Timer

Animazioni

Classe Timer

- Costruttore:

```
new Timer(int delay,
          ActionListener listener)
```

  - Esegue il metodo del listener ogni delay millisecondi…
- Come si usa?
  - Creare una classe interna che implementa ActionListener
    - Definire la logica della animazione
  - Creare un timer con delay e la propria classe.
- Metodi di gestione
  - Metodo **start()** avvia l'animazione
  - Metodo **stop()** termina l'animazione

Esercizio

- JFrame che ad ogni secondo cambia il colore dello sfondo in maniera randomica
  - Classe principale che estende JFrame (**classe1**)
  - Classe interna che estende ActionListener per la gestione degli eventi del Timer (**classe2**)
    - Il costruttore della **classe1** crea il Timer
    - Il metodo actionPerformed della **classe2** contiene le istruzioni per l'animazione

```
public class SimpleTimer extends JFrame {
  public SimpleTimer() {
    //crea un oggetto Timer e avvia il timer
  }
  //gestore del listener
  private class ListenerTimerColore
          implements ActionListener {
      public void actionPerformed(ActionEvent e)
      {
       //genera il colore random e
       //lo imposta come sfondo
       //del contentpane del frame
      }
  }
  public static void main(String[] a){
     //crea il frame e lo rende visibile
  }
}
```

Soluzione/1

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class SimpleTimer extends JFrame {
    Timer timer;
    public SimpleTimer() {
        //crea un oggetto Timer
        ListenerTimerColore listener =
                    new ListenerTimerColore()
        timer = new Timer(1000, listener);
        timer.start(); //avvia il timer
    }
    ......
```

Soluzione/2

```
private class ListenerTimerColore implements
   ActionListener {
     public void actionPerformed(ActionEvent e)
        {
          //genera il colore random
          int r = (int) (Math.random() * 255);
          int g = (int) (Math.random() * 255);
          int b = (int) (Math.random() * 255);

           //imposta il background random
          getContentPane().
                setBackground(new Color(r, g, b));
     }
}
```

Soluzione/3

```
public static void main(String[] args) {
        SimpleTimer f = new SimpleTimer();
        f.setSize(100, 100);

    f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        f.setVisible(true);
     }
```

Esercizio

- Modificare il precedente programma in modo da:
  - Aggiungere alla finestra due pulsanti con etichette start e stop;
  - Il primo avvia l’animazione e il secondo la interrompe.

Soluzione/1

```
private class ListenerPulsanteStart implements
  ActionListener {
      public void actionPerformed(ActionEvent e) {
         timer.start();
      }
  }

  private class ListenerPulsanteStop implements
  ActionListener {
      public void actionPerformed(ActionEvent e) {
         timer.stop();
      }
  }
```

Soluzione/2

```
public SimpleTimer() {
        //creo un mio content pane
        final JPanel contentPane = new JPanel();
        this.setContentPane(contentPane);
        //crea un oggetto Timer
        timer = new Timer(1000, new ListenerTimerColore());
        //crea i due pulsanti
        JButton start=new JButton("START");
        JButton stop=new JButton("STOP");

        //li aggiunge
        this.add(start);
        this.add(stop);

        //aggiunge i listener
        start.addActionListener(new ListenerPulsanteStart());
        stop.addActionListener(new ListenerPulsanteStop());
    }
```

Esercizio

- Realizzare l’animazione di una pallina che cade con velocità orizzontale e accelerazione verticale non nulle.
- Fermare l’animazione quando la pallina esce dalla zona visiva
- Aggiungere un pulsante che avvia/riavvia l’animazione
- Aggiungere un pulsante che interrompe l’animazione

Screenshot