

Programación Orientada a Objetos

Reporte práctica 4: Hilos



Alumno: Vargas Romero Erick Efraín

Profesor: Tecla Parra Roberto

Fecha: 16-05-2017

Grupo: 2CM4

**Introducción**

Hasta el momento se han realizado implementaciones con diferentes elementos, pero los hilos es una parte fundamental, ya que podemos realizar diferentes tareas de forma concurrente, de tal manera que podemos crear programas que se ejecuten en menor tiempo y además sean más eficientes. Además, debemos tener claro que no podemos ejecutar procesos infinitos de forma concurrente ya que también el hardware tiene sus limitaciones.

**Desarrollo**

Para la práctica 4 se retomó el programa de la práctica 2 que es el “Whack a mole” pero se ha implementado el uso de los hilos. Primeramente, debemos realizar cambios en la declaración de la clase implementando la interfaz Runnable, la cual nos permite utilizar el método run en el cual realizaremos lo que requiramos en nuestros hilos.

public class Hilos **implements** Runnable**,** ActionListener**{**

Al hacer uso de hilos debemos de declarer y crear un objeto de tipo hilo, que en java tiene el nombre de Thread para nuestro programa se ha declarado como tipo private.

private Thread hilo**;**

Una vez hecho lo anterior en el constructor de nuestra clase se ha creado nuestro objeto tipo Thread lo cual deja al hilo en estado recién nacido, y finalmente para dejarlo en estado listo utilizamos el método .start()

public Hilos**(**int sleep**,** int howMany**){**

...

hilo **=** **new** Thread**(this);**

hilo**.**start**();**

**}**

Como es posible observar el constructor recibe dos parámetros de tipo entero, donde el parámetro llamado sleep es el tiempo que se detendrá el hilo y howMany serán las veces que se repetirá lo que realizará el hilo, todo esto se realizará en el método run() donde podemos observar que count y TimeSleep son variables de instancia.

Finalmente, cuando la variable count la cual se decrementa uno a uno llega a un valor igual o menor a cero termina el ciclo while y se envía un mensaje al jugador utilizando un JOptionPane

public void run**(){**

**try{**

**while(**count**--** **>** 0**){**

Thread**.**sleep**(**TimeSleep**);**

UpdateWhackAMole**();**

**}**

JOptionPane**.**showMessageDialog**(null,** "El juego ha terminado, has obtenido " **+** score **+** " puntos"**);**

**}** **catch(**Exception e**)** **{**

e**.**printStackTrace**();**

**}**

**}**

En el siguiente trozo de código es posible observar cómo es que se utilizan nuestras variables de instancia, las cuales utilizamos en el método run

public Hilos**(**int sleep**,** int howMany**){**

TimeSleep **=** sleep**;**

count **=** howMany**;**

...

**}**

**Conclusión**

Como es posible apreciar, es sencillo el utilizar hilos, pero debemos tener en cuenta algunos conceptos como lo son multitarea y multiproceso, ya que consiste en realizar varias cosas a la vez sin que nuestro resultado final sea alterado, además de que nos ayuda a paralelizar procesos. Pero a pesar de esto es importante tomar en cuenta que no siempre será sencillo el paralelizar estos procesos.