Eventi del mouse

Esempio di **applet** con gestione **eventi del mouse:**

```
// ClickMe.java  versione Applet
 import java.applet.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

// classe di ascolto
public class ClickMe extends JApplet implements MouseListener {   //  tipo di evento ad esempio
                                                                   //  pressione del mouse
   private Spot spot = null;
   private static final int RADIUS = 7;

   public void init()  {
      addMouseListener(this);   // per collegare ascoltatore all'origine dell'evento: tutta l'applet
    }

   public void paint(Graphics g) {

         //draw a black border and a white background
        g.setColor(Color.white);
         g.fillRect(0, 0, getSize().width - 1, getSize().height - 1);
                                              //oppure  g.fillRect(0, 0,  getWidth() - 1, getHeight()- 1);
        g.setColor(Color.black);
         g.drawRect(0, 0, getSize().width - 1, getSize().height - 1);
                                                  //oppure  g. drawRect(0, 0,  getWidth() - 1, getHeight()- 1);
         //draw the spot
        g.setColor(Color.red);
         if (spot != null) {
            g.fillOval(spot.x - RADIUS, spot.y - RADIUS, RADIUS * 2, RADIUS * 2);
      }
   }
   public void mousePressed(MouseEvent event) {          //  tipo evento: pressione del mouse
      if (spot == null) {  spot = new Spot(RADIUS);  }
         spot.x = event.getX();
         spot.y = event.getY();
         repaint();
   }
   public void mouseClicked(MouseEvent event) {}      // altri metodi dell'interfaccia di ascolto
   public void mouseReleased(MouseEvent event) {}     // per evitare di scrivere tali segnature
   public void mouseEntered(MouseEvent event) {}      // di metodi si usa classe Adapter che
   public void mouseExited(MouseEvent event) {}       // implementa interfaccia MouseListener
}

// Spot.java
public class Spot {
   public int size;
   public int x, y;

   public Spot(int intSize) {
      size = intSize;
      x = -1;
      y = -1;
   }
}
```

Nel codice HTML le **dimensioni obbligatorie** non saranno nulle:

- con codice più datato (IE):

  <**object** code ="GuiA.class" width= "200" height= "100">></object>

- con codice HTML che carica ed esegue **Applet** con "*Java Plug in*" per consentire al browser di *usare un interprete Java esterno (più aggiornato)* e adatta anche a Mozilla – FireFox ed altri browser che non supportano il tag <object>:

```
<html>
<head><title>Uso Eventi</title></head>
<body>
<p>Carica ed esegue Applet con uso di Java Plug-in (anche per NN versioni precedenti alla 6):
<p>appare uno spot rosso nel punto dove fai click col mouse
<p><object classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93"
      width="200" height="100">
          <param name="code" value="ClickMe.class">
          <param name="type" value="application/x-java-applet;version=1.6">
          <comment><embed type="application/x-java-applet" width="200" height="100"
                                code="ClickMe.class">
                    </embed>
    </comment>
</object>
</body>
</html>
```

NB: **getWidth**() e **getHeight**() per recupero dimensioni senza creare oggetto Dimension con getSize()

Esempio con gestione **eventi del mouse** con **adattatore** (classe di raccordo *Adapter*) che evita di dover scrivere tutte le segnature dei metodi dell'*interfaccia* di ascolto :

```
// Pallini.java  versione Applet
import java.applet.*;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Pallini extends JApplet{

      private int xP,yP;
      public void init()   {

            MiaClasseAscoltoMouse A1 = new MiaClasseAscoltoMouse ();
           addMouseListener (A1);     // lega la classe di ascolto all'origine dell'evento: cioè
                                      // tutta l'applet (è sottinteso this : manca oggetto origine)
       }

       public void paint(Graphics g) {
           g.setColor(Color.red);
           g.fillOval(xP,yP,20,20);
        }
```

```
        public void test()   {                          //  per test  in ambiente JCreator
              int x = getLocationOnScreen().x;       //  ritorna l'ascissa del punto
              int y = getLocationOnScreen().y;       //  ritorna l'ordinata del punto
             System.out.println( "punto (" + xP + "," + yP +")" );
      }

// classe interna cioè innestata non statica
private class MiaClasseAscoltoMouse extends MouseAdapter {

   public void mousePressed (MouseEvent e) {
        xP = e.getX();          // accede alla variabili istanza xP e yP della classe esterna
        yP = e.getY();          // recupera gli attributi dell'oggetto di tipo Point "puntato" dal mouse
                                // cioè le coordinate x ed y
        repaint();

        test();   //   per visualizzare coordinate
       }
  }
}// fine classe Pallini
```

