## Funciones de Visual Basic

### Contenido

<table>
<thead>
<tr>
<th>Funciones matemáticas</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Funciones matemáticas derivadas</td>
<td>2</td>
</tr>
<tr>
<td>Abs (Función)</td>
<td>4</td>
</tr>
<tr>
<td>Atn (Función)</td>
<td>4</td>
</tr>
<tr>
<td>Asc (Función)</td>
<td>5</td>
</tr>
<tr>
<td>Chr (Función)</td>
<td>5</td>
</tr>
<tr>
<td>Cos (Función)</td>
<td>6</td>
</tr>
<tr>
<td>Date (Función)</td>
<td>6</td>
</tr>
<tr>
<td>DateAdd (Función)</td>
<td>6</td>
</tr>
<tr>
<td>DateSerial (Función)</td>
<td>8</td>
</tr>
<tr>
<td>DateValue (Función)</td>
<td>9</td>
</tr>
<tr>
<td>Day (Función)</td>
<td>9</td>
</tr>
<tr>
<td>Exp (Función)</td>
<td>9</td>
</tr>
<tr>
<td>Hour (Función)</td>
<td>10</td>
</tr>
<tr>
<td>InputBox (Función)</td>
<td>10</td>
</tr>
<tr>
<td>InStr (Función)</td>
<td>12</td>
</tr>
<tr>
<td>InStrRev (Función)</td>
<td>13</td>
</tr>
<tr>
<td>Int, Fix (Funciones)</td>
<td>14</td>
</tr>
<tr>
<td>IsDate (Función)</td>
<td>15</td>
</tr>
<tr>
<td>IsEmpty (Función)</td>
<td>15</td>
</tr>
<tr>
<td>IsNumeric (Función)</td>
<td>16</td>
</tr>
<tr>
<td>LCase (Función)</td>
<td>17</td>
</tr>
<tr>
<td>Left (Función)</td>
<td>17</td>
</tr>
<tr>
<td>Len (Función)</td>
<td>18</td>
</tr>
<tr>
<td>LoadPicture (Función)</td>
<td>19</td>
</tr>
<tr>
<td>Log (Función)</td>
<td>21</td>
</tr>
<tr>
<td>LTrim, Rtrim y Trim (Funciones)</td>
<td>22</td>
</tr>
<tr>
<td>Mid (Función)</td>
<td>23</td>
</tr>
<tr>
<td>Minute (Función)</td>
<td>24</td>
</tr>
<tr>
<td>Month (Función)</td>
<td>24</td>
</tr>
<tr>
<td>MonthName (Función)</td>
<td>25</td>
</tr>
<tr>
<td>MsgBox (Función)</td>
<td>25</td>
</tr>
<tr>
<td>Now (Función)</td>
<td>28</td>
</tr>
<tr>
<td>QBColor (Función)</td>
<td>28</td>
</tr>
<tr>
<td>Replace (Función)</td>
<td>29</td>
</tr>
<tr>
<td>RGB (Función)</td>
<td>30</td>
</tr>
<tr>
<td>Right (Función)</td>
<td>31</td>
</tr>
<tr>
<td>Rnd (Función)</td>
<td>31</td>
</tr>
<tr>
<td>Randomize (Instrucción)</td>
<td>32</td>
</tr>
<tr>
<td>Round (Función)</td>
<td>33</td>
</tr>
<tr>
<td>Second (Función)</td>
<td>33</td>
</tr>
<tr>
<td>Sgn (Función)</td>
<td>33</td>
</tr>
</tbody>
</table>
Funciones matemáticas

- Abs (Función)
- Atn (Función)
- Cos (Función)
- Exp (Función)
- Fix (Función)
- Int (Función)
- Log (Función)
- Rnd (Función)
- Sgn (Función)
- Sin (Función)
- Sqr (Función)
- Tan (Función)

Funciones matemáticas derivadas

La siguiente es una lista de funciones matemáticas no intrínsecas que pueden derivarse de funciones matemáticas intrínsecas:

<table>
<thead>
<tr>
<th>Función</th>
<th>Derivadas equivalentes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secante</td>
<td>Sec(X) = 1 / Cos(X)</td>
</tr>
<tr>
<td>Cosecante</td>
<td>Cosec(X) = 1 / Sin(X)</td>
</tr>
<tr>
<td>Cotangente</td>
<td>Cotan(X) = 1 / Tan(X)</td>
</tr>
<tr>
<td>Función</td>
<td>Fórmula</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>Seno inverso</td>
<td>Arccos(X) = Atn(-X / Sqr(-X * X + 1))</td>
</tr>
<tr>
<td>Coseno inverso</td>
<td>Arccos(X) = Atn(-X / Sqr(-X * X + 1)) + 2 * Atn(1)</td>
</tr>
<tr>
<td>Secante inversa</td>
<td>Arcsec(X) = Atn(X / Sqr(X * X - 1)) + Sgn((X) - 1) * (2 * Atn(1))</td>
</tr>
<tr>
<td>Cosecante inversa</td>
<td>Arccosec(X) = Atn(X / Sqr(X * X - 1)) + (Sgn(X) - 1) * (2 * Atn(1))</td>
</tr>
<tr>
<td>Cotangente inversa</td>
<td>Arccotan(X) = Atn(X) + 2 * Atn(1)</td>
</tr>
<tr>
<td>Seno hiperbólico</td>
<td>HSin(X) = (Exp(X) - Exp(-X)) / 2</td>
</tr>
<tr>
<td>Coseno hiperbólico</td>
<td>HCos(X) = (Exp(X) + Exp(-X)) / 2</td>
</tr>
<tr>
<td>Tangente hiperbólica</td>
<td>HTan(X) = (Exp(X) - Exp(-X)) / (Exp(X) + Exp(-X))</td>
</tr>
<tr>
<td>Secante hiperbólica</td>
<td>HSec(X) = 2 / (Exp(X) + Exp(-X))</td>
</tr>
<tr>
<td>Cosecante hiperbólica</td>
<td>HCosec(X) = 2 / (Exp(X) - Exp(-X))</td>
</tr>
<tr>
<td>Cotangente hiperbólica</td>
<td>HCotan(X) = (Exp(X) + Exp(-X)) / (Exp(X) - Exp(-X))</td>
</tr>
<tr>
<td>Seno hiperbólico inverso</td>
<td>HArcsin(X) = Log(X + Sqr(X * X + 1))</td>
</tr>
<tr>
<td>Coseno hiperbólico inverso</td>
<td>HArcos(X) = Log(X + Sqr(X * X - 1))</td>
</tr>
<tr>
<td>Tangente hiperbólica inversa</td>
<td>HArcctan(X) = Log((1 + X) / (1 - X)) / 2</td>
</tr>
<tr>
<td>Secante hiperbólica inversa</td>
<td>HArcsec(X) = Log((Sqr(-X * X + 1) + 1) / X)</td>
</tr>
<tr>
<td>Cosecante inversa hiperbólica</td>
<td>HArc cosec(X) = Log((Sgn(X) * Sqr(X * X + 1) + 1) / X)</td>
</tr>
<tr>
<td>Cotangente inversa hiperbólica</td>
<td>HArc cotan(X) = Log((X + 1) / (X - 1)) / 2</td>
</tr>
<tr>
<td>Logaritmo en base</td>
<td>N LogN(X) = Log(X) / Log(N)</td>
</tr>
</tbody>
</table>
**Abs (Función)**

Devuelve un valor del mismo tipo que el que se pasó como parámetro y que especifica el valor absoluto de un número.

*Sintaxis*

```
Abs(número)
```

El argumento número puede ser cualquier expresión numérica válida. Si número contiene Null, la función devolverá Null; si es una variable no inicializada, devolverá cero.

**Comentarios**

El valor absoluto de un número es su magnitud sin signo. Por ejemplo, ABS(-1) y ABS(1) devuelven 1.

**Ejemplos**

Dim MiNúmero

MiNúmero = Asc("A")  ' Devuelve 65.
MiNúmero = Asc("a")  ' Devuelve 97.
MiNúmero = Asc("Apple")  ' Devuelve 65.

---

**Atn (Función)**

Devuelve un tipo Double que especifica el arcotangente de un número.

*Sintaxis*

```
Atn(número)
```

El argumento número es un tipo Double o cualquier expresión numérica válida.

**Comentarios**

La función Atn toma la razón de dos lados de un triángulo rectángulo (número) y devuelve el ángulo correspondiente en radianes. La razón es la longitud del lado opuesto al ángulo dividida por la longitud del lado adyacente al ángulo.

El ángulo del resultado es \(-\pi/2\) a \(\pi/2\) radianes.

Para convertir grados en radianes, multiplique los grados por \(\pi/180\). Para convertir radianes en grados, multiplique los radianes por \(180/\pi\).

*Nota* Atn es la función trigonométrica inversa de Tan, que toma un ángulo y su argumento y devuelve la razón de dos lados de un triángulo rectángulo. No confunda con la cotangente, que es el inverso de la tangente (1/tangente).

**Ejemplo**

Dim pi

pi = 4 * Atn(1)  ' Calcula el valor de pi.
**Asc (Función)**

Devuelve un tipo Integer que representa el código de carácter correspondiente a la primera letra de una cadena.

**Sintaxis**

\[
\text{Asc(cadena)}
\]

El argumento obligatorio cadena es una expresión de cadena válida. Si cadena no contiene caracteres, se produce un error en tiempo de ejecución.

**Comentarios**

El intervalo del valor de retorno es entre 0 y 255 en sistemas que no sean DBCS, y entre –32768 y 32767 en sistemas DBCS.

Nota La función AscB se utiliza con bytes de datos contenidos en una cadena. En lugar de devolver el código de carácter para el primer carácter, AscB devuelve el primer byte. La función AscW siempre devuelve el código de carácter Unicode en plataformas donde no se admite Unicode, en cuyo caso, el comportamiento es idéntico al de la función Asc.

