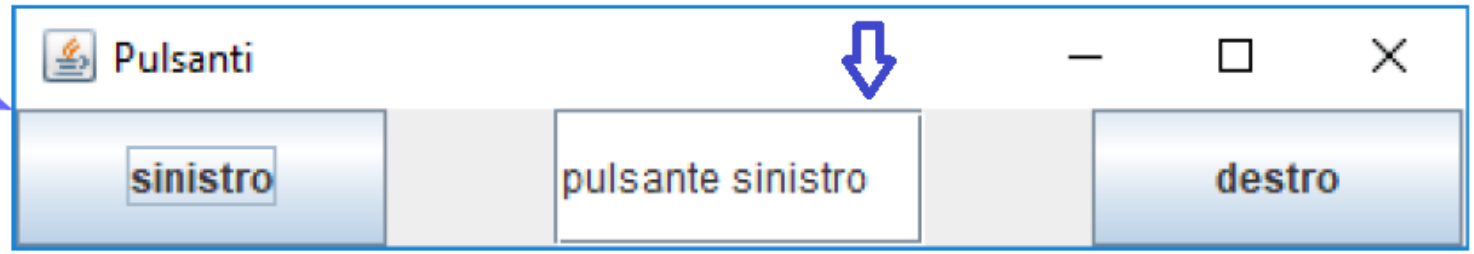
  String ***getActionCommand()*** restituisce la stringa contenuta nell'oggetto (ad esempio il testo all'interno del bottone)

Problema: un GUI con due pulsanti ed una casella di testo

```
public void actionPerformed (ActionEvent e)  {

    String bottone = e.getActionCommand();  // recupera testo mostrato da pulsante
     if (bottone.equals ("sinistro") )
           t.setText("pulsante sinistro");
     else
           t.setText("pulsante destro");
}
```

casella di testo
JTextField di nome t

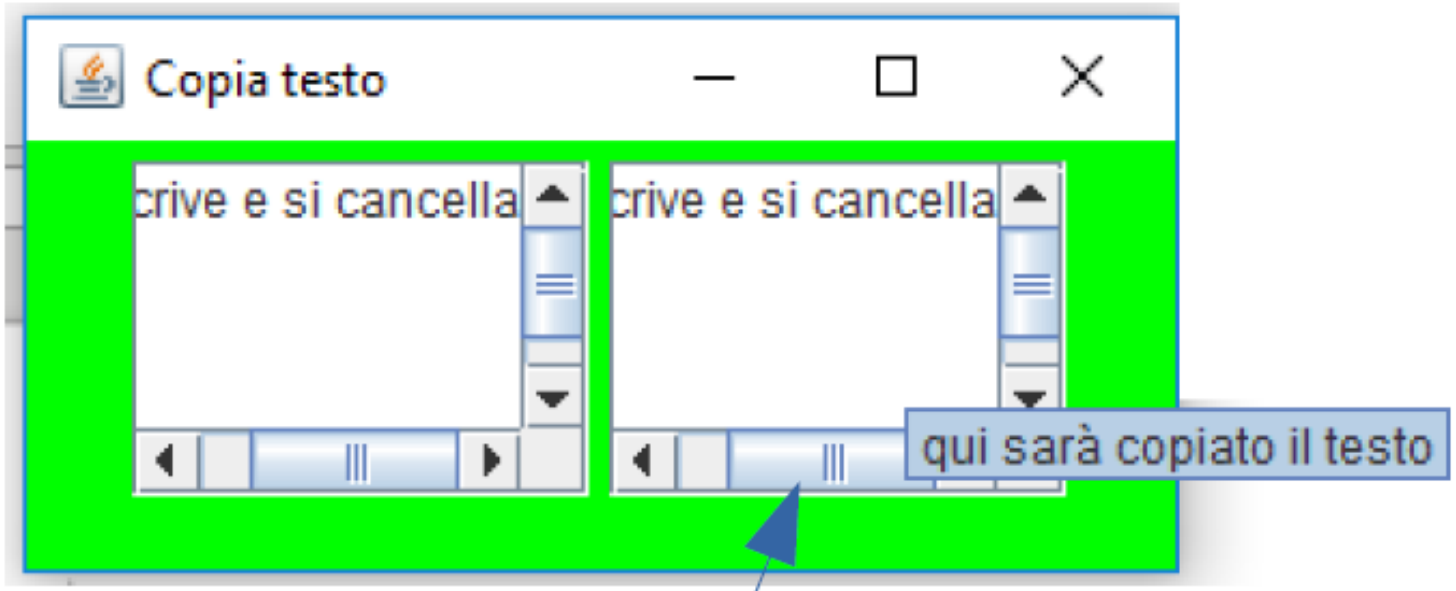


Eventi SWING

Aree di testo ed eventi associati al *documento*

⇨ **DocumentListener**

import **javax.swing.event.***; // non esiste DocumentAdapter



Esercitarsi