Con codice HTML:

```
<html>
        <head><title>Uso Classe Adapter</title></head>
        <body>
                Se premi con il mouse (all'interno di una zona) compare un pallino rosso <br>
                <object code ="Pallini.class" width = "500" height = "400"></object>
        </body>
</html>
```

NB: per estrarre coordinate sullo schermo **getLocationOnScreen**() della classe **java.awt.Component** può essere invocato per recuperare le coordinate del punto in alto a sinistra ad esempio dell'applet; il metodo ritorna infatti un oggetto di tipo **Point** con attributi **x** ed **y** ed è applicabile al componente corrente

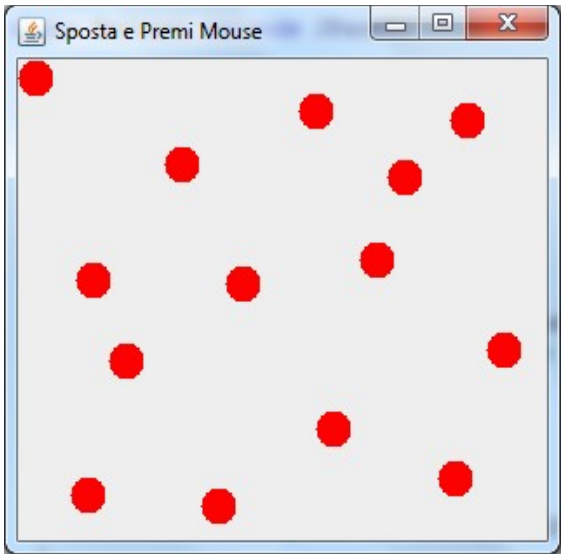
Classi innestate e interne: definizione

Una classe innestata non è altro che una classe che viene definita all'interno di un'altra classe.

Il vantaggio di implementare una classe all'interno di un'altra, riguarda principalmente il **risparmio di codice**. Infatti, la classe innestata ha accesso alle variabili di istanza della classe in cui è *innestata*.

N.B. : se compiliamo una classe che contiene una classe innestata, saranno creati due file:
"nomeClasseEsterna.class" e "nomeClasseEsterna**$**nomeClasseinnestata.class".

N.B. : fino alla versione 1.3 di Java, il termine "classe innestata" non era stato ancora adottato. Si parlava invece di "classe interna" ("inner class").
*Dalla versione 1.4 in poi, la Sun ha deciso di sostituire il termine "classe interna", con il termine "classe innestata" o classe annidata ("**nested class**"). Il termine "classe interna" deve ora essere utilizzato solo per le classi **innestate** che **non** sono **dichiarate statiche**. In realtà la stragrande maggior parte delle volte si utilizzano classi interne.*




```
/**
 *  Pallini.java
 *
 * Pallini application
 *
 * @author classi
 * @version 1.00 2011/4/14
 */
import java.awt.event.*;
import java.awt.*;
import javax.swing.*;

public class Pallini extends JPanel{

        private int x,y;
        private JFrame f ;

        public Pallini()   {

                f= new JFrame();
                f.setContentPane(this);

                MiaClasseAscoltoMouse A1 = new MiaClasseAscoltoMouse ();
                addMouseListener (A1);     // lega la classe di ascolto all'origine dell'evento: cioè tutta
                                           // l'applicazione  (this è sottinteso)

                f.setTitle("Sposta e Premi Mouse");
                f.setLocation(300,300);
                f.setSize(320,300);
                f.setVisible(true);
                f.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);
        }

        public void paintComponent(Graphics g) {
                 // super.paintComponent(g);   // senza ricostruzione dello sfondo: non aggiorna
                 g.setColor(Color.red);
                 g.fillOval(x,y,20,20);
        }

        private class MiaClasseAscoltoMouse extends MouseAdapter { // classe interna

             public void mousePressed (MouseEvent e) {
                x = e.getX();          // accede alla variabili istanza x e y della classe
                y = e.getY();
                repaint();
               }
         }

          public static void main(String[] args) {
                   new Pallini();
           }

}// fine applicazione Pallini
```