

Applet

Programmazione in Ambienti Distribuiti

V 1.1 © Marco Torchiano 2005

Sommario

- Thread
 - Applet Wave
 - ◆ Animazione
 - ◆ Interazione con l'HTML
 - Applet configurabili attivi
 - ◆ Il disegno è configurabile
 - ◆ L'applet comanda il browser
 - Tris
-

Thread

- Un *thread* è un flusso di esecuzione all'interno di un processo.
 - I normali processi hanno un unico thread che inizia l'esecuzione dal metodo (o funzione) **main()**.
 - Quando si avvia un thread occorre fornire un metodo (o funzione) da cui iniziare l'esecuzione
-

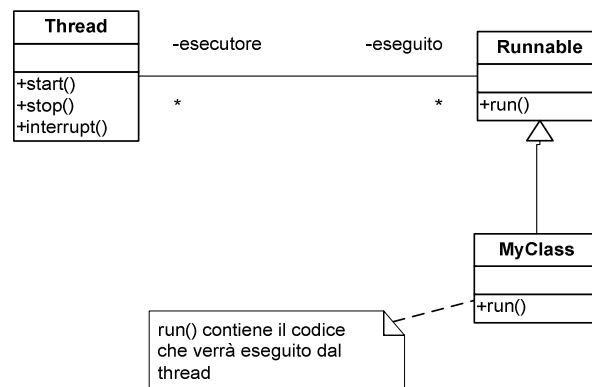
Classe Thread

- L'entità thread del sistema operativo è rappresentata dalla classe Thread.
 - Un oggetto di classe Thread rappresenta un thread che è possibile:
 - ♦ Avviare (**start()**)
 - ♦ Fermare (**stop()**) in maniera brutale
 - ♦ Interrompere (**interrupt()**)
 - Fornisce metodi statici che operano sul thread corrente (quello che esegue l'istruzione) per
 - ♦ Sospendere (**sleep(long millis)**)
-

Interfaccia Runnable

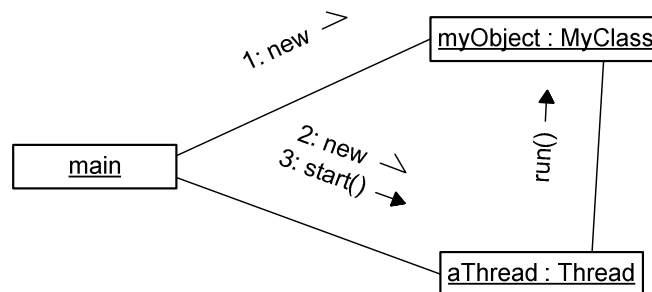
- Descrive un oggetto eseguibile da un thread.
 - Definisce il metodo **void run()** che è quello da cui per convenzione inizia l'esecuzione di un thread
 - Quando si crea un thread occorre passare (nel costruttore) un oggetto che implementa l'interfaccia runnable.
 - ♦ Se non si passa nulla si assume che l'oggetto thread stesso implementi il metodo **run()**
-

Esempio - Classi



Esempio - Oggetti

- 1 `MyClass c = new MyClass();`
- 2 `Thread t = new Thread(c);`
- 3 `t.start();`



Interruzione

- Il metodo **stop()** termina brutalmente il thread.
- Il metodo **interrupt()**
 - ◆ Imposta il thread nello stato di interrupted
 - Verificabile con il metodo **interrupted()**
 - ◆ genera un'eccezione se il thread è in pausa
 - Ad. esempio durante una `sleep()`
- Se il thread non fa pause si dovrebbe controllare lo stato di interruzione
- Altrimenti si devono intercettare le eccezioni di interruzione.

Esempio - Interruzione

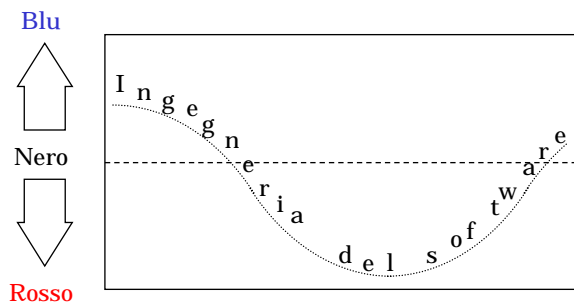
```
public void run() {
    try {
        while (true) {
            System.out.println("Here am I!");
            Thread.sleep(1000);
        }
    } catch (InterruptedException e) {
        System.out.println("Interrupted! dying!");
    }
}
```

Esempio - Interruzione

```
public void run() {
    while (true) {
        try {
            System.out.println("Here am I!");
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            System.out.println("Ignoring interrupt");
        }
    }
}
```

Applet Wave

- Si vuole realizzare un applet che mostri il testo su un'onda sinusoidale in movimento.
- Il colore del testo deve essere blu in alto, rosso in basso e nero al centro, con le relative sfumature.



