Unidad

5

DIPLOMATURA EN PROGRAMACION .NET

Ejercicios

Capítulo 10

Threads

Ejercicio 1

En este ejercicio creará un programa mulithread para familiarizarse con los conceptos del procesamiento con múltiples procesos y su naturaleza asincrónica.
Abrir la solución de Visual Studio del Capítulo 10 que contiene los proyectos de los ejercicios

1. Crear la clase ImprimirMiNombre
2. Crear el método EjecutarThread para ejecutar dentro de un ciclo que se imprima 10 veces el nombre del subproceso actual y después esperar un tiempo igual al que se obtiene de realizar la siguiente operación: `new Random(10).Next(1, 10)`. La razón de esto es generar un número aleatorio de tiempo de espera para darle oportunidad a los threads que se interrumpan
3. Dentro del método Main.
    a. Crear tres objetos del tipo Thread y pasar como parámetro la referencia al método EjecutarThread de una instancia de la clase ImprimirMiNombre en cada constructor de manera de inicializar adecuadamente el delegado.
    b. Asignar un nombre único a cada subproceso usando la propiedad Name.
    c. Iniciar cada subproceso.

Compilación y ejecución

4. Verificar que ninguna de las clases tenga ningún error de compilación
5. Ejecutar el programa y verificar que la salida no es secuencial cuando se imprime el nombre de cada thread, ni se puede asegurar tampoco en sucesivas ejecuciones que los threads terminen antes que las instrucciones del método Main.

Ejercicio 2

En base al ejercicio anterior, genere un proyecto cuyo nombre sea “ejercicio2” de manera que se asegure la ejecución de todos los subprocesos antes que finalicen las instrucciones del método Main. Una corroboración simple es que la última instrucción que ejecute el programa sea `Console.ReadKey()`

Ejercicio 3

El ejercicio simula una cola de impresión. Para ello posee una clase que maneja una cola circular que está compuesta como un miembro de la clase Impresora. Para mantener un mejor diseño de clases, la funcionalidad principal de la cola se define en una interfaz para que sea implementada en una clase concreta.

Para que la simulación sea efectiva, la clase Impresora se ejecuta en un subproceso propio de manera que cuando se producen trabajos de impresión externos a su instancia y se agregan a la cola de impresión, trabaje de manera independiente.

Existe una clase denominada Productor que se encarga de producir trabajos de impresión. Los usuarios crean instancias de esta clase cada vez que “generan” un trabajo para la impresora. Por lo tanto, cada usuario debe indicar a una instancia independiente de Productor cuántos trabajos enviará a la cola. Esto implica que cada usuario deberá generar

un subproceso propio (a fines de la simulación) con todos los trabajos que desee mandar a la cola de impresión.

También se generan demoras en el procesamiento de los trabajos de impresión a fin de simular el tiempo que supuestamente tardaría en imprimir un trabajo.

La impresora se convierte en un "consumidor" de trabajos en sí misma y cada vez que lo hace informa por pantalla el resultado de su trabajo anteponiendo una letra "C" (consumiendo). Análogamente, cada vez que se producen trabajos por parte del Productor se informa por pantalla con una letra "P".

Tareas

Buscar en el código del proyecto ejercicio3 los comentarios indicados con "Paso X", donde X es el número de paso a completar, que anteceden a las modificaciones necesarias para que el programa funcione según los objetivos expuestos anteriormente.

Completar los pasos indicados en el código para las siguientes clases:

1. Program
2. Impresora

Compilación y ejecución

3. Verificar que ninguna de las clases tenga ningún error de compilación
4. Ejecutar el programa y verificar que la salida esperando a que todos los trabajos en cola se completen. El programa debe finalizar cuando todos los threads se hayan ejecutado adecuadamente, caso contrario no se cumplen los objetivos propuestos.