PROGRAMACIÓN ORIENTADA A OBJETOS

Administración de la Memoria

Titular: Dario Guillermo Cardacci

Garbage Collection
GC (Garbage Collector)

- Es el encargado de liberar memoria cuando se ha agotado.
- En general libera memoria colocando como memoria disponible a aquella que posee objetos que ya no se utilizan pues no tienen referencias que los apunten.
- Funciona automáticamente aunque se lo puede invocar manualmente.

Garbage Collection

- REFERENCIAS CIRCULARES:
  - Era uno de los problemas de COM.
  - En general las referencias circulares generaban consumo innecesario de memoria.
  - En .NET no existen los contadores de referencia (AddRef y Release) en lugar de ello el objeto se ubica en un bloque de memoria administrado (managed heap).

Garbage Collection

- GENERACIONES:
  - Indica la “edad” de un objeto.
  - Indica el número de recolecciones de objetos no utilizados a la que el objeto ha sobrevivido.
  - Este valor oscila entre 0 y 2.

Garbage Collection

Ejemplo

```
Public Class Form1
   'Definición  de un Objeto de tipo String
   Dim G As String = "Prueba sobre subsistencia de generaciones“

   Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
      ' Primera consulta. G es un objeto de generación 0.
      TextBox1.Text = TextBox1.Text & Leyenda() & GC.GetGeneration(G)
      Call SaltoDeLinea()
   End Sub

   Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
      ' Segunda consulta. G es un objeto de generación 1. Observar que se ejecutó el GC.
      GC.Collect() : GC.WaitForPendingFinalizers()
      TextBox1.Text = TextBox1.Text & Leyenda() & GC.GetGeneration(G)
      Call SaltoDeLinea()
   End Sub

   Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
      ' Tercera consulta. G es un objeto de generación 2. Observar que se ejecutó el GC.
      GC.Collect() : GC.WaitForPendingFinalizers()
      TextBox1.Text = TextBox1.Text & Leyenda() & GC.GetGeneration(G)
      Call SaltoDeLinea()
   End Sub
```

