

1. Cadenas de caracteres.

1.1 Introducción.

En esta versión nos encontramos con el tipo Char que antes no había, y con el tipo String, ya conocido. Un Char un carácter, y un String muchos.

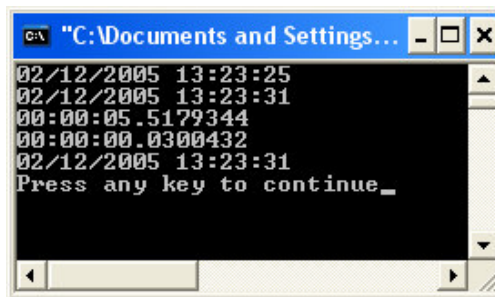
También hay que tener presente que Don Bill Gates, ha dejado de usar la tabla Ascii, y se ha pasado a su UniCode. O dicho de otra forma 2 bytes por carácter en lugar de uno.

El manejo de las cadenas no puede cambiar por mucho que se quiera.

Lo que si sucede es que la forma de utilizar los métodos para hacer una misma cosa, puede hacer que el tiempo se multiplique en la ejecución.

En los datos de la ventana podemos ver los tiempos de ejecución para el ejemplo, como podemos observar la cadena generada en el caso inferior es casi siete veces superior en tamaño, y el tiempo que ha utilizado es de 3 centésimas de segundo, contra 5 segundos y medio del primero, o sea 180 veces más rápido.

Claro que a nivel de uno o dos caracteres, da igual usar uno u otro sistema, pero hay que dejar constancia de la existencia de la clase Text con el método StringBuilder.



```
Module Ejemplo
Sub main()
    Dim Cadenita As New Text.StringBuilder
    Dim Cadena As String = "Abecedario  "
    Dim HoraActual As DateTime
    Dim X As Int32

    HoraActual = Now
    Console.WriteLine(HoraActual)
    For X = 1 To 30000
        Cadena = Cadena & "A"
    Next

    Console.WriteLine(Now)
    Console.WriteLine(Now.Subtract(HoraActual))
    HoraActual = Now

    For X = 1 To 200000
        Cadenita = Cadenita.Append("A")
    Next
    Console.WriteLine(Now.Subtract(HoraActual))
    Console.WriteLine(Now)
End Sub
End Module
```

El siguiente ejemplo es la agrupación de todos los métodos usados para unas cuantas operaciones con cadenas.

```
Module Ejemplo
Sub main()
    Dim Cadena As String = "Abecedario  "

    Console.WriteLine("La cadena [{0}] ", Cadena)
    Console.WriteLine("Su longitud {0} ", [Strings].Len(Cadena))
    Console.WriteLine("En mayúsculas {0} ", [Strings].UCase(Cadena))
    Console.WriteLine("En minúsculas {0} ", [Strings].LCase(Cadena))
    Console.WriteLine("Sin blancos [{0}] ", [Strings].Trim(Cadena))
End Sub
End Module
```

```

Console.WriteLine("Código Ascii {0} ", [Strings].Asc(Cadena))
Console.WriteLine("Carácter de un código {0} ", [Strings].Chr(65))

Console.WriteLine("Fecha y hora {0} ", [Strings].FormatDateTime(Now))

Console.WriteLine("2ª 3 caracteres {0} ", [Strings].Mid(Cadena, 2, 3))
Console.WriteLine("5 izquierda [{0}] ", [Strings].Left(Cadena, 5))
Console.WriteLine("5 derecha  [{0}] ", [Strings].Right(Cadena, 5))

Console.WriteLine("Crea 25 blancos [{0}] ", [Strings].Space(25))
Console.WriteLine("Invierte [{0}] ", [Strings].StrReverse(Cadena))
Console.WriteLine("Crea cadena 3 B [{0}] ", [Strings].StrDup(3, "B"))

Mid(Cadena, 3, 4) = "Hola"
Console.WriteLine("La cadena [{0}] ", Cadena)

Cadena = [Strings].Replace(Cadena, "Hola", "eced")
Console.WriteLine("Buscar y reemplazar {0}", Cadena)
Console.WriteLine("Longitud  [{0}] ", [Strings].Len(Cadena))

Cadena = [Strings].LSet(Cadena, 20)
Console.WriteLine("Alineación  {0}] ", Cadena)
Console.WriteLine("Longitud  [{0}] ", [Strings].Len(Cadena))

Cadena = [Strings].Trim(Cadena) ' para quitar los blancos de antes
Cadena = [Strings].RSet(Cadena, 20)
Console.WriteLine("Alineación  {0}] ", Cadena)
Console.WriteLine("Longitud  [{0}] ", [Strings].Len(Cadena))
Console.WriteLine(Asc(Cadena))

End Sub
End Module

```

1.2 Utilización.

Hay que tener presente que cada variable es en realidad un objeto, y como tal dispone de métodos, que existirán en función del tipo de variable declarada.