**Ejemplos**

En este ejemplo se utiliza la función Asc para devolver el código de carácter correspondiente a la primera letra de una cadena de caracteres.

Dim MiNúmero

MiNúmero = Asc("A")  ' Devuelve 65.
MiNúmero = Asc("a")  ' Devuelve 97.
MiNúmero = Asc("Apple")  ' Devuelve 65.

---

**Chr (Función)**

Devuelve un tipo String que contiene el carácter asociado con el código de carácter especificado.

**Sintaxis**

\[
\text{Chr(códigocar)}
\]

El argumento códigocar es un tipo Long que identifica a un carácter.

**Comentarios**

Los números del 0 al 31 son los mismos que los códigos ASCII estándar no imprimibles. Por ejemplo, Chr(10) devuelve un carácter de avance de línea. El intervalo normal de códigocar es 0–255. Sin embargo, en sistemas DBCS, el intervalo real de códigocar es de -32768 a 65535.

Nota La función ChrB se utiliza con datos de byte incluidos en un tipo String. En lugar de devolver un carácter, que puede ser de uno o de dos bytes, ChrB siempre devuelve un único byte. La función ChrW devuelve un tipo String que contiene el carácter Unicode excepto en plataformas donde no se admite Unicode, en cuyo caso, el comportamiento es idéntico al de la función Chr.

**Ejemplos**

Dim MiCar
MiCar = Chr(65)  ' Devuelve A.
MiCar = Chr(97)  ' Devuelve a.
MiCar = Chr(62)  ' Devuelve >.
MiCar = Chr(37)  ' Devuelve %.

**Cos (Función)**
Devuelve un tipo Double que especifica el coseno de un ángulo.

**Sintaxis**

\[
\text{Cos(\text{número})}
\]

El argumento número es un tipo Double o cualquier expresión numérica válida que expresa un ángulo en radianes.

**Comentarios**
La función Cos toma un ángulo y devuelve la razón de dos lados de un triángulo rectángulo. La razón es la longitud del lado adyacente al ángulo dividida por la longitud de la hipotenusa.

El resultado está dentro del intervalo -1 a 1.

Para convertir grados en radianes, multiplique los grados por \(\pi/180\). Para convertir radianes en grados, multiplique los radianes por 180/\(\pi\).

**Ejemplo**
Dim MiÁngulo, MiSecante
MiÁngulo = 1.3  ' Define el ángulo en radianes.
MiSecante = 1 / Cos(MiÁngulo)  ' Calcula la secante.

**Date (Función)**
Devuelve un tipo Variant (Date) que contiene la fecha actual del sistema.

**Sintaxis**

\[
\text{Date}
\]

**Comentarios**
Utilice la instrucción Date para establecer la fecha del sistema.

**Ejemplo**
Dim MiFecha
MiFecha = Date  ' MiFecha contiene la fecha del sistema actual.

**DateAdd (Función)**
Devuelve un valor de tipo Variant (Date) con una fecha a la que se ha agregado un intervalo de tiempo especificado.
Sintaxis

**DateAdd(intervalo, número, fecha)**

La sintaxis de la función DateAdd consta de los siguientes argumentos con nombre:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>intervalo</td>
<td>Requerido. Expresión de tipo cadena (String) que contiene el intervalo de tiempo que se desea agregar.</td>
</tr>
<tr>
<td>número</td>
<td>Requerido. Expresión numérica con el número de intervalos que se desea agregar. Puede ser positiva (para obtener fechas futuras) o negativa (para obtener fechas pasadas).</td>
</tr>
<tr>
<td>fecha</td>
<td>Requerido. Valor de tipo Variant (Date) o literal que representa la fecha en la que se agregó el intervalo.</td>
</tr>
</tbody>
</table>

**Valores**

El argumento intervalo tiene los siguientes valores:

<table>
<thead>
<tr>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>yyyy</td>
<td>Año</td>
</tr>
<tr>
<td>q</td>
<td>Trimestre</td>
</tr>
<tr>
<td>m</td>
<td>Mes</td>
</tr>
<tr>
<td>y</td>
<td>Día del año</td>
</tr>
<tr>
<td>d</td>
<td>Día</td>
</tr>
<tr>
<td>w</td>
<td>Día de la semana</td>
</tr>
<tr>
<td>ww</td>
<td>Semana</td>
</tr>
<tr>
<td>h</td>
<td>Hora</td>
</tr>
<tr>
<td>n</td>
<td>Minuto</td>
</tr>
<tr>
<td>s</td>
<td>Segundo</td>
</tr>
</tbody>
</table>

**Comentarios**

Puede utilizar la función DateAdd para agregar o restar un intervalo de tiempo especificado de una fecha. Por ejemplo, con DateAdd podría calcular una fecha 30 días posterior al día de hoy o una hora 45 minutos posterior a la actual.

Si desea agregar días a fecha, puede utilizar Día del año ("y"). Día ("d") o Día de la semana ("w").

La función DateAdd no devuelve nunca una fecha no válida. En el ejemplo siguiente se agrega un mes al 31 de enero:

```vba
DateAdd("m", 1, "31-ene-95")
```

En este caso, DateAdd devuelve 28-feb-95 y no 31-feb-95. Si fecha es 31-ene-96, devolverá 29-feb-96, ya que 1996 es un año bisiesto.

Si la fecha calculada va a resultar inferior al año 100 (esto es, se restan más años que los que hay en fecha), se produce un error.

Si número no es un valor de tipo Long se redondeará al número entero más cercano antes de ser evaluado.

Nota   El formato del valor devuelto por DateAdd es determinado por la configuración del Panel de
control, no por el formato que se pasa en el argumento fecha.

**DateSerial (Función)**

Devuelve un tipo Variant (Date) para un año, mes y día especificados.

Sintaxis

\[
\text{DateSerial(año, mes, día)}
\]

La sintaxis de la función DateSerial consta de los siguientes argumentos con nombre:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>año</td>
<td>Se requiere; Integer. Número entre 100 y 9999, inclusive o una expresión numérica.</td>
</tr>
<tr>
<td>mes</td>
<td>Se requiere; Integer. Cualquier expresión numérica.</td>
</tr>
<tr>
<td>día</td>
<td>Se requiere; Integer. Cualquier expresión numérica.</td>
</tr>
</tbody>
</table>

**Comentarios**

Para especificar una fecha, como el 31 de diciembre de 1991, el intervalo de números para cada argumento de DateSerial debería estar en el intervalo normalmente aceptado para la unidad; es decir, 1–31 para días y 1–12 para meses. Sin embargo, puede especificar fechas relativas para cada argumento, usando cualquier expresión numérica que represente algún número de días, meses o años, antes o después de cierta fecha.

El ejemplo siguiente usa expresiones numéricas en lugar de números de fecha absoluta. Aquí la función DateSerial devuelve una fecha que es el día antes del primer día del mes (1 - 1), dos meses antes de agosto (8 - 2), 10 años antes de 1990 (1990 - 10); en otras palabras, el 31 de mayo de 1980.

DateSerial(1990 - 10, 8 - 2, 1 - 1)

Para el argumento año, los valores entre 0 y 29, inclusive, se interpretan como los años 2000–2029. Los valores entre 30 y 99, se interpretan como los años 1930–1999. Para el resto de los argumentos año se usa un año con los cuatro dígitos (por ejemplo, 1800).

Cuando un argumento no esté incluido en el intervalo aceptado normalmente para él, se incrementa hasta la siguiente unidad mayor, según convenga. Por ejemplo, si especifica 35 días, se evalúa como un mes y algunos días, dependiendo de la parte del año donde se aplique. No obstante, si un solo argumento está fuera de su intervalo -32.768 a 32.767 o si la fecha especificada por los tres argumentos, directamente o mediante expresión, no se incluye en el intervalo aceptable de fechas, se produce un error.

**Ejemplo**

Dim MiFecha

' MiFecha contiene la fecha correspondiente al 12 de febrero de 1969.

MiFecha = DateSerial(1969, 2, 12) ' Devuelve una fecha.
**DateValue (Función)**

Devuelve un tipo Variant (Date).

**Sintaxis**

```vba
DateValue(fecha)
```

El argumento `fecha` es normalmente una expresión de tipo cadena (String) que representa una fecha del 1 de enero del año 100, al 31 de diciembre de 9999. Sin embargo, `fecha` también puede ser cualquier expresión que pueda representar una fecha, una hora o una fecha y una hora en ese intervalo.

**Comentarios**

Si `fecha` es una cadena que incluye solamente números separados por separadores de fecha, DateValue reconoce el orden para mes, día y año de acuerdo al formato de Fecha abreviado que se haya especificado para el sistema. DateValue también reconoce fechas claramente definidas que contengan nombres de mes, en forma completa o abreviada. Por ejemplo, además de reconocer 30/12/1991 y 30/12/91, DateValue también reconoce 30 de diciembre de 1991 y dic 31, 1991.

Si se omite la parte de año de `fecha`, DateValue usa el año actual de la fecha del sistema de su PC.

Si el argumento `fecha` incluye información de hora, DateValue no la devuelve. Sin embargo, si `fecha` incluye información de hora no válida (tal como "89:98"), ocurrirá un error.

**Ejemplo**

```vba
Dim MiFecha
MiFecha = DateValue("12 de febrero de 1969")
' Devuelve una fecha.
```

**Day (Función)**

Devuelve el valor de tipo Variant (Integer) que especifica un número entero entre 1 y 31, inclusive, que representa el día del mes.

**Sintaxis**

```vba
Day(fecha)
```

El argumento `fecha` es cualquier valor de tipo Variant, expresión numérica, cadena, o cualquier combinación que pueda representar una fecha. Si `fecha` contiene Null, la función devolverá Null.