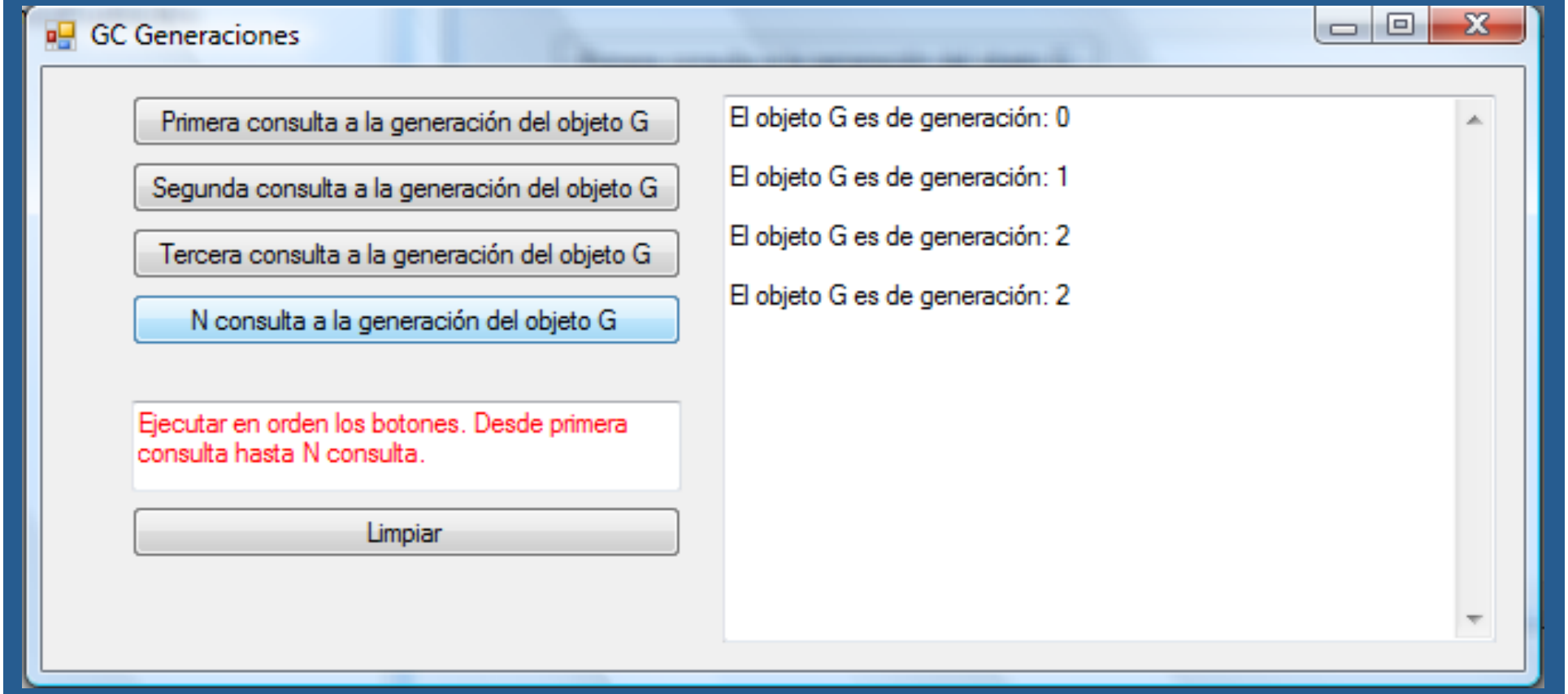
Garbage Collection

Ejemplo

```
Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
    ' N consulta. G es un objeto de generación 2. Observar que se ejecutó el GC.
    ' Observar que la máxima generación es 2.
    GC.Collect() : GC.WaitForPendingFinalizers()
    TextBox1.Text = TextBox1.Text & Leyenda() & GC.GetGeneration(G)
    Call SaltoDeLinea()
End Sub

Private Sub SaltoDeLinea()
    TextBox1.Text = TextBox1.Text & vbCrLf & vbCrLf
End Sub

Private Function Leyenda() As String
    Return "El objeto G es de generación: "
End Function

End Class
```

Garbage Collection

Ejemplo



Garbage Collection

MIEMBROS – MÉTODOS 1

Nombre	Descripción
AddMemoryPressure	Informa al motor en tiempo de ejecución de una asignación grande de memoria no administrada que se debe tener en cuenta al programar la recolección de elementos no utilizados.
Collect	Sobrecargado. Obliga a que se lleve a cabo la recolección de elementos no utilizados.
CollectionCount	Devuelve el número de veces que se ha producido la recolección de elementos no utilizados para la generación de objetos especificada.
GetGeneration	Sobrecargado. Devuelve el número de generación actual de un objeto.
GetTotalMemory	Recupera el número de bytes que se considera que están asignados en la actualidad. Un parámetro indica si este método puede esperar un breve intervalo de tiempo antes de regresar, para permitir que el sistema recoja los elementos no utilizados y finalice los objetos.
KeepAlive	Hace referencia al objeto especificado, convirtiéndolo en un objeto no válido para la recolección de elementos no utilizados desde el principio de la rutina actual hasta el momento en que se llamó a este método.

Garbage Collection

MIEMBROS – MÉTODOS 2

Nombre	Descripción
RemoveMemoryPressure	Informa al motor en tiempo de ejecución de que se ha liberado la memoria no administrada y ya no se necesita tener en cuenta al programar la recolección de elementos no utilizados.
ReRegisterForFinalize	Solicita que el sistema llame al finalizador del objeto especificado, para el que previamente se ha llamado a SuppressFinalize.
SuppressFinalize	Solicita que el sistema no llame al finalizador del objeto especificado.
WaitForPendingFinalizers	Suspende el subproceso actual hasta que el subproceso que está procesando la cola de finalizadores vacíe dicha cola.

Garbage Collection

MIEMBROS – MÉTODOS 1

Nombre	Descripción
MaxGeneration	Obtiene el número máximo de generaciones que el sistema admite en la actualidad.

Método Finalize

- Es llamado por el recolector de basura antes de liberar la memoria asignada a un objeto.
- Se hereda este método de System.Object.
- Se deberá declarar como “Protected” y “Overrides”.
- Se dispara cuando todos los punteros al objetos se han perdido. Esto pudo haber ocurrido por asignarles Nothing a las variables que los apuntaban o por que las variables quedaron fuera de scope.

Método Dispose

- Como los objetos no poseen destructor las clases bien diseñadas deberían exponer el 'Dispose'.
- Se debe implementar por medio de la interfaz IDisposable.
- Esta interfaz expone un único método denominado "Dispose".
- Una buena práctica de programación indica que el método "Dispose" de un objeto debe llamar a los métodos "Dispose" de los objetos internos que estén contenidos en él.
- El "Dispose se utiliza cuando usamos objetos que son "NO Administrados" (p.e apertura de un archivo), ya que la real liberación de la memoria se produce cuando actúa el GC

Finalize y Dispose

- Se puede utilizar el “Finalize y el “Dispose” de manera conjunta.
- El “Dispose” en general para liberar los recursos no administrados.
- El “Finalize” para los recursos administrados por el GC.


FIN