Evento sulla modifica delle caselle di testo

La casella di testo, ma anche l'area di testo, è per sua natura interessata a vari tipi di evento prodotti dalle azioni degli utenti. Il clic del mouse sulla casella di testo la abilita per l'inserimento del testo mentre la pressione dei tasti della tastiera modificano il contenuto della casella.

Nel seguito descriveremo quali sono le modalità per gestire due tipi di evento:
1) la pressione del tasto *Invio*,
2) i cambiamenti nel contenuto della casella di testo.

L'ascoltatore dell'evento di pressione del tasto *Invio* è l'**ActionListener**. La classe utilizzata per la gestione di questo evento deve implementare il metodo **actionPerformed** e assume la seguente struttura generica:

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

class GestoreCasella implements ActionListener
{
   public void actionPerformed(ActionEvent e)
   {
      // Operazioni di gestione dell'evento
   }
}
```

Il gestore dell'evento deve essere registrato nella componente che intende utilizzarlo, nel nostro caso la casella di testo. Per esempio, se *txtNome* è un *JTextField*, l'istruzione per associare a questa casella di testo un gestore è

```
txtNome.addActionListener(new GestoreCasella());
```

L'ascoltatore che monitora i cambiamenti nel contenuto della casella di testo, o dell'area di testo, è il **DocumentListener**. Gli eventi sono gestiti ed elaborati dai seguenti metodi:

- **insertUpdate**: richiamato quanto viene aggiunto un nuovo carattere,
- **removeUpdate**: richiamato quanto un carattere viene cancellato,
- **changedUpdate**: richiamato quando lo stile di una parte del documento cambia.

La classe utilizzata per la gestione dei cambiamenti nel contenuto della casella di testo deve implementare i metodi precedenti e assume la seguente struttura generica:

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

class GestoreDocumento implements DocumentListener
{
   public void changedUpdate(DocumentEvent e)
   {
      // Operazioni di gestione dell'evento
   }

   public void removeUpdate(DocumentEvent e)
   {
      // Operazioni di gestione dell'evento
   }
```

```
    public void insertUpdate(DocumentEvent e)
    {
      // Operazioni di gestione dell'evento
    }
}
```

Si noti che l'interfaccia *DocumentListener* è contenuta nel package *javax.swing.event* che deve essere aggiunto alla classe con l'istruzione *import*.
Il gestore dell'evento deve essere registrato nella componente che intende utilizzarlo. Per esempio, se *txtDesc* è un *JTextArea*, l'istruzione per associare a questa area di testo un gestore è

```
txtDesc.getDocument().addDocumentListener(new GestoreDocumento());
```

I seguenti progetti realizzano la gestione dei due tipi di evento generati su una casella di testo e su un'area di testo.

ESEMPIO

Realizzare un programma che calcola il quadrato di un numero.

La finestra dell'applicazione contiene una casella di testo in cui l'utente può inserire un numero. Quando viene premuto il tasto *Invio*, si apre una finestra di dialogo che visualizza il quadrato del numero inserito.
Il costruttore del *GestoreCasella* riceve come parametro la casella di testo perché, all'interno del metodo *actionPerformed*, deve utilizzarla per leggere il valore inserito dall'utente, con il metodo *getText*, e convertirlo in numero.

IMPLEMENTAZIONE DELLA CLASSE (*GestoreCasella.java*)

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

class GestoreCasella implements ActionListener
{
  private JTextField txt;

  public GestoreCasella(JTextField txt)
  {
    this.txt = txt;
  }

  public void actionPerformed(ActionEvent e)
  {
    int numero;

    numero = Integer.parseInt(txt.getText());