**Ejemplo**

```vba
Dim MiFecha, MiDía
MiFecha = #12 febrero 1969#
' Asigna una fecha.
MiDía = Day(MiFecha)
' MiDía contiene 12.
```

**Exp (Función)**

Devuelve un tipo Double que especifica e (la base de los logaritmos naturales) elevado a una potencia.
Sintaxis

**Exp(número)**

El argumento número es un tipo Double o cualquier expresión numérica válida

**Comentarios**

Si el valor de número es mayor que 709.782712893, se produce un error. El valor de la constante e es 2.718282 aproximadamente.

Nota    La función Exp complementa la acción de la función Log y a veces se llama antilogaritmo.

**Ejemplo**

Dim MiÁngulo, MiSenoH

' Define el ángulo en radianes.
MiÁngulo = 1.3

' Calcula el seno hiperbólico.
MiSenoH = (Exp(MiÁngulo) - Exp(-1 * MiÁngulo)) / 2

---

**Hour (Función)**

Devuelve un valor de tipo Variant (Integer) que especifica un número entero entre 0 y 23, inclusive, que representa la hora del día.

Sintaxis

**Hour(hora)**

El argumento hora puede ser un valor de tipo Variant, una expresión numérica, expresión de tipo cadena o cualquier combinación, que pueda representar una hora. Si hora contiene Null, la función devolverá Null.

**Ejemplo**

Dim MiTiempo, MiHora

MiTiempo = #4:35:17 PM#   ' Asigna una hora.
MiHora = Hour(MiTiempo)   ' MiHora contiene 16.

---

**InputBox (Función)**

Muestra un mensaje en un cuadro de diálogo, espera que el usuario escriba un texto o haga clic en un botón y devuelve un tipo String con el contenido del cuadro de texto.

Sintaxis

**InputBox(prompt[, title][, default][, xpos][, ypos][, helpfile, context])**

La sintaxis de la función InputBox consta de estos argumentos con nombre:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>prompt</td>
<td></td>
</tr>
<tr>
<td>title</td>
<td></td>
</tr>
<tr>
<td>default</td>
<td></td>
</tr>
<tr>
<td>xpos</td>
<td></td>
</tr>
<tr>
<td>ypos</td>
<td></td>
</tr>
<tr>
<td>helpfile</td>
<td></td>
</tr>
<tr>
<td>context</td>
<td></td>
</tr>
</tbody>
</table>
prompt | Requerido. Expresión de cadena que se muestra como mensaje en el cuadro de diálogo. La longitud máxima de prompt es de aproximadamente 1024 caracteres, según el ancho de los caracteres utilizados. Si prompt consta de más de una línea, puede separarlos utilizando un carácter de retorno de carro (Chr(13)), un carácter de avance de línea (Chr(10)) o una combinación de los caracteres de retorno de carro-avance de línea (Chr(13) y Chr(10)) entre cada línea y la siguiente.

title | Opcional. Expresión de cadena que se muestra en la barra de título del cuadro de diálogo. Si omite title, en la barra de título se coloca el nombre de la aplicación.

default | Opcional. Expresión de cadena que se muestra en el cuadro de texto como respuesta predeterminada cuando no se suministra una cadena. Si omite default, se muestra el cuadro de texto vacío.

xpos | Opcional. Expresión numérica que especifica, en twips, la distancia en sentido horizontal entre el borde izquierdo del cuadro de diálogo y el borde izquierdo de la pantalla. Si se omite xpos, el cuadro de diálogo se centra horizontalmente.

ypos | Opcional. Expresión numérica que especifica, en twips, la distancia en sentido vertical entre el borde superior del cuadro de diálogo y el borde superior de la pantalla. Si se omite ypos, el cuadro de diálogo se coloca a aproximadamente un tercio de la altura de la pantalla, desde el borde superior de la misma.

helpfile | Opcional. Expresión de cadena que identifica el archivo de Ayuda que se utilizará para proporcionar ayuda interactiva para el cuadro de diálogo. Si se especifica helpfile, también deberá especificarse context.

context | Opcional. Expresión numérica que es el número de contexto de Ayuda asignado por el autor al tema de Ayuda correspondiente. Si se especifica context, también deberá especificarse helpfile.

Comentarios

Cuando se especifica tanto helpfile como context, el usuario puede presionar F1 para ver el tema de Ayuda correspondiente a context. Algunas aplicaciones host, por ejemplo, Microsoft Excel, también agregarán automáticamente un botón Ayuda al cuadro de diálogo. Si el usuario hace clic en Aceptar o presiona Enter, la función InputBox devuelve lo que haya en el cuadro de texto. Si el usuario hace clic en Cancelar, la función devuelve una cadena de caracteres de longitud cero (""").

Nota Si desea especificar más que el primer argumento con nombre, debe utilizar InputBox en una expresión. Si desea omitir algunos argumentos de posición, debe incluir el delimitador de coma correspondiente.

Ejemplos

En este ejemplo se muestran distintas maneras de utilizar la función InputBox para indicar al usuario que debe introducir un valor. Si se omiten las posiciones x e y, el diálogo se centra automáticamente según los ejes respectivos. La variable MyValue contiene el valor introducido por el usuario, si éste hace clic sobre Aceptar o presiona Enter. Si el usuario hace clic sobre Cancelar, se devuelve una cadena de caracteres de longitud cero.

Dim Mensaje, Título, ValorPred, MiValor
Mensaje = " Introduzca un número del 1 a 3" ' Establece el mensaje.
Título = "Demostración de InputBox" ' Establece el título.
ValorPred = "1" ' Establece el valor predeterminado.
'Muestra el mensaje, el título, y el valor predeterminado.
MiValor = InputBox(Mensaje, Título, ValorPred)

' Muestra el mensaje, el título y el valor predeterminado.
MiValor = InputBox(Mensaje, Título, , , "DEMO.HLP", 10)

' Se muestra el diálogo en la posición 100, 100.
MiValor = InputBox(Mensaje, Título, ValorPred, 100, 100)

**InStr (Función)**

Devuelve un tipo Variant (Long) que especifica la posición de la primera aparición de una cadena en otra.

_Sintaxis_

\[
\text{InStr}([\text{start}, \text{string1}, \text{string2}[\text{, compare}])}
\]

La sintaxis de la función InStr tiene estos argumentos:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>start</td>
<td>Opcional. Expresión numérica que establece la posición inicial para cada búsqueda. Si se omite, la búsqueda comienza en la posición del primer carácter. Si inicio contiene un valor null, se produce un error. El argumento inicio se requiere si se especifica compare.</td>
</tr>
<tr>
<td>string1</td>
<td>Requerido. Expresión de cadena en la que se busca.</td>
</tr>
<tr>
<td>string2</td>
<td>Requerido. Expresión de cadena buscada.</td>
</tr>
<tr>
<td>compare</td>
<td>Opcional. Especifica el tipo de comparación de cadena. Si se omite compare, el valor Option Compare determina el tipo de comparación.</td>
</tr>
</tbody>
</table>

_Valores_

Los valores del argumento compare son:

<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>vbUseCompareOption</td>
<td>1</td>
<td>Realiza una comparación con los valores de la instrucción Option Compare.</td>
</tr>
<tr>
<td>vbBinaryCompare</td>
<td>0</td>
<td>Realiza una comparación binaria.</td>
</tr>
<tr>
<td>vbTextCompare</td>
<td>1</td>
<td>Realiza una comparación textual.</td>
</tr>
<tr>
<td>vbDatabaseCompare</td>
<td>2</td>
<td>Sólo para Microsoft Access. Realiza una comparación basada en la información en las bases de datos.</td>
</tr>
</tbody>
</table>

Valores devueltos

<table>
<thead>
<tr>
<th>Si</th>
<th>La función InStr devuelve</th>
</tr>
</thead>
<tbody>
<tr>
<td>string1 es de longitud cero</td>
<td>0</td>
</tr>
<tr>
<td>string1 es Null</td>
<td>Null</td>
</tr>
<tr>
<td>string2 es de longitud cero</td>
<td>start</td>
</tr>
<tr>
<td>string2 es Null</td>
<td>Null</td>
</tr>
<tr>
<td>string2 no se encontró</td>
<td>0</td>
</tr>
<tr>
<td>string2 se encontró dentro de string1</td>
<td>Posición en la que se halla la coincidencia</td>
</tr>
<tr>
<td>start &gt; string2</td>
<td>0</td>
</tr>
</tbody>
</table>

**Comentarios**

La función InStrB se utiliza con los datos de byte contenidos en una cadena. En lugar de devolver la posición del carácter de la primera aparición de una cadena en otra, InStrB devuelve la posición del byte.

**Ejemplo**

En este ejemplo se utiliza la función InStr para devolver la posición de la primera ocurrencia de una cadena en otra.

Dim SearchString, SearchChar, MyPos
SearchString = "XXpXXpXXPXXP" ' Cadena en la que se busca.
SearchChar = "P" ' Se busca "P".
' Una comparación textual que comienza en la posición 4. Devuelve 6.
MyPos = Instr(4, SearchString, SearchChar, 1)
' Una comparación binaria que comienza en la posición 1. Devuelve 9.
MyPos = Instr(1, SearchString, SearchChar, 0)
' La comparación es binaria por configuración predeterminada (si se omite el último argumento).
MyPos = Instr/SearchString, SearchChar) ' Devuelve 9.
MyPos = Instr(1, SearchString, "W") ' Devuelve 0.

---

**InStrRev (Función)**

Descripción

Devuelve la posición de una aparición de una cadena dentro de otra, desde el final de la cadena.