TAG applet

```
<html >
<head><title>Prova di applet</title>
</head>
<body bgcolor=white>
<h1>Prova di applet</h1>
<hr>
<applet code="is1_1.class" width=500 height=200>
  <param name=lbl value="Prova">
</applet>
</body>
</html >
```

Esempio di applet

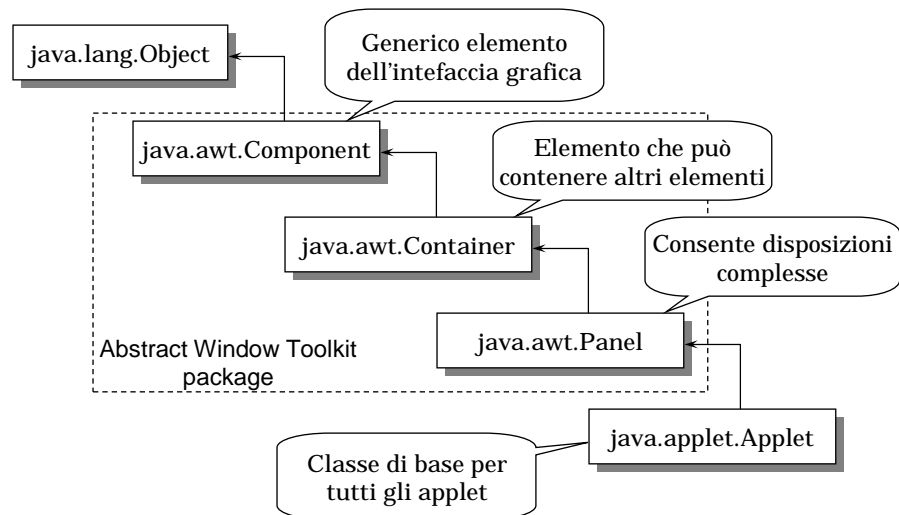
- Tutti gli applet derivano dalla classe **Applet** (o JApplet se si usano le swing).

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

```
public class is1_1
extends java.applet.Applet
implements Runnable {
    Thread run_me;
    String lbl;
    Font font;
    int speed;
    float offset;
    ... }

```

La classe Applet



Esempio

- Il metodo **init** viene chiamato dal browser (o dall'applet viewer) per informare l'applet che è stato caricato nel sistema.

```
public void init() {  
    font = new java.awt.Font("TimesRoman",  
        Font.PLAIN, 40);  
    String att = getParameter("speed");  
    speed = (att == null) ? 200  
        : (1000 / Integer.valueOf(att).intValue());  
    att=getParameter("lbl");  
    lbl=(att == null)? "Ingegneria del Software":att;  
    offset=0;  
}
```

Esempio

- Il metodo **paint** viene chiamato quando è necessario ridisegnare un componente grafico.
- Riceve come parametro un oggetto Graphics con cui disegnare.

```
public void paint(Graphics g) {  
    Dimension d = size();  
    g.setColor(Color.black);  
    setBackground(Color.white);  
    g.setFont(font);  
    FontMetrics fm = g.getFontMetrics();
```

Esempio

```
float angle=offset;

delta=(float)(Math.PI/100);
Ampl=(d.height - fm.getMaxAscent()
      - fm.getDescent())/2;
y0 = Ampl + fm.getAscent();
x0 =(d.width - fm.stringWidth(lbl))/2;

int x=x0;
int i;
for(i=0; i<lbl.length(); ++i){
    double yd = Math.cos(angle);
    int y = (int)( yd * Ampl ) + y0;
```

Esempio

```
if(yd>0){
    g.setColor(new java.awt.Color((int)(yd*200), 0, 0));
}else{
    g.setColor(new java.awt.Color(0, 0, (int)(-yd*200)));
}
String letter=String.valueOf(lbl.charAt(i));
g.drawString(letter, x, y);
int w=fm.stringWidth(letter);
x+=w;
angle-=delta*w/2;
if(angle < -2*Math.PI) angle += 2*Math.PI;
}
offset += delta * fm.stringWidth(" ");
}
```

Run

- L'applet implementa l'interfaccia **Runnable**, che deve fornire il metodo **run**, eseguito da un thread.

```
public void run() {  
    while (true) {  
        try {  
            Thread.currentThread().sleep(speed);  
        } catch (InterruptedException e) { }  
        repaint();  
    }  
}
```

Start e Stop

- Il metodo **Start** è chiamato per avviare l'esecuzione ogni volta che l'applet viene rivisitato in una pagina HTML.
- Il metodo **Stop** per fermare l'esecuzione dell'applet, quando la pagina contenente l'applet viene sostituita da un'altra.

```
public void start() {  
    run_me = new Thread(this);  
    run_me.start();  
}
```

```
public void stop() {  
    run_me.stop();  
}
```