    JOptionPane.showMessageDialog(null,
      "" + (numero*numero),
      "Quadrato del numero",
      JOptionPane.INFORMATION_MESSAGE);
  }
}
```

PROGRAMMA JAVA (*ProgCasella.java*)

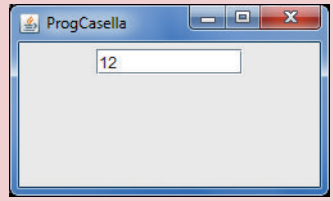
```
import javax.swing.*;
import java.awt.*;

class ProgCasella
{
   public static void main(String argv[])
   {
      JFrame f = new JFrame("ProgCasella");
      JPanel p = new JPanel();
      JTextField txtNumero = new JTextField(10);

      txtNumero.addActionListener(new GestoreCasella(txtNumero));

      p.add(txtNumero);

      f.getContentPane().add(p);
      f.setSize(250,150);
      f.setLocation(100,100);
      f.setVisible(true);
   }
}
```

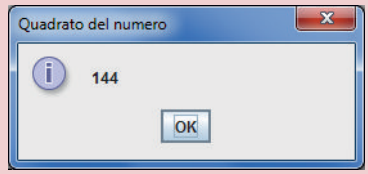
Il programma mostrato precedentemente può essere realizzato con un'unica classe, inserendo il gestore dell'evento direttamente come parametro del metodo *addActionListener*. Per leggere il valore della casella di testo *txtNumero* all'interno del gestore, si deve dichiarare l'attributo con la parola chiave *final*. Il codice completo dell'unica classe è riportato di seguito.

PROGRAMMA JAVA (*ProgCasellaUnico.java*)

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

class ProgCasellaUnico
{
   public static void main(String argv[])
   {
      JFrame f = new JFrame("ProgCasella");
      JPanel p = new JPanel();

      final JTextField txtNumero = new JTextField(10);

      txtNumero.addActionListener(new ActionListener() {
         public void actionPerformed(ActionEvent e)
         {
            int numero;

            numero = Integer.parseInt(txtNumero.getText());
```

```
            JOptionPane.showMessageDialog(null,
              "" + (numero*numero),
              "Quadrato del numero",
              JOptionPane.INFORMATION_MESSAGE);
          }
        });

        p.add(txtNumero);

        f.getContentPane().add(p);
        f.setSize(250,150);
        f.setLocation(100,100);
        f.setVisible(true);
    }
}
```





ESEMPIO

Realizzare un programma che conta il numero di caratteri inseriti in un'area di testo.

La finestra è organizzata con un *BorderLayout*. Nella zona in alto si dispone l'area di testo in cui l'utente può inserire i messaggi. La zona in basso contiene un pannello che mostra un'etichetta e una casella di testo in cui vengono visualizzati i caratteri inseriti.
Il costruttore del *GestoreDocumento* riceve come parametri sia l'area di testo che la casella di testo. Ha bisogno di entrambi i riferimenti per poter calcolare il numero di caratteri inseriti, con il metodo *length*, e per visualizzarli nella casella di testo, con il metodo *setText*.
Tutti e tre i possibili eventi di modifica, attivati ogni volta che l'utente modifica il messaggio, richiamano il metodo *aggiorna*.

IMPLEMENTAZIONE DELLA CLASSE (*GestoreDocumento.java*)

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

class GestoreDocumento implements DocumentListener
{
   private JTextArea msg;
   private JTextField conta;

   public GestoreDocumento(JTextArea msg, JTextField conta)
   {
      this.msg   = msg;
      this.conta = conta;
   }

   public void changedUpdate(DocumentEvent e)
   {
      aggiorna();
   }
```

```
   public void removeUpdate(DocumentEvent e)
   {
      aggiorna();
   }

   public void insertUpdate(DocumentEvent e)
   {
      aggiorna();
   }

   public void aggiorna()
   {
      int num = msg.getText().length();

      conta.setText(""+num);
   }
}
```

PROGRAMMA JAVA (*ProgDocumento.java*)

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

class ProgDocumento
{
   public static void main(String argv[])
   {
      JFrame f = new JFrame("ProgDocumento");
      JPanel p = new JPanel();
      JPanel pConta = new JPanel();
      JTextArea  txtMsg   = new JTextArea(7,20);
      JTextField txtConta = new JTextField(5);

      GestoreDocumento g = new GestoreDocumento(txtMsg, txtConta);
      txtMsg.getDocument().addDocumentListener(g);

      pConta.setLayout(new BorderLayout());
      pConta.add(new JLabel("Caratteri:"), "West");
      pConta.add(txtConta, "Center");

      p.setLayout(new BorderLayout());
      p.add(txtMsg, "North");
      p.add(pConta, "South");

      f.getContentPane().add(p);
      f.setSize(350,250);
      f.setLocation(100,100);
      f.setVisible(true);
   }
}
```