Sintaxis

`InstrRev(cadena1, cadena2[, inicio[, comparar]])`

La sintaxis de la función InstrRev consta de las siguientes partes:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>cadena1</td>
<td>Requerido. Expresión de cadena en la que se busca.</td>
</tr>
<tr>
<td>cadena2</td>
<td>Requerido. Expresión de cadena que se busca.</td>
</tr>
<tr>
<td>inicio</td>
<td>Opcional. Una expresión numérica que establece la posición inicial para cada búsqueda. Si se omite, se utiliza –1, que significa que la búsqueda empieza en la posición del último carácter. Si el inicio contiene Nulo, se produce un error.</td>
</tr>
<tr>
<td>comparar</td>
<td>Opcional. Un valor numérico que indica la clase de comparación que se utilizará cuando se evalúen subcadenas. Si se omite, se realiza una comparación binaria. Veá la sección Valores para consultar los valores posibles.</td>
</tr>
</tbody>
</table>

**Valores**
El argumento comparar puede tener los valores siguientes:

<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>vbUseCompareOption</td>
<td>–1</td>
<td>Realizar una comparación utilizando el valor de la instrucción Option Compare.</td>
</tr>
<tr>
<td>vbBinaryCompare</td>
<td>0</td>
<td>Realizar una comparación binaria.</td>
</tr>
<tr>
<td>vbTextCompare</td>
<td>1</td>
<td>Realiza una comparación basada en el texto.</td>
</tr>
<tr>
<td>vbDatabaseCompare</td>
<td>2</td>
<td>Sólo para Microsoft Access. Realizar una comparación basada en la información de su base de datos.</td>
</tr>
</tbody>
</table>

**Valores de retorno**

<table>
<thead>
<tr>
<th>InStrRev</th>
<th>devuelve los valores siguientes:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Si I</td>
<td>nStrRev devuelve</td>
</tr>
<tr>
<td>cadena1 es de longitud cero</td>
<td>0</td>
</tr>
<tr>
<td>cadena1 es Null</td>
<td>Null</td>
</tr>
<tr>
<td>cadena2 es de longitud cero</td>
<td>Inicio</td>
</tr>
<tr>
<td>cadena2 es Null</td>
<td>Null</td>
</tr>
<tr>
<td>cadena2 no se encuentra</td>
<td>0</td>
</tr>
<tr>
<td>cadena2 se encuentra dentro de cadena1</td>
<td>La posición en la que se encuentra la coincidencia</td>
</tr>
<tr>
<td>inicio &gt; Len(cadena2)</td>
<td>0</td>
</tr>
</tbody>
</table>

**Comentarios**

Observe que la sintaxis para la función InstrRev no es la misma que la sintaxis para la función Instr.

**Int, Fix (Funciones)**

Devuelve la parte entera de un número.

Sintaxis

```plaintext
Int(úmero)
```

```plaintext
Fix(úmero)
```

El argumento número es un tipo Double o cualquier expresión numérica válida. Si número contiene Null, la función devolverá Null.

**Comentarios**

Las funciones Int y Fix eliminan la fracción de un número y devuelven el valor entero resultante.

La diferencia entre Int y Fix es que si el número es negativo, Int devuelve el primer entero negativo menor o igual a número, mientras que Fix devuelve el primer entero negativo mayor o igual a número. Por ejemplo, Int convierte -8.4 en -9 y Fix convierte -8.4 a -8.

La función Fix(úmero) es equivalente a:

```plaintext
Sgn(úmero) * Int(Abs(úmero))
```

**Ejemplo**
Este ejemplo muestra cómo las funciones Int y Fix devuelven la parte entera de los números. En el caso de que el argumento sea un número negativo, la función Int devuelve el primer número entero negativo menor o igual que el número; la función Fix devuelve el primer entero negativo mayor o igual que el número.

Dim MiNúmero

MiNúmero = Int(99.8)   ' Devuelve 99.
MiNúmero = Fix(99.2)   ' Devuelve 99.

MiNúmero = Int(-99.8)   ' Devuelve -100.
MiNúmero = Fix(-99.8)   ' Devuelve -99.

MiNúmero = Int(-99.2)   ' Devuelve -100.
MiNúmero = Fix(-99.2)   ' Devuelve -99.

---

**IsDate (Función)**

Devuelve un valor de tipo Boolean que indica si una expresión se puede convertir en una fecha.

**Sintaxis**

    IsDate(expresión)

El argumento expresión requerido, es un tipo de datos Variant que contiene una expresión de fecha o una expresión de cadena reconocible como una fecha o una hora.

**Comentarios**

La función IsDate devuelve True si la expresión es una fecha o se puede reconocer como una fecha válida; en caso contrario, devuelve False. En Microsoft Windows, el intervalo de fechas válidas va desde el 1 de enero de 100 D. de C. hasta el 31 de diciembre de 9999 D. de C.; los intervalos varían de un sistema operativo a otro.

**Ejemplo**

Dim MiFecha, TuFecha, SinFecha, MiPrueba

MiFecha = "12 febrero 1969": TuFecha = #2/12/69#: SinFecha = "Hola"

MiPrueba = IsDate(MiFecha)      ' Devuelve True.
MiPrueba = IsDate(TuFecha)      ' Devuelve True.
MiPrueba = IsDate(SinFecha)      ' Devuelve False.

---

**IsEmpty (Función)**

Devuelve un valor de tipo Boolean que indica si una variable ha sido inicializada.

**Sintaxis**

    IsEmpty(expresión)
El argumento expresión requerido es un tipo de datos Variant que contiene una expresión de cadena o una expresión numérica. Sin embargo, ya que la función IsEmpty se utiliza para determinar si las variables individuales están inicializadas, el argumento expresión es generalmente un único nombre de variable.

Comentarios

La función IsEmpty devuelve True si la variable no está inicializada o está explícitamente configurada a Empty; en cualquier otro caso, devuelve False. Se devuelve False siempre que expresión contenga más de una variable. La función IsEmpty solamente devuelve información significativa para datos de tipo Variant.

Ejemplo

Dim MiVar, MiPrueba
MiPrueba = IsEmpty(MiVar) ' Devuelve True.
MiVar = Null ' Asigna Null.
MiPrueba = IsEmpty(MiVar) ' Devuelve False.
MiVar = Empty ' Asigna Empty.
MiPrueba = IsEmpty(MiVar) ' Devuelve True.

IS_NUMERIC (Función)

Devuelve un valor de tipo Boolean que indica si una expresión se puede evaluar como un número.

Sintaxis

    IS_NUMERIC(expresión)

El argumento expresión requerido, es un tipo de datos Variant que contiene una expresión numérica o una expresión de tipo cadena.

Comentarios

La función IS_NUMERIC devuelve True si la expresión completa se reconoce como un número; en otro caso, devuelve False.

La función IS_NUMERIC devuelve False si expresión es una expresión de fecha.

Ejemplo

Dim MiVar, MiPrueba
MiVar = "53" ' Asigna un valor.
MiPrueba = IS_NUMERIC(MiVar) ' Devuelve True.
MiVar = "459.95" ' Asigna valor.
MiPrueba = IS_NUMERIC(MiVar) ' Devuelve True.
MiVar = "45 Help" ' Asigna valor.
MiPrueba = IS_NUMERIC(MiVar) ' Devuelve False.
**LCase (Función)**

Devuelve un tipo String que se ha convertido a minúsculas.

**Sintaxis**

```
LCase(cadena)
```

El argumento cadena es cualquier expresión de cadena válida. Si string contiene Null, se devuelve Null.

**Comentarios**

Sólo se convierten a minúsculas las letras mayúsculas. Las letras minúsculas o los caracteres que no son letras no sufren cambios.

**Ejemplo**

Dim UpperCase, LowerCase
Uppercase = "Hola Mundo 1234"  ' Cadena a convertir.
Lowercase = LCase(UpperCase)   ' Devuelve "hola mundo 1234".

---

**Left (Función)**

Devuelve un tipo Variant (String) que contiene un número especificado de caracteres del lado izquierdo de una cadena.

**Sintaxis**

```
Left(string, length)
```

La sintaxis de la función Left tiene estos argumentos con nombre:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>Requerido. Expresión de cadena de la cual se devuelven los caracteres que están más a la izquierda. Si string contiene Null, se devuelve Null.</td>
</tr>
<tr>
<td>length</td>
<td>Requerido, un tipo Variant (Long). Expresión numérica que indica cuántos caracteres se van a devolver. Si es 0, devuelve una cadena de longitud cero (&quot;&quot;). Si es mayor o igual al número de caracteres en string, se devuelve la cadena entera.</td>
</tr>
</tbody>
</table>

**Comentarios**

Para determinar el número de caracteres en string, utilice la función Len.

**Nota** Utilice la función LeftB con datos de byte contenidos en una cadena. En lugar de especificar el número de caracteres que se van a devolver, length especifica el número de bytes.

**Ejemplo**

Dim UnaCadena, MiCadena
UnaCadena = "Hola Mundo"  ' Define la cadena.
MiCadena = Left(UnaCadena, 1)  ' Devuelve "H".
MiCadena = Left(UnaCadena, 6)  ' Devuelve "Hola M".
MiCadena = Left(UnaCadena, 10)  ' Devuelve "Hola Mundo".
Len (Función)

Devuelve un tipoLong que contiene el número de caracteres en una cadena o el número de bytes necesarios para almacenar una variable.

Sintaxis

**Len(cadena | nombrevar)**

La sintaxis de la función Len consta de las siguientes partes:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>cadena</td>
<td>Cualquier expresión de cadena válida. Si cadena contiene Null, se devuelve Null.</td>
</tr>
<tr>
<td>Nombrevar</td>
<td>Cualquier nombre de variable válido. Si nombrevar contiene Null, se devuelve Null. Si nombrevar es un tipo Variant, Len actúa igual que con un tipo String y siempre devuelve el número de caracteres que contiene.</td>
</tr>
</tbody>
</table>

Comentarios

Se deben especificar uno (y sólo uno) de los dos argumentos posibles. Contipos definidos por el usuario, Len devuelve el tamaño como se escribirá en el archivo.

Nota  Utilice la función LenB con datos de byte incluidos en una cadena, como en los lenguajes del conjunto de caracteres de doble byte (DBCS). En lugar de devolver el número de caracteres en una cadena, LenB devuelve el número de bytes utilizados para representar esa cadena. Con tipos de datos definidos por el usuario, LenB devuelve el tamaño en memoria, incluyendo cualquier relleno entre código que utiliza LenB, vea el segundo ejemplo en el tema de ejemplo.

