ut2. Programación multihilo.

La interfaz Runnable

Interfaz Runnable

- Añadir funcionalidad de hilo a una clase que deriva de otra clase, siendo esta distinta de Thread → Utilizamos interfaz **Runnable**

- Ejemplo clase deriva de Applet:

```
public class Reloj extends Applet implements Runnable {}
```

Interfaz Runnable. Estructura

```
class NomHilo implements Runnable {
    //propiedades , constructores y métodos
    public void run(){
        //acciones del hilo
        }
}
```

Ejemplo PrimerHiloR

```
public class PrimerHiloR implements Runnable{

    //propiedades
    private int x;

    //constructor
    PrimerHiloR (int x){
        this.x=x;
    }
    //run
    public void run(){
        for (int i=0;i<x;i++)
              System.out.println("En el hilo..."+i);
    }
}
```

clase llama a hilo

```
public class UsaPrimerHiloR{
    public static void main (String[] args){
        PrimerHiloR p=new PrimerHiloR(10); //creo objeto hilo
        new Thread(p).start(); //inicio ejecución
        }

}
```

```
david@david-OEM ~/pss/ut2/2 $ javac PrimerHiloR.java
david@david-OEM ~/pss/ut2/2 $ javac UsaPrimerHiloR.java
david@david-OEM ~/pss/ut2/2 $ java UsaPrimerHiloR
En el hilo...0
En el hilo...1
En el hilo...2
En el hilo...3
En el hilo...4
En el hilo...5
En el hilo...6
En el hilo...7
En el hilo...8
En el hilo...9
```

Ejemplo Uso hilo

Vamos a ver como usar hilo en un applet.

El hilo debe realizar tarea repetitiva..actualizar reloj.

Método clase applet

init()--> Instrucciones inicializar applet.

start → similar init pero es llamado al reiniciar.

paint → pinta applet

stop → se utiliza para detener hilos.

Estructura clase Usa hilo en applet

```
//librerias importo
import java.awt.*
import java.applet.*;

public class AppletThread extends Applet implements Runnable {

    //propiedades
    private Thread hilo=null;

    //método init
    public void init(){

    }

    public void start(){
        if (hilo==null){
                hilo=new Thread(this);  //creo el hilo
                hilo.start(); //lanzo hilo
            }
    }


    public void run(){
        Thread hiloActual=Thread.currentThread();
        while (hilo==hiloactual) //vble hilo apunta a hilo en ejecución
            {
            //Tarea repetitiva
            }
    }

    public void stop(){
        hilo=null;
    }

    public void paint(Graphics g){

    }

}
```

Estructura clase Usa hilo en applet

```
public class AppletThread extends Applet implements Runnable {

    //propiedades
    private Thread hilo=null;

    //método init
    public void init(){

    }

    public void start(){
            if (hilo==null){
                        hilo=new Thread(this);    //creo el hilo
                        hilo.start(); //lanzo hilo
                        }



    }

public void run(){
     Thread hiloActual=Thread.currentThread();
     while (hilo==hiloactual) //vble hilo apunta a hilo en ejecución
                {
                //Tarea repetitiva
                }
  }
  public void stop(){
    hilo=null;
  }
     public void paint(Graphics g){  }  }
```

Ejemplo Reloj en Applet

Objetivo: Crear applet con reloj y un hilo que actualice la hora cada segundo.

Clases que utilizaremos:

- Calendar → Obtener la hora.
- SimpleDateFormat → Dar formato a la hora.

Ejemplo Reloj en Applet

El método **paint( )** corre cada vez que Java carga un applet. Si el programador no ha escrito su propio método paint( ), Java utiliza uno establecido por omisión (default). El método **repaint( )** se utiliza para actualizar el applet.

En el Método Paint del applet utilizamos los siguiente métodos:

- clearRect (int x, int y, int ancho, int alto)→ Borra el rectángulo especificado, rellenandolo con color de fondo superficie color actual.
- setBackground (Color c): Establece color de fondo.
- setFont (Font fuente): Específica la fuente)
- drawString (String texto, int x, int y): pinta el texto en las posiciones x e y.

Reloj.java

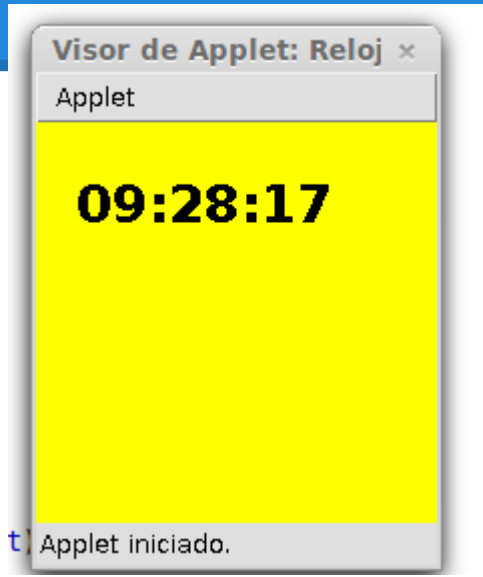
1. En Eclipse crea proyecto java llamado Applet_reloj.
2. En su interior crea clase Reloj.java
3. Va a seguir Estructura de clase usa hilo en applet.

Reloj.java

```
//Librerias importo
import java.awt.*;
import java.applet.*;
import java.text.SimpleDateFormat;
import java.util.*;

public class Reloj extends Applet implements Runnable {

    //propiedades
    private Thread hilo=null; //hilo
    private Font fuente;  //tipo de letra de la hora
    private String horaActual="";


//método init
public void init(){
    fuente=new Font("Verdana",Font.BOLD,26);

}

public void start(){
    if (hilo==null){
            hilo=new Thread(this);  //creo el hilo
            hilo.start(); //lanzo hilo
        }
}
```

Reloj.java

```
    public void run(){
        Thread hiloActual=Thread.currentThread();
        while (hilo==hiloActual) //vble hilo apunta a hilo en ejecución
            {
                //actualizo vble horaActual
                SimpleDateFormat sdf=new SimpleDateFormat("HH:mm:ss");
                Calendar cal=Calendar.getInstance();
                horaActual=sdf.format(cal.getTime());

                repaint(); //actualiza contenido applet

                //espero un segundo
                try {
                    Thread.sleep(1000);
                }catch (InterruptedException e){}
            }
    }

    public void stop(){
        hilo=null;
    }

    public void paint(Graphics g){
        g.clearRect(1,1, getSize().width,getSize().height);
        setBackground(Color.yellow); //color de fondo
        g.setFont (fuente); //fuente
        g.drawString(horaActual,20,50); // muestra la hora
    }

}
```



Actividad. Reloj2.java

Modifica el código para que se cumplan los siguientes requisitos:

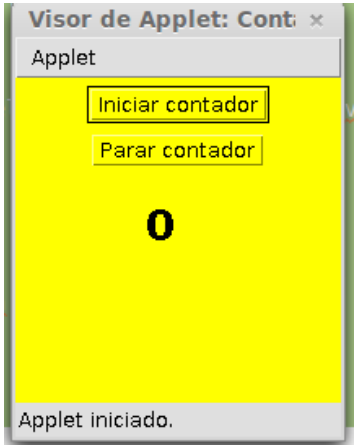
- Color de fondo sea blanco.
- La actualización se realice cada 3 segundos.
- La hora aparezca más hacia abajo.

Ejemplo ContadorApplet

Objetivo: Crear Applet que crea hilo irá incrementando en 1 un contador, tiene dos botones Iniciar contador:crea el hilo y al pulsarlo aparece continuar y Parar Contador que hace hilo se detenga.

Applet implementa clases Runnable y ActionListener (para detectar eventos de acción)

Vamos a observar como implementarlo



Ejemplo ContadorApplet

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


public class ContadorApplet extends Applet implements Runnable, ActionListener {

    //Propiedades
    private Thread h;
    private Font fuente;
    long CONTADOR=0;
    private boolean parar;
    private Button b1,b2; //botones del Applet


    public void start(){}
```

Ejemplo ContadorApplet

```
//método init
public void init(){
    setBackground(Color.yellow);//color de fondo

    //añado botón 1 y su listener
    add(b1=new Button("Iniciar contador"));
    b1.addActionListener(this);

    //añado botón 2 y su listener
    add(b2=new Button("Parar contador"));
    b2.addActionListener(this);

    fuente=new Font("Verdana",Font.BOLD,26); //tipo de letra
}
```

Ejemplo ContadorApplet

```
public void run() {
    //inicializo parar a falso
    parar=false;

    //recojo hiloActual
    Thread hiloActual=Thread.currentThread();

    while (h==hiloActual && !parar){
        try{
            Thread.sleep(300);
        }catch (InterruptedException e){e.printStackTrace();}
        repaint();
        CONTADOR++;
    }//fin while

}//fin run
```

Ejemplo ContadorApplet

```
public void paint(Graphics g){
    g.setFont(fuente);
    g.drawString(Long.toString((long)CONTADOR), 80, 100); //escribe contador

}
```

Ejemplo ContadorApplet

```
    //Parar controlar pulsación botones
    public void actionPerformed(ActionEvent e) {
        b1.setLabel("Continuar");

        if (e.getSource()==b1){//comienzo
            if (h!=null && h.isAlive()){ //si el hilo está corriendo y vivo no hago nada
                    }
            else
            {
                h=new Thread(this);
                h.start();
            }
        } else if (e.getSource()==b2){ //parada
                parar=true;
        }


    }//actionperformed

    public void stop(){

        h=null;
    }

}
```

Actividad 2.2

Se debe separar el hilo de la clase principal.

Se debe crear un applet que lance dos hilos y muestre dos botones para finalizarlos.

Define en la clase **HiloContador** un constructor que reciba el valor inicial del contador a partir del cual empezará a contar; y el método getContador() que devuelve el valor actual del contador

Actividad 2.2

El applet debe crear e inicializar 2 hilos de la clase HiloContador, cada uno empezará con un valor. Mostrará 2 botones, uno para detener el primer hilo y el otro el segundo.

Para detener los hilos usa el método stop(), hilo.stop() (Método en desuso más adelante veremos alternativa), tras pulsar cambia el botón para que muestre Finaliza Hilo 1/2