INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO

ALUMNO: Meza Vargas Brandon David.

PRÁCTICA: Práctica No. 4.

TEMA: Hilos.

OPCIÓN: Opción 4, Animación con sprites.

FECHA: 03-dic-2020

GRUPO: 2CM1

MATERIA: Programación Orientada a Objetos

INTRODUCCIÓN

Un hilo es un flujo de control dentro de un programa. Creando varios hilos podremos realizar varias tareas simultáneamente. Cada hilo tendrá sólo un contexto de ejecución (contador de programa, pila de ejecución.

En Java los hilos están encapsulados en la clase Thread. Para crear un hilo tenemos dos posibilidades:

- Heredar de Thread redefiniendo el método run().
- Crear una clase que implemente la interfaz Runnable que nos obliga a definir el método run().

En ambos casos debemos definir un método run() que será el que contenga el código del hilo. Desde dentro de este método podremos llamar a cualquier otro método de cualquier objeto, pero este método run() será el método que se invoque cuando iniciemos la ejecución de un hilo. El hilo terminará su ejecución cuando termine de ejecutarse este método run().

Aunque un programa utilice varios hilos y aparentemente estos se ejecuten simultáneamente, el sistema ejecuta una única instrucción cada vez (esto es particularmente cierto en sistemas con una sola CPU), aunque las instrucciones se ejecutan concurrentemente (entremezclándose sus éstas). El mecanismo por el cual un sistema controla la ejecución concurrente de procesos se llama planificación (scheduling). Java soporta un mecanismo simple denominado planificación por prioridad fija (fixed priority scheduling). Esto significa que la planificación de los hilos se realiza en base a la prioridad relativa de un hilo frente a las prioridades de otros.

DESARROLLO

En esta práctica se escogió la opción numero 4, la cual consiste en una animación con sprites, la animación consta del pájaro de flappy bird moviéndose de derecha a izquierda y al llegar al borde empiece desde el inicio. Esta animación se debe de hacer con hilos implementando la interfaz Runnable.

Primeramente, tenemos nuestras variables globales que usaremos en nuestra aplicación, vemos que en nuestra clase ProgDibuja extendemos a JFrame e implementamos Runnable. (ver figura 1).

```
public class ProgDibuja extends JFrame implements Runnable{

Thread hilo, hilof;
Graphics g;
BufferedImage birds[], fondos[];
BufferedImage spriteSheet=null;
int cta=0, x=0, longi=0, ctaf=0;
final int width = 34, height = 42;
final int rows = 3, cols = 1;
```

Figura 1. Variables globales

Posteriormente usaremos getSprite, esto para obtener los elementos de nuestro Sprite y, posteriormente, realizar la animación solicitada, esto lo podemos ver en la figura 2.

```
private BufferedImage getSprite(BufferedImage spriteSheet, int i, int j,
        int width, int height){
    return  spriteSheet.getSubimage(j , i , width, height);
}
```

Figura 2. Obteniendo los sprites gracias a getSprite.

En nuestro constructor cargaremos las imágenes que serán asignadas a un arreglo de tipo BufferedImage correspondientes a los fondos y a los movimientos del pájaro, estos con sus correspondientes parámetros, ancho, alto, x, y y. (ver figura 3).

```
try {
    spriteSheet = ImageIO.read(new File("C:\\Users\\PC\\Desktop\\3er Semestre ESCOM\\POO\\Programas\\Practicas\\Practica4\\Flappy-Graphics.png"));
}catch(IOException e){  System.out.println("Image not found"); }
fondos = new BufferedImage[2];
birds = new BufferedImage[3];
fondos[0] =getSprite(spriteSheet,0, 0, 290, 512);  //dia
birds [0] =getSprite(spriteSheet,750, 220, 50, 50);//alas arriba
birds [1] =getSprite(spriteSheet,800, 220, 50, 50);//alas enmedio
birds [2] =getSprite(spriteSheet,850, 220, 50, 50);//alas abajo
```

Figura 3. Carga de las imágenes.

Muy importante que al final del constructor añadamos las dos líneas de código mostradas en la figura 4. ya que al estar trabajando con hilos las necesitamos para poner el hilo en estado recién nacido y pasarlo al estado listo.

```
hilo=new Thread(this); //edo recien nacido
hilo.start(); //edo. listo
```

Figura 4. Edo. recién nacido y edo. listo.

En la figura 5 tenemos el método Paint, que se encarga de dibujar los sprites en nuestro frame.

```
public void paint(Graphics g){

    g.drawImage(fondos[ctaf], 0, 0, this);

    g.drawImage(birds[cta], x, 250, this);

}
```

Figura 5. Método Paint.

Lo que sigue en nuestro código es el método run() indispensable al trabajar con hilos, en este tenemos un ciclo while que hace que el código de el siempre se ejecute, dentro del try encontramos el código que hace que se produzca la animación, en primer lugar tenemos un incremento de la variable *cta*, encargada de llevar el control de la imagen del pájaro que se muestra, si esta variable llega a 3, se reinicia para que se vea el aleteo del ave.