Interazione Script-Applet

```
<applet code="is1_1.class" name="Wave" ... >
...
<script Language="JavaScript">
function Incr(){ document.Wave.SpeedUp(); }
function Decr(){ document.Wave.SpeedDown(); }
</script>
```

```
public class is1_1 extends Applet {
...
public void SpeedUp()
    { if(speed > 10) speed-=10; }
public void SpeedDown()
    { if(speed < 500) speed+=10; }
... }
```

Interazione Utente-Applet

- È possibile associare ad eventi di una form chiamate ad uno script:

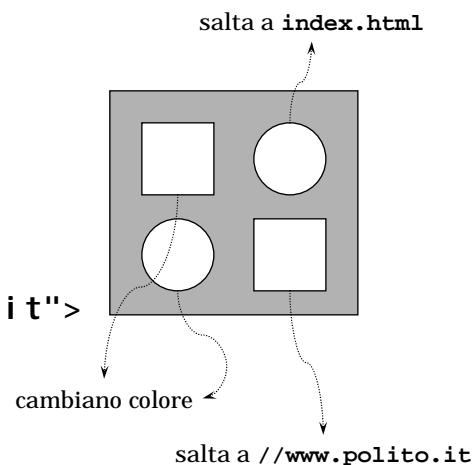
```
<form name="f1">
<input type="button" name="U" value="Up"
    onClick="Incr()" Language="JavaScript">
<input type="button" name="D" value="Down"
    onClick="Decr()" Language="JavaScript">
</form>
```

Applet con disegno attivo

- Realizzare un applet che disegni delle figure in base ai parametri che gli vengono forniti.
 - Le figura devono essere *attive* nel senso che devono rispondere ad un click in uno dei due modi seguenti:
 - ♦ cambiando colore
 - ♦ fungendo da link verso un'altra pagina html
 - Il tipo di reazione è determinato dai parametri.
-

Esempio

```
<applet ... >  
<param name=num value="4">  
<param name=shapes value=  
"R 10 10 30 30 *  
E 60 10 30 30 index.html  
E 10 60 30 30 *  
R 60 60 30 30 //www.polito.it">  
</applet>
```



Draw

```
public class Draw extends JApplet implements
    MouseLi stener, MouseMoti onLi stener {
    Vector shapes;
    boolean border;
    public Draw(){
        thi s.addMouseLi stener(thi s);
        thi s.addMouseMoti onLi stener(thi s);
    }
    public void init(){
        int num;
        shapes=new Vector(1);
        String par = getParameter("num");
        if(par == null) num=0;
        else num=Integer.parseInt(par);
```

Draw

```
par = getParameter("shapes");
if(par != null){
    StringTokenizer tk=new StringTokenizer(par, "\\t\\n\\r ", false);
    String T, PosX, PosY, SzX, SzY, url ;
    int i =0;
    try{
        for(i=0; i<num; ++i){
            T=tk.nextToken();
            PosX=tk.nextToken(); PosY=tk.nextToken();
            SzX=tk.nextToken(); SzY=tk.nextToken();
            url =tk.nextToken();
            shapes.addElement(new Shape(T, PosX, PosY, SzX, SzY, url ));
        }
    }catch(NoSuchElementExcepti on nse){}
}}
```

Shape

```
public class Shape {
    public void draw(Graphics g){
        switch(T){
            case RECT: g.drawRect(x, y, sx, sy);
                if(highlight){
                    Color c=g.getColor();
                    g.setColor(Color.red);
                    g.fillRect(x+1, y+1, sx-1, sy-1);
                    g.setColor(c);
                } break;
            case ELLI: g.drawOval(x, y, sx, sy);
                if(highlight){
                    Color c=g.getColor();
                    g.setColor(Color.red);
                    g.fillOval(x+1, y+1, sx-1, sy-1);
                    g.setColor(c);
                } break;
        }
    }
}
```

Mouse Click

```
public void mouseClicked(MouseEvent e) {
    for(int i=0; i<shapes.size(); i++){
        Shape shape=(Shape)shapes.elementAt(i);
        shape.hit(e.getX(), e.getY(), this);
    }
}

void hit(int ex, int ey, Applet ap){
    if(ex>x && ex<x+sx && ey>y && ey<y+sy){
        if(url.equals("")){
            highlight=! highlight;
            ap.repaint();
        }else{ try{
            ap.getAppletContext().showDocument(
                new URL(ap.getDocumentBase(), url));
        }catch(Exception e){ }
    }
}
```

Mouse Over

```
public void mouseMoved(MouseEvent e) {
    getAppletContext().showStatus("");
    for(int i=0; i<shapes.size(); i++){
        Shape shape=(Shape)shapes.elementAt(i);
        shape.rollOver(e.getX(), e.getY(), this);
    }
}

public void rollover(int ex, int ey,
    Applet applet) {
    if(ex>x && ex<x+sx && ey>y && ey<y+sy){
        applet.getAppletContext().showStatus(url);
    }
}
```