Nota  Puede que Len no sea capaz de determinar el número real requerido de bytes de almacenamiento cuando se utiliza con cadenas de longitud variable entipos de datos definidos por el usuario.

Ejemplo

En este ejemplo se utiliza la función Len para devolver el número de caracteres en una cadena o el número de bytes requeridos para almacenar una variable. El bloque Type...End Type que define RegistroCliente debe ir precedido por la palabra clave Private si aparece en un módulo de clase. En un módulo estándar, una instrucción Type puede ser Public.

```vba
Type RegistroCliente     ' Define un tipo definido por el usuario.
    ID As Integer      ' Incluye esta definición en
    Name As String * 10 ' un módulo estándar.
    Address As String * 30
End Type

Dim Customer As RegistroCliente   ' Declara variables.
Dim MiEntero As Integer, MyCur As Currency
Dim MiCadena, MiLong
MiCadena = "Hola a todos"      ' Inicializa la cadenas.
```
MiLong = Len(MiEntero) ' Devuelve 2.
MiLong = Len(Customer) ' Devuelve 42.
MiLong = Len(MiCadena) ' Devuelve 11.
MiLong = Len(MyCur) ' Devuelve 8.

Este segundo ejemplo utiliza LenB y una función definida por el usuario (LenMbcs) para devolver el número de caracteres byte en una cadena si se utiliza ANSI para representar la cadena.

Function LenMbcs (ByVal str as String)
    LenMbcs = LenB(StrConv(str, vbFromUnicode))
End Function

Dim MiCadena, MyLen
MiCadena = "ABc"
' Donde "A" y "B" son DBCS y "c" es SBCS.
MyLen = Len(MyString)
' Devuelve 3 - 3 caracteres en la cadena.
MyLen = LenB(MyString)
' Devuelve 6 - 6 bytes en Unicode.
MyLen = LenMbcs(MiCadena)
' Devuelve 5 - 5 bytes en ANSI.

LoadPicture (Función)
Carga un gráfico en un objeto Picture, un control PictureBox o un control Image.

Sintaxis

LoadPicture([nombreArchivo], [tamaño], [profundidadColor],[x,y])

La sintaxis de la función LoadPicture consta de las siguientes partes:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>nombreArchivo</td>
<td>Opcional. Expresión de cadena que indica un nombre de archivo. Puede incluir carpetas y unidades. Si no se indica un nombre de archivo LoadPicture borra el control Image o PictureBox.</td>
</tr>
<tr>
<td>tamaño</td>
<td>Tipo Variant opcional. Si nombreArchivo es un cursor o un archivo de icono, indica el tamaño de imagen deseado.</td>
</tr>
<tr>
<td>profundidadColor</td>
<td>Tipo Variant opcional. Si nombreArchivo es un archivo de cursor o icono, indica la profundidad de color deseada.</td>
</tr>
</tbody>
</table>
### Valores

Los valores para tamaño son:

<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>vbLPSmall</td>
<td>0</td>
<td>Icono pequeño del sistema.</td>
</tr>
<tr>
<td>vbLPLarge</td>
<td>1</td>
<td>Icono grande del sistema, determinado por el controlador de vídeo.</td>
</tr>
<tr>
<td>vbLPSmallShell</td>
<td>2</td>
<td>Tamaño del icono pequeño de Ayuda, determinado por el tamaño establecido para los títulos de los botones en la ficha Apariencia en el cuadro de diálogo Propiedades de la pantalla del Panel de control.</td>
</tr>
<tr>
<td>vbLPLargeShell</td>
<td>3</td>
<td>Tamaño del icono grande de Ayuda, determinado por la ficha Apariencia en el cuadro de diálogo Propiedades de la pantalla del Panel de control.</td>
</tr>
<tr>
<td>vbLPCustom</td>
<td>4</td>
<td>Tamaño predeterminado, valores proporcionados por los argumentos x e y</td>
</tr>
</tbody>
</table>

Los valores para profundidadColor son:

<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>vbLPDefault</td>
<td>0</td>
<td>Mejor coincidencia si se usa el archivo especificado.</td>
</tr>
<tr>
<td>vbLPMonochrome</td>
<td>1</td>
<td>2 colores.</td>
</tr>
<tr>
<td>vbLPVGAColor</td>
<td>2</td>
<td>16 colores.</td>
</tr>
<tr>
<td>vbLPColor</td>
<td>3</td>
<td>256 colores.</td>
</tr>
</tbody>
</table>

### Comentarios

Los formatos gráficos reconocidos por Visual Basic incluyen archivos de mapas de bits (.bmp), archivos de iconos (.ico), archivos de cursor (.cur), archivos de longitud codificada (.rle), metarchivos (.wmf), metarchivos mejorados (.emf), archivos GIF (.gif) y archivos JPEG (.jpg).

Es posible limpiar los gráficos de formularios, cuadros de imagen y controles de imagen si asigna LoadPicture sin argumentos.

Para cargar gráficos para presentarlos en un control PictureBox, en un control Image o como fondo de un formulario, el valor devuelto por LoadPicture debe asignarse a la propiedad Picture del objeto en el que se quiere presentar la imagen. **Por ejemplo:**

```vba
Set Picture = LoadPicture("FIESTA.BMP")
```

```vba
Set Image1.Picture = LoadPicture("FIESTA.BMP")
```

Para asignar un icono a un formulario, establezca el valor devuelto por la función LoadPicture a la propiedad Icon del objeto Form:
Set Form1.Icon = LoadPicture("MIICONO.ICO")

Los iconos también pueden asignarse a la propiedad DragIcon de todos los controles excepto los controles Timer y Menu. **Por ejemplo:**

Set Command1.DragIcon = LoadPicture("MIICONO.ICO")

Para cargar gráficos en el Portapapeles del sistema se usa LoadPicture de la forma siguiente:

Clipboard.SetData LoadPicture("FIESTA.BMP")

**Ejemplo**

Ejemplo de la propiedad LoadPicture

Este ejemplo usa la función LoadPicture para cargar una imagen en un control PictureBox y para borrar la imagen del control. Para probar este ejemplo, agregue un control PictureBox a un objeto Form, pegue el código en la sección Declaraciones del objeto Form y después ejecute el ejemplo y haga clic en el formulario.

Private Sub Form_Click()
    Dim Msg as String  ' Declara variables.
    On Error Resume Next  ' Establece el tratamiento de errores.
    Height = 3990
    Width = 4890  ' Define el alto y el ancho.
    Picture1.Picture = LoadPicture("PAPEL.CUR", vbLPCustom, vbLPColor, 32, 32)  ' Carga un cursor.
    If Err Then
        Msg = "Imposible encontrar el archivo .cur."
        MsgBox Msg  ' Presenta el mensaje de error.
        Exit Sub  ' Sale si hay error.
    End If
    MsgBox "Elija "Aceptar" para borrar el mapa de bits del cuadro de imagen."
    MsgBox Msg
    Picture1.Picture = LoadPicture()  ' vacía el cuadro de imagen.
End Sub

**Log (Función)**

Devuelve un tipo Double que especifica el logaritmo natural de un número.

Sintaxis

    Log(número)

El argumento número es un tipo Double o cualquier expresión numérica válida mayor que cero.
Comentarios
El logaritmo natural es el logaritmo en base e. El valor de la constante e es 2.718282 aproximadamente.
Puede calcular logaritmos en base-n para cualquier número x dividiendo el logaritmo natural de x por el logaritmo natural de n de la siguiente manera:
\[ \text{Log}_n(x) = \frac{\log(x)}{\log(n)} \]
El ejemplo siguiente ilustra una función personalizada (Function) que calcula logaritmos en base 10:

```vba
Static Function Log10(X)
    Log10 = Log(X) / Log(10#)
End Function
```

Ejemplo
Dim MiÁngulo, MiLog
' Define el ángulo en radianes.
MiÁngulo = 1.3
' Calcula el seno hiperbólico inverso.
MiLog = Log(MiÁngulo + Raíz(MiÁngulo * MiÁngulo + 1))

LTrim, Rtrim y Trim (Funciones)
Devuelve un tipo Variant (String) que contiene una copia de una cadena determinada sin espacios a la izquierda (LTrim), sin espacios a la derecha (RTrim) o sin espacios ni a la derecha ni a la izquierda (Trim).
Sintaxis

```
LTrim(cadena)
```

```
RTrim(cadena)
```

```
Trim(cadena)
```
El argumento cadena obligatorio es cualquier expresión de cadena válida. Si cadena contieneNull, se devuelve Null.

Ejemplo
Dim MiCadena, CadenaCorte
MiCadena = "  <Cortar-> "  ' Inicializa la cadena.
CadenaCorte = LTrim(MiCadena)  ' CadenaCorte = "<Cortar-> ".
CadenaCorte = RTrim(MiCadena)  ' CadenaCorte = "  <Cortar->".
CadenaCorte = LTrim(RTrim(MiCadena))  ' CadenaCorte = "<-Cortar->".
' El uso de la función Trim por sí sola logra el mismo resultado.
CadenaCorte = Trim(MiCadena)    ' CadenaCorte = "<-Cortar->".

**Mid (Función)**

Devuelve un tipo Variant (String) que contiene un número especificado de caracteres de una cadena.

**Sintaxis**

```
Mid(string, start[, length])
```

La sintaxis de la función Mid tiene estos argumentos con nombre:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>Requrido. Expresión de cadena de la cual se devuelven los caracteres. Si string contiene Null, se devuelve Null.</td>
</tr>
<tr>
<td>start</td>
<td>Requrido; un tipo Long. Posición de carácter en string en la cual comienza la parte que se quiere tomar. Si start es mayor que el número de caracteres en la string, Mid devuelve una cadena de longitud cero (&quot;&quot;).</td>
</tr>
<tr>
<td>length</td>
<td>Requrido; un tipo Variant (Long). Número de caracteres que se van a devolver. Si se omite o en el texto hay menos de length caracteres (incluyendo el carácter de start), se devuelven todos los caracteres desde la posición de start hasta el final de la cadena.</td>
</tr>
</tbody>
</table>

**Comentarios**

Para determinar el número de caracteres en string, utilice la función Len.

Nota  Utilice la función MidB con datos de byte incluidos en una cadena, como en los lenguajes de conjunto de caracteres de doble byte. En lugar de especificar el número de caracteres, los argumentos especifican el número de bytes. Para ver código de ejemplo que utiliza MidB, vea el segundo ejemplo en el tema de bytes.

**Ejemplo**

En este ejemplo se utiliza la función Mid para devolver un número especificado de caracteres de una cadena.

Dim MiCadena, PrimeraPalabra, UltimaPalabra, PalabraMedia
MiCadena = "Demostración función Mid"  ' Crea la cadena de texto.
PrimeraPalabra = Mid(MiCadena, 1, 12)  ' Devuelve "Demostración".
UltimaPalabra = Mid(MiCadena, 21, 3)  ' Devuelve "Mid".
PalabraMedia = Mid(MiCadena, 14)  ' Devuelve "función Mid".

Este segundo ejemplo utiliza MidB y una función definida por el usuario (MidMbcs) para devolver también caracteres de una cadena. La diferencia en este ejemplo es que la cadena de entrada es ANSI y la longitud en bytes.

Function MidMbcs(ByVal str as String, start, length)
MidMbcs = StrConv(MidB(StrConv(str, vbFromUnicode), start, length), vbUnicode)
End Function

Dim MiCadena
MiCadena = "AbCdEfG"
' Donde "A", "C", "E", y "G" son DBCS y "b", "d",
' y "f" son SBCS.
MiNuevaCadena = Mid(MiCadena, 3, 4)
' Devuelve "CdEf"
MiNuevaCadena = MidB(MiCadena, 3, 4)
' Devuelve "bC"
MiNuevaCadena = MidMbcs(MiCadena, 3, 4)
' Devuelve "bCd"

Minute (Función)
Devuelve un valor de tipo Variant (Integer) que especifica un número entero entre 0 y 59, inclusive, que representa el minuto de la hora.
Sintaxis

Minute(hora)

El argumento hora es cualquier valor de tipo Variant, una expresión numérica, expresión de tipo cadena o cualquier combinación que pueda representar una hora. Si hora contiene Null, la función devolverá Null.

Ejemplo
En este ejemplo se utiliza la función Minute para obtener los minutos de una hora especificada. En el entorno de desarrollo, el literal de hora se presentará en formato abreviado, utilizando la configuración regional del código.
Dim MiTiempo, MiMinuto
MiTiempo = #4:35:17 PM# ' Asigna una hora.
MiMinuto = Minute(MiTiempo) ' MiMinuto contiene 35.

Month (Función)
Devuelve un valor de tipo Variant (Integer) que especifica un número entero entre 1 y 12, inclusive, que representa el mes del año.
Sintaxis

Month(fecha)

El argumento fecha es cualquier valor de tipo Variant, una expresión numérica, expresión de tipo
cadena o cualquier combinación, que pueda representar una fecha. Si fecha contiene Null, la función devolverá Null.

**Ejemplo**

Dim MiFecha, MiMes
MiFecha = #12 febrero 1969# ' Asigna una fecha.
MiMes = Month(MiFecha) ' MiMes contiene 2.

---

**MonthName (Función)**

Devuelve una cadena que indica el mes especificado.

**Sintaxis**

MonthName(mes [, abreviar])

La sintaxis de la función MonthName consta de las siguientes partes:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>mes</td>
<td>Requerido. La designación numérica del mes. Por ejemplo, enero es 1, febrero es 2, etcétera.</td>
</tr>
<tr>
<td>abreviar</td>
<td>Opcional. El valor Boolean que indica si se va a abreviar el nombre del mes. Si se omite, el valor predeterminado es False, que significa que el nombre del mes no se abrevia.</td>
</tr>
</tbody>
</table>

---

**MsgBox (Función)**

Muestra un mensaje en un cuadro de diálogo, espera a que el usuario haga clic en un botón y devuelve un tipo Integer correspondiente al botón elegido por el usuario.

**Sintaxis**

MsgBox(prompt[, buttons][, title][, helpfile, context])

La sintaxis de la función MsgBox consta de estos argumentos con nombre:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>prompt</td>
<td>Requerido. Expresión de cadena que representa el prompt en el cuadro de diálogo. La longitud máxima de prompt es de aproximadamente 1024 caracteres, según el ancho de los caracteres utilizados. Si prompt consta de más de una línea, puede separarlos utilizando un carácter de retorno de carro (Chr(13)) o un carácter de avance de línea (Chr(10)), o una combinación de caracteres de retorno de carro – avance de línea (Chr(13) y Chr(10)) entre cada línea y la siguiente.</td>
</tr>
<tr>
<td>buttons</td>
<td>Opcional. Expresión numérica que corresponde a la suma de los valores que especifican el número y el tipo de los botones que se pretenden mostrar, el estilo de ícono que se va a utilizar, la identidad del botón predeterminado y la modalidad del cuadro de mensajes. Si se omite este argumento, el valor predeterminado para buttons es 0.</td>
</tr>
<tr>
<td>title</td>
<td>Opcional. Expresión de cadena que se muestra en la barra de título del cuadro de diálogo. Si se omite title, en la barra de título se coloca el nombre de la aplicación.</td>
</tr>
</tbody>
</table>
helpfile | Opcional. Expresión de cadena que identifica el archivo de Ayuda que se utiliza para proporcionar ayuda interactiva en el cuadro de diálogo. Si se especifica helpfile, también se debe especificar context.

context | Opcional. Expresión numérica que es igual al número de contexto de Ayuda asignado por el autor al tema de Ayuda correspondiente. Si se especifica context, también se debe especificar helpfile.

Valores

El argumento buttons tiene estos valores:

<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>VbOKOnly</td>
<td>0</td>
<td>Muestra solamente el botón Aceptar.</td>
</tr>
<tr>
<td>VbOKCancel</td>
<td>1</td>
<td>Muestra los botones Aceptar y Cancelar.</td>
</tr>
<tr>
<td>VbAbortRetryIgnore</td>
<td>2</td>
<td>Muestra los botones Anular, Reintentar e Ignorar.</td>
</tr>
<tr>
<td>VbYesNoCancel</td>
<td>3</td>
<td>Muestra los botones Sí, No y Cancelar.</td>
</tr>
<tr>
<td>VbYesNo</td>
<td>4</td>
<td>Muestra los botones Sí y No.</td>
</tr>
<tr>
<td>VbRetryCancel</td>
<td>5</td>
<td>Muestra los botones Reintentar y Cancelar.</td>
</tr>
<tr>
<td>VbCritical</td>
<td>16</td>
<td>Muestra el icono de mensaje crítico.</td>
</tr>
<tr>
<td>VbQuestion</td>
<td>32</td>
<td>Muestra el icono de pregunta de advertencia.</td>
</tr>
<tr>
<td>VbExclamation</td>
<td>48</td>
<td>Muestra el icono de mensaje de advertencia.</td>
</tr>
<tr>
<td>VbInformation</td>
<td>64</td>
<td>Muestra el icono de mensaje de información.</td>
</tr>
<tr>
<td>VbDefaultButton1</td>
<td>0</td>
<td>El primer botón es el predeterminado.</td>
</tr>
<tr>
<td>VbDefaultButton2</td>
<td>256</td>
<td>El segundo botón es el predeterminado.</td>
</tr>
<tr>
<td>VbDefaultButton3</td>
<td>512</td>
<td>El tercer botón es el predeterminado.</td>
</tr>
<tr>
<td>VbDefaultButton4</td>
<td>768</td>
<td>El cuarto botón es el predeterminado.</td>
</tr>
<tr>
<td>VbApplicationModal</td>
<td>0</td>
<td>Aplicación modal; el usuario debe responder al cuadro de mensajes antes de poder seguir trabajando en la aplicación actual.</td>
</tr>
<tr>
<td>VbSystemModal</td>
<td>4096</td>
<td>Sistema modal; se suspenden todas las aplicaciones hasta que el usuario responda al cuadro de mensajes.</td>
</tr>
<tr>
<td>VbMsgBoxHelpButton</td>
<td>16384</td>
<td>Agrega el botón Ayuda al cuadro de mensaje.</td>
</tr>
<tr>
<td>VbMsgBoxSetForeground</td>
<td>65536</td>
<td>Especifica la ventana del cuadro de mensaje como la ventana de primer plano.</td>
</tr>
<tr>
<td>VbMsgBoxRight</td>
<td>524288</td>
<td>El texto se alinea a la derecha.</td>
</tr>
<tr>
<td>VbMsgBoxRtlReading</td>
<td>1048576</td>
<td>Especifica que el texto debe aparecer para ser leído de derecha a izquierda en sistemas hebreo y árabe.</td>
</tr>
</tbody>
</table>

El primer grupo de valores (0 a 5) describe el número y el tipo de los botones mostrados en el cuadro de diálogo; el segundo grupo (16, 32, 48, 64) describe el estilo del icono, el tercer grupo (0, 256, 512) determina el botón predeterminado y el cuarto grupo (0, 4096) determina la modalidad del cuadro de mensajes. Cuando se suman números para obtener el valor final del argumento buttons, se utiliza solamente un número de cada grupo.

Nota  Estas constantes las especifica Visual Basic for Applications. Por tanto, el nombre de las mismas puede utilizarse en cualquier lugar del código en vez de sus valores reales.

Valores devueltos
<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>vbOK</td>
<td>1</td>
<td>Aceptar</td>
</tr>
<tr>
<td>vbCancel</td>
<td>2</td>
<td>Cancelar</td>
</tr>
<tr>
<td>vbAbort</td>
<td>3</td>
<td>Anular</td>
</tr>
<tr>
<td>vbRetry</td>
<td>4</td>
<td>Reintentar</td>
</tr>
<tr>
<td>vbIgnore</td>
<td>5</td>
<td>Ignorar</td>
</tr>
<tr>
<td>vbYes</td>
<td>6</td>
<td>Sí</td>
</tr>
<tr>
<td>vbNo</td>
<td>7</td>
<td>No</td>
</tr>
</tbody>
</table>

**Comentarios**

Cuando se proporcionan tanto helpfile como context, el usuario puede presionar F1 para ver el tema de Ayuda correspondiente al context. Algunas aplicaciones host, por ejemplo Microsoft Excel, también agregan automáticamente un botón Ayuda al cuadro de diálogo.

Si el cuadro de diálogo cuenta con un botón Cancelar, presionar la tecla ESC tendrá el mismo efecto que hacer clic en este botón. Si el cuadro de diálogo contiene un botón Ayuda, se suministra ayuda interactiva para ese cuadro de diálogo. Sin embargo, no se devuelve valor alguno hasta que se hace clic en uno de estos botones.

Nota: Si desea especificar más que el primer argumento con nombre, debe utilizar MsgBox en una expresión. Si desea omitir algún argumento de posición, debe incluir el delimitador de coma correspondiente.

**Ejemplo**

En este ejemplo se utiliza la función MsgBox para mostrar un mensaje de error crítico en un cuadro de diálogo con botones Sí y No. El botón No se considera la respuesta predeterminada. El valor devuelto por la función MsgBox depende del botón elegido por el usuario. En este ejemplo, se supone que DEMO.HLP es un archivo de Ayuda que contiene un tema con un número de contexto igual a 1000.

```vbs
Dim Mensaje, Estilo, Título, Ayuda, Ctxt, Respuesta, MiCadena
Mensaje = "¿Desea continuar?" ' Define el mensaje.
Estilo = vbYesNo + vbCritical + vbDefaultButton2 ' Define los botones.
Título = "Demostración de MsgBox" ' Define el título.
Ayuda = "DEMO.HLP" ' Define el archivo de ayuda.
Ctxt = 1000 ' Define el tema
   ' el contexto
   ' Muestra el mensaje.
Respuesta = MsgBox(Mensaje, Estilo, Título, Ayuda, Ctxt)
If Respuesta = vbYes Then ' El usuario eligió el botón Sí.
   MiCadena = "Sí" ' Ejecuta una acción.
Else ' El usuario eligió el botón No.
   MiCadena = "No" ' Ejecuta una acción.
```
End If

**Now (Función)**
Devuelve un valor de tipo Variant (Date) que especifica la fecha y hora actuales de acuerdo con la configuración de la fecha y la hora del sistema de su equipo.

Sintaxis

```
Now
```

**Ejemplo**

```
Dim Hoy
Hoy = Now  ' Obtiene la fecha y la hora del sistema actual.
```

**QBColor (Función)**
Devuelve un tipoLong que representa el código de color RGB correspondiente al número de color especificado.

Sintaxis

```
QBColor(color)
```

El argumento color es un número entero entre 0-15.

**Valores**
El argumento color toma los siguientes valores:

<table>
<thead>
<tr>
<th>Número</th>
<th>Color</th>
<th>Número</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Negro</td>
<td>8</td>
<td>Gris</td>
</tr>
<tr>
<td>1</td>
<td>Azul</td>
<td>9</td>
<td>Azul claro</td>
</tr>
<tr>
<td>2</td>
<td>Verde</td>
<td>10</td>
<td>Verde claro</td>
</tr>
<tr>
<td>3</td>
<td>Aguamarina</td>
<td>11</td>
<td>Aguamarina claro</td>
</tr>
<tr>
<td>4</td>
<td>Rojo</td>
<td>12</td>
<td>Rojo claro</td>
</tr>
<tr>
<td>5</td>
<td>Fucsia</td>
<td>13</td>
<td>Fucsia claro</td>
</tr>
<tr>
<td>6</td>
<td>Amarillo</td>
<td>14</td>
<td>Amarillo claro</td>
</tr>
<tr>
<td>7</td>
<td>Blanco</td>
<td>15</td>
<td>Blanco brillante</td>
</tr>
</tbody>
</table>

**Comentarios**
El argumento color representa los valores de color que utilizan otras versiones de Basic (como Microsoft Visual Basic para MS-DOS y Basic Compiler). Comenzando por el byte menos significativo, el valor devuelto especifica los valores de rojo, verde y azul necesarios para establecer el color correspondiente en el sistema RGB que utiliza Visual Basic para Aplicaciones.

**Ejemplo**
Este ejemplo utiliza la función QBColor para cambiar la propiedad BackColor del formulario transferido como MiFormulario al color indicado por CódigoColor. QBColor acepta valores enteros entre 0 y 15.
Sub CambiarColorFondo (CódigoColor As Integer, MiFormulario As Form)
    MiFormulario.BackColor = QBColor(CódigoColor)
End Sub

---

**Replace (Función)**

Devuelve una cadena en la que se reemplazó una subcadena específica con otra subcadena un número especificado de veces.

**Sintaxis**

```
Replace(expresión, encontrar, reemplazarCon [, inicio[, contar[, comparar]]])
```

La sintaxis de la función Replace consta de las siguientes partes:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>expresión</td>
<td>Requerido. Una expresión de cadena que contiene la subcadena que reemplazar.</td>
</tr>
<tr>
<td>encontrar</td>
<td>Requerido. La subcadena que se busca.</td>
</tr>
<tr>
<td>reemplazarCon</td>
<td>Requerido. La subcadena de sustitución.</td>
</tr>
<tr>
<td>inicio</td>
<td>Opcional. La posición dentro de la expresión donde se empezará a buscar la subcadena. Si se omite, se supone que es 1.</td>
</tr>
<tr>
<td>contar</td>
<td>Opcional. El número de sustituciones de subcadenas que se realizarán. Si se omite, el valor predeterminado es –1, que significa que se hagan todas las sustituciones posibles.</td>
</tr>
<tr>
<td>comparar</td>
<td>Opcional. El valor numérico que indica la clase de comparación que se realizará cuando se evalúen las subcadenas. Vea la sección Valores para consultar los valores posibles.</td>
</tr>
</tbody>
</table>

**Valores**

El argumento comparar puede tener los valores siguientes:

<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>vbUseCompareOption</td>
<td>1</td>
<td>Realizar una comparación utilizando el valor de la instrucción Option Compare.</td>
</tr>
<tr>
<td>vbBinaryCompare</td>
<td>0</td>
<td>Realiza una comparación binaria.</td>
</tr>
<tr>
<td>vbTextCompare</td>
<td>1</td>
<td>Realizar una comparación basada en el texto.</td>
</tr>
<tr>
<td>vbDatabaseCompare</td>
<td>2</td>
<td>Sólo para Microsoft Access. Realizar una comparación basada en la información de su base de datos.</td>
</tr>
</tbody>
</table>

**Valores de retorno**

La función Replace devuelve los valores siguientes:

<table>
<thead>
<tr>
<th>Si</th>
<th>Reemplazar devuelve</th>
</tr>
</thead>
<tbody>
<tr>
<td>expresión es de longitud cero</td>
<td>Una cadena de longitud cero (&quot;&quot;).</td>
</tr>
<tr>
<td>expresión es Null</td>
<td>Un error.</td>
</tr>
<tr>
<td>encontrar es de longitud cero</td>
<td>Una copia de expresión.</td>
</tr>
<tr>
<td>reemplazarCon es de longitud cero</td>
<td>Una copia de expresión con todas las apariciones de encontrar quitadas.</td>
</tr>
<tr>
<td>inicio &gt; Len(expresión)</td>
<td>Una cadena de longitud cero.</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>contar es 0</td>
<td>Una copia de expresión.</td>
</tr>
</tbody>
</table>

**Comentarios**

El valor de retorno de la función Replace es una cadena con sustituciones hechas que comienza en la posición especificada por inicio y concluye al final de la cadena expresión. No es una copia de la cadena original desde el inicio al final.

**RGB (Función)**

Devuelve un número entero tipoLong que representa un valor de color RGB.

**Sintaxis**

RGB(red, green, blue)

La sintaxis de la función RGB consta de los siguientes argumentos:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>red</td>
<td>Obligatorio. tipo Variant (Integer). Número en el intervalo de 0-255, ambos inclusive, que representa el componente rojo del color.</td>
</tr>
<tr>
<td>green</td>
<td>Obligatorio; tipo Variant (Integer). Número en el intervalo de 0-255, ambos inclusive, que representa el componente verde del color.</td>
</tr>
<tr>
<td>blue</td>
<td>Obligatorio; tipo Variant (Integer). Número en el intervalo de 0-255, ambos inclusive, que representa el componente azul del color.</td>
</tr>
</tbody>
</table>

**Comentarios**

Los métodos de aplicación y las propiedades que aceptan una especificación de color esperan que ésta sea un número que represente un valor de color RGB. Un valor de color RGB especifica la intensidad relativa de rojo, verde y azul que hacen que se muestre un color determinado.

El valor de cualquier argumento para la función RGB que sea superior a 255 se considerará como 255.

La siguiente tabla muestra algunos colores estándar y sus valores de rojo, verde y azul:

<table>
<thead>
<tr>
<th>Color</th>
<th>Valor de rojo</th>
<th>Valor de verde</th>
<th>Valor de azul</th>
</tr>
</thead>
<tbody>
<tr>
<td>Negro</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Azul</td>
<td>0</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>Verde</td>
<td>0</td>
<td>255</td>
<td>0</td>
</tr>
<tr>
<td>Cián</td>
<td>0</td>
<td>255</td>
<td>255</td>
</tr>
<tr>
<td>Rojo</td>
<td>255</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Magenta</td>
<td>255</td>
<td>0</td>
<td>255</td>
</tr>
<tr>
<td>Amarillo</td>
<td>255</td>
<td>255</td>
<td>0</td>
</tr>
<tr>
<td>Blanco</td>
<td>255</td>
<td>255</td>
<td>255</td>
</tr>
</tbody>
</table>

**Ejemplo**

Este ejemplo muestra cómo se utiliza la función RGB para devolver un número entero que representa un valor de color RGB. Se utiliza por aquellos métodos y propiedades de aplicación que aceptan una especificación de color. El objeto MiObjeto y su propiedad se utilizan sólo como
ejemplo. Si MiObjeto no existe o si no tiene una propiedad Color, se produce un error.

Dim Rojo, I, ValorRGB, MiObjeto
Red = RGB(255, 0, 0)   ' Devuelve el valor de Rojo.
I = 75    ' Inicializa un desplazamiento.
ValorRGB = RGB(I, 64 + I, 128 + I) ' Igual que RGB(75, 139, 203).
MiObjeto.Color = RGB(255, 0, 0)   ' Establece la propiedad Color de
   ' MiObjeto a Rojo.

---

**Right (Función)**

Devuelve un tipo Variant (String) que contiene un número especificado de caracteres del lado derecho de una cadena.

Sintaxis

```
Right(string, lenght)
```

La sintaxis de la función Right tiene estos argumentos con nombre:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>Requerido. Expresión de cadena desde la cual se devuelven los caracteres que están más a la derecha. Si string contiene Null, se devuelve Null.</td>
</tr>
<tr>
<td>lenght</td>
<td>Requerido, un tipo Variant (Long). Expresión numérica que indica cuántos caracteres se van a devolver. Si es 0, se devuelve una cadena de longitud cero (&quot;&quot;&quot;). Si es mayor o igual al número de caracteres en string, se devuelve la cadena completa.</td>
</tr>
</tbody>
</table>

**Comentarios**

Para determinar el número de caracteres en string, utilice la función Len.

Nota   Utilice la función RightB con datos de byte incluidos en una cadena. En lugar de especificar el número de caracteres para devolver, longitud especifica el número de bytes.

**Ejemplo**

Dim UnaCadena, MiCadena
UnaCadena = "Hola Mundo"  ' Define una cadena.
MiCadena = Right(UnaCadena, 1)  ' Devuelve "o".
MiCadena = Right(UnaCadena, 6)  ' Devuelve "Mundo".
MiCadena = Right(UnaCadena, 20)  ' Devuelve "Hola Mundo".

---

**Rnd (Función)**

Devuelve un tipo Single que contiene un número aleatorio.

Sintaxis

```
Rnd[(número)]
```
El argumento opcional número es un tipo Single o cualquier expresión numérica válida.

**Valores devueltos**

<table>
<thead>
<tr>
<th>Si número es</th>
<th>Rnd genera</th>
</tr>
</thead>
<tbody>
<tr>
<td>Menor que cero</td>
<td>El mismo número todas las veces, utilizando número como valor desemilla.</td>
</tr>
<tr>
<td>Mayor que cero</td>
<td>El siguiente número aleatorio de la secuencia.</td>
</tr>
<tr>
<td>Igual a cero</td>
<td>El número generado más recientemente.</td>
</tr>
<tr>
<td>No se especifica</td>
<td>El siguiente número aleatorio de la secuencia</td>
</tr>
</tbody>
</table>

**Comentarios**

La función Rnd devuelve un valor menor que 1 pero mayor o igual que cero.

El valor de número determina cómo la función Rnd genera un número aleatorio:

Para cualquier valor de semilla inicial se genera la misma secuencia de números. Esto es debido a que cada llamada sucesiva a la función Rnd usará el número anterior como valor de semilla para el siguiente número de la secuencia.

Antes de llamar a la función Rnd, utilice la instrucción **Randomize** sin argumento para inicializar el generador de números aleatorios con un valor de semilla basado en el reloj del sistema.

Para producir enteros aleatorios en un intervalo dado, use esta fórmula:

\[ \text{Int}((\text{Límite}\_\text{superior} - \text{Límite}\_\text{inferior} + 1) \times \text{Rnd} + \text{Límite}\_\text{inferior}) \]

Aquí, **Límite\_superior** es el número mayor del intervalo y **Límite\_inferior** es el número menor del intervalo.

**Nota** Para repetir secuencias de números aleatorios, llame a la función Rnd con un argumento negativo antes de utilizar la función Randomize con un argumento numérico. Al utilizar la instrucción Randomize con el mismo valor de número no se repite la secuencia anterior.

**Ejemplo**

En este ejemplo se utiliza la función Rnd para generar un valor entero aleatorio de 1 a 6.

Dim MiValor

\[ \text{MiValor} = \text{Int}(6 \times \text{Rnd}) + 1 \]  `Genera un valor aleatorio entre 1 y 6.

**Randomize (Instrucción)**

Inicializa el generador de números aleatorios.

Sintaxis

**Randomize [número]**

El argumento opcional número es un tipo Variant o cualquier expresión numérica válida.

**Comentarios**

La instrucción **Randomize** utiliza número para inicializar el generador de números aleatorios de la función Rnd y le asigna un nuevo valor de semilla. Si omite número, el valor devuelto por el reloj del sistema se usa como el nuevo valor de semilla.

Si no utiliza la instrucción **Randomize**, la función Rnd (sin argumentos) utiliza el mismo número como valor de semilla la primera vez que se la invoca, usando después como valor
de semilla el último número generado.

Nota    Para repetir secuencias de números aleatorios, llame a la función Rnd con un
argumento negativo antes de utilizar la instrucción Randomize con un argumento
numérico. Al utilizar la instrucción Randomize con el mismo valor de número no se
repite la secuencia anterior.

Ejemplo
Dim MiValor
Randomize  ' Inicializa el generador de números aleatorios.
MiValor = Int((6 * Rnd) + 1)  ' Genera valores aleatorios entre 1 y 6.

**Round (Función)**

Devuelve un número redondeado en el número especificado de lugares decimales.

Sintaxis

`Round(expresión [, númLugaresDecimales])`

La sintaxis de la función Round consta de las siguientes partes:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>expresión</td>
<td>Requiere. La expresión numérica que se redondea.</td>
</tr>
<tr>
<td>númLugaresDecimales</td>
<td>Opcional. Un número que indica cuántos lugares a la derecha del decimal se incluyen en el redondeo. Si se omite, la función Round devuelve enteros.</td>
</tr>
</tbody>
</table>

**Second (Función)**

Devuelve un valor de tipo Variant (Integer) que especifica un número entero entre 0 y 59, inclusive, que representa un segundo.

Sintaxis

`Second(hora)`

El argumento hora es cualquier valor de tipo Variant, una expresión numérica, expresión de tipo cadena o cualquier combinación, que pueda representar una hora. Si hora contiene Null, la función devolverá Null.

Ejemplo
Dim MiTiempo, MiSegundo
MiTiempo = #4:35:17 PM#  ' Asigna una hora.
MiSegundo = Second(MiTiempo)  ' MiSegundo contiene 17.

**Sgn (Función)**

Devuelve un tipo Variant (Integer) que indica el signo de un número.
Sintaxis

Sgn(número)

El argumento número puede ser cualquier expresión numérica válida.

Valores devueltos

<table>
<thead>
<tr>
<th>Si número es Sgn</th>
<th>devuelve</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mayor que cero</td>
<td>1</td>
</tr>
<tr>
<td>Igual a cero</td>
<td>0</td>
</tr>
<tr>
<td>Menor que cero -</td>
<td>1</td>
</tr>
</tbody>
</table>

Comentarios

El signo del argumento número determina el valor devuelto por la función Sgn.

Ejemplo

Dim MiVar1, MiVar2, MiVar3, MiSigno
MiVar1 = 12: MiVar2 = -2.4: MiVar3 = 0
MiSigno = Sgn(MiVar1) ' Devuelve 1.
MiSigno = Sgn(MiVar2) ' Devuelve -1.
MiSigno = Sgn(MiVar3) ' Devuelve 0.

Sin (Función)

Devuelve un tipo Double que especifica el seno de un ángulo.

Sintaxis

Sin(número)

El argumento número es un tipo Double o cualquier expresión numérica válida que expresa un ángulo en radianes.

Comentarios

La función Sin toma un ángulo y devuelve la razón de dos lados de un triángulo rectángulo. La razón es la longitud del lado opuesto al ángulo dividido por la longitud de la hipotenusa.

El resultado entra dentro del intervalo -1 a 1.

Para convertir grados en radianes, multiplique los grados por pi/180. Para convertir radianes en grados, multiplique los radianes por 180/pi.

Ejemplo

Dim MiÁngulo, MiCosecante
MiÁngulo = 1.3 ' Define el ángulo en radianes.
MiCosecante = 1 / Sin(MiÁngulo) ' Calcula la cosecante.
**Space (Función)**

Devuelve un tipo Variant (String) que consiste en un número especificado de espacios.

**Sintaxis**

```
Space(número)
```

El argumento número es el número de espacios que se desea incluir en la cadena.

**Comentarios**

La función Space es útil para dar formato a la salida y para borrar datos en cadenas de longitud fija.

**Ejemplo**

Dim MiCadena
' Devuelve una cadena con 10 espacios.
MiCadena = Space(10)
' Inserta 10 espacios entre 2 cadenas.
MiCadena = "Hola" & Space(10) & "Mundo"

---

**Sqr (Función)**

Devuelve un tipo Double que especifica la raíz cuadrada de un número.

**Sintaxis**

```
Sqr(número)
```

El número argumento es un tipo Double o cualquier expresión numérica válida mayor o igual a cero.

**Ejemplo**

Dim MiRaíz
MiRaíz = Sqr(4)   ' Devuelve 2.
MiRaíz = Sqr(