Además siguen existiendo las funciones de manejo de caracteres que ya conocemos, con lo que una misma tarea puede hacerse de varias formas.

1.2.1 Alinear y rellenar, Lset y Rset

Estos métodos rellenan con blancos la variable hasta la longitud indicada, por la derecha o por la izquierda.

```

Cadena = [Strings].LSet(Cadena, 20)
Console.WriteLine("Alineación  [{0}] ", Cadena)
Console.WriteLine("Longitud  [{0}] ", [Strings].Len(Cadena))

Cadena = [Strings].Trim(Cadena) ' para quitar los blancos de antes

Cadena = [Strings].RSet(Cadena, 20)
Console.WriteLine("Alineación  {0}] ", Cadena)
Console.WriteLine("Longitud  [{0}] ", [Strings].Len(Cadena))

Console.WriteLine(Asc(Cadena))

```

1.2.2 Extraer, Mid, Left, Right

El funcionamiento de estas funciones es el ya conocido, y no ha sufrido cambios.

```
Console.WriteLine("2ª 3 caracteres {0} ", [Strings].Mid(Cadena, 2, 3))
Cadena = [Strings].Right(Cadena, 2)
Cadena = [Strings].Left(Cadena, 2)
```

1.2.3 Convertir a cadena, Format, Str, ToString.

La conversión de número a cadena se puede hacer con el método Format, incluyendo como argumentos la variable número a convertir a cadena y el número de decimales después de la coma, 2 en el ejemplo. El ejemplo da como salida 1.234,00

```
Dim X As Int16 = 1234
Cadena = [Strings].FormatNumber(X, 2)
Console.WriteLine(Cadena) ` 1.234,00
```

El uso de la función clásica Str, realiza la conversión generando un blanco por la izquierda para el signo, cosa que no hace Format.

```
Cadena = Str(X)
Console.WriteLine(Cadena) ` " 1234"
```

Con el método ToString se obtiene el mismo resultado.

```
Cadena = X.ToString
Console.WriteLine(Cadena)
```

Esta asignación también la admite VB

```
X = Cadena
Console.WriteLine(Cadena)
```

1.2.4 Convertir a número, Val

La conversión de cadena a número con la función Val, sigue siendo válida.

```
Cadena = "1234"
X = Val(Cadena)
Console.WriteLine(X) ` 1234
```

Aunque esta asignación funciona perfectamente, sorprendentemente no, porque en VB6 ya pasaba.

```
X = Cadena
Console.WriteLine(X)
```

1.2.5 Sustituir

El uso como instrucción de Mid, de la forma habitual, variable, primera posición de sustitución y cuantos a sustituir.

```
Mid(Cadena, 3, 4) = "Hola"
```

1.2.6 Buscar y reemplazar, Replace

```
Cadena = [Strings].Replace(Cadena, "Hola", "eced")
Console.WriteLine("Buscar y reemplazar {0}", Cadena)
```

1.2.7 Buscar posición de cadena, InStr, InStrRev

```
Console.WriteLine("Existe cadena {0} ", [Strings].InStr(Cadena, "A"))
```

1.2.8 Generar cadenas, StrDup

Desaparece la función Strings, que es sustituida por StrDup.

```
Console.WriteLine("Crea cadena 3 B [{0}] ", [Strings].StrDup(3, "B"))
```

1.2.9 Obtener código de carácter, Asc

```
Console.WriteLine("Código Ascii {0} ", [Strings].Asc(Cadena))
```

1.2.10 Crear un carácter, Chr

```
Console.WriteLine("Carácter de un código {0} ", [Strings].Chr(65))
```

1.2.11 Longitud de la cadena, Len

```
Console.WriteLine("Longitud de cadena {0}", [Strings].Len(Cadena))
```

1.2.12 Eliminar blancos de una variable, Trim, Ltrim, Rtrim.

En el ejemplo que sigue aparece el uso de las funciones de eliminar blancos, Ltrim, Rtrim y Trim.

También se ha utilizado dos sintaxis distintas de las funciones, con y sin Strings delante de la función.

Hay que tener presente que hay alguna función que coincide en nombre con otras propiedades y en ese caso, Left por ejemplo, es necesario hacer mención al espacio de nombres Strings, de forma obligatoria.

```
Module Ejemplo

Sub Main()
    Dim Cadena = "  Hola mundo  "
    Console.WriteLine("Longitud de cadena {0}", Strings.Len(Cadena))
    Cadena = Trim(Cadena)
    Console.WriteLine("Longitud de {0}, cadena [{1}]", Len(Cadena), Cadena)
    Cadena = "  Hola mundo  "
    Cadena = LTrim(Cadena)
    Console.WriteLine("Longitud de {0}, cadena [{1}]", Len(Cadena), Cadena)
    Cadena = "  Hola mundo  "
    Cadena = Strings.RTrim(Cadena)
    Console.WriteLine("Longitud de {0}, cadena [{1}]", Len(Cadena), Cadena)
    Console.ReadLine()
End Sub

End Module
```