De igual forma, iremos modificando la posición en x para que se produzca el movimiento, y si x es mas grande que el ancho de nuestro frame, se reinicia para que el ave aparezca desde el principio.

Al final, pausamos el hilo por 100 milisegundos para que se vea el aleteo del ave.

El código de todo lo anterior lo vemos en la figura 6;

```
public void run(){
        while(true){

            try{
                cta++;
                if(cta==3)
                    cta=0;

                this.repaint();
                x+=10;
                if(x>this.getWidth()-34)
                    x=0;
                hilo.sleep(100);
            }
            catch(Exception e){
                return;
            }

        }
```

Figura 6. Método run()

Al final tenemos nuestro main, donde creamos un objeto ProgDibuja para ejecutar nuestra aplicación. (ver figura 7).

```
public static void main(String[] args){ ProgDibuja pd=new  ProgDibuja(); }

}
```

Figura 7. Método main.

Al ejecutar el programa podemos ver algo como se ve en las figuras 8-10.

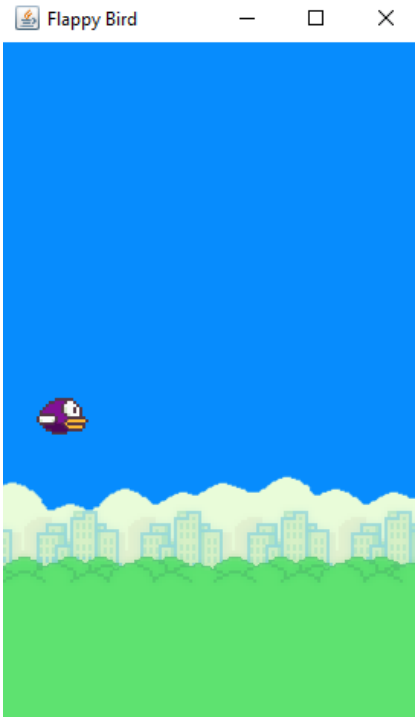

Figura 8.

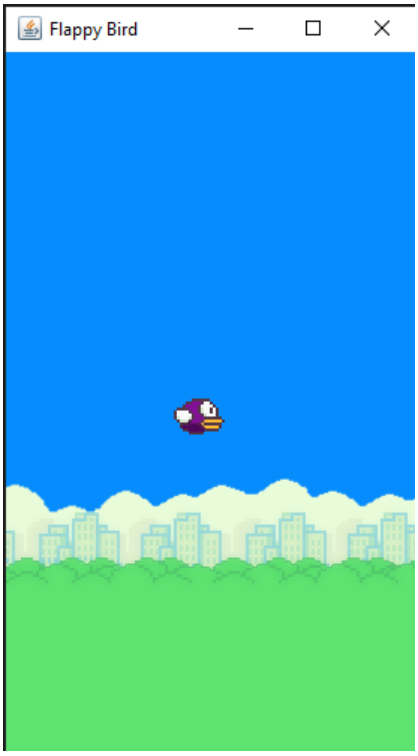

Figura 9

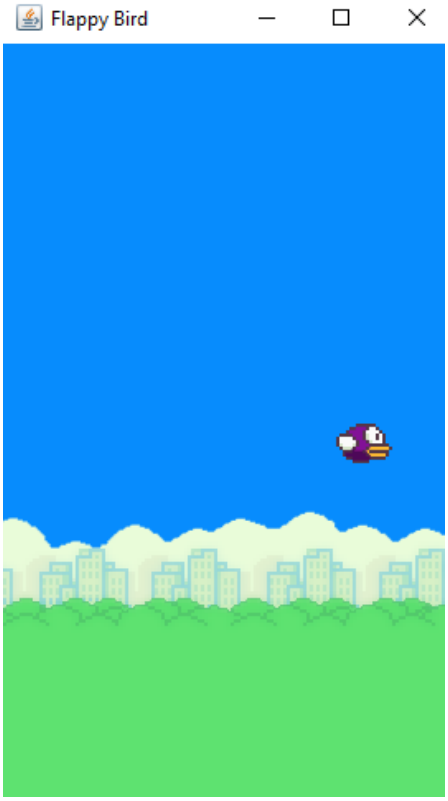

Figura 10

La ejecución se ve mejor en el video adjunto a este reporte.

CONCLUSIÓN

Como vimos en la introducción y desarrollo de la práctica, los hilos en Java nos ayudan a realizar varias tareas simultáneamente, en esta práctica se usaron sprites que hacen uso de los hilos de java gracias a la implementación de la interfaz runnable, esto provoca una pequeña modificación en la que se crean e inician los nuevos hilos.

A pesar de que el concepto de multitarea es algo sencillo de entender, es muy diferente al implementarlo en un caso práctico, pues hay veces que resulta complicado paralelizar procesos en una aplicación sin que afecte al resultado de la misma.