###### Licenciatura en Ingeniería de Software

###### Programación Visual III

|  |  |
| --- | --- |
| UNIDAD III | Tarea # 7 |
| Nombre: Sergio Herrera Rivera | Fecha: 20/01/2019 |

**Garbage Collector**

Garbage Collector funciona en el heap, cuando el proceso de recolección de basura se pone en movimiento, comprueba los objetos muertos y los objetos que ya no se utilizan, luego compacta el espacio del objeto vivo e intenta liberar más memoria.

El heap está gestionado por diferentes 'Generaciones', almacena y maneja objetos de larga duración y de corta duración.

* Generación 0: esta generación contiene objetos de corta duración, por ejemplo, objetos temporales. Garbage Collector inicia el proceso de recolección de basura con frecuencia en esta generación.
* 1 Generación: esta generación es el búfer entre objetos de corta duración y de larga duración.
* 2 generaciones: esta generación contiene objetos de larga duración, como una variable estática y global, que deben persistir durante un cierto tiempo.

Cuando un objeto es creado en C#, el CLR (Common Language Runtime) asigna memoria para el objeto desde el heap, este proceso se repite para cada objeto recién creado, pero hay una limitación para todo ya que la memoria es limitada y necesitamos limpiar un poco de espacio usado para dejar espacio para nuevos objetos. Aquí se presenta el concepto de Garbage Collector. Garbage Collector gestiona la asignación de memoria. Garbage Collector realiza un viaje al heap y recopila todos los objetos que la aplicación ya no utiliza y los libera de la memoria. Cuando se desencadena un proceso, se asigna un espacio virtual separado a ese proceso, desde una memoria física que es la misma y utilizada por cada proceso de un sistema, cualquier programa trata con un espacio virtual que no tiene memoria física, Garbage Collector también se ocupa de los mismos procesos virtuales. Básicamente, hay bloques libres que existen en la memoria virtual, cuando cualquier solicitud de objetos para el administrador de asignación de memoria busca bloques libres y asigna memoria a dicho objeto.

La memoria virtual tiene tres bloques: Gratis (espacio vacío), Reservado (ya asignado) y el Comprometido (este bloque es para la memoria física y no está disponible para la asignación de espacio)

No hay tiempos específicos para que el Garbage Collector se active, el Garbage Collector inicia automáticamente, cuando la memoria virtual se está quedando sin espacio o cuando se suprime el umbral aceptable de la memoria asignada (cuando se encuentra Garbage Collector si la tasa de supervivencia (objetos vivos) es alta, entonces aumenta la asignación del umbral) y cuando llamamos al método Garbage Collector.Collect() explícitamente, como Garbage Collector se ejecuta continuamente, en realidad no se necesita llamar a este método.

Cuando creamos objetos no administrados, Garbage Collector no puede borrarlos y debemos liberarlos explícitamente cuando terminamos de usarlos. La mayoría de los objetos no administrados se envuelven / ocultan en torno a los recursos del sistema operativo, como las secuencias de archivos, las conexiones de base de datos, las instancias relacionadas con la red, los identificadores de diferentes clases, registros, punteros, etc. No es consciente de liberar recursos no administrados.

**Conclusión:**

EL Garbage Collector es de gran utilidad ya que este tiene la función de liberar memoria que ya no se está utilizando, teniendo una jerarquía de las cosas que con más importantes conservar y cuáles no, tomando en cuenta el tiempo que serán utilizadas.

El Garbage Collector se ejecuta continuamente, pero existe la función Collect() que sirve para llamarlo en el momento que nosotros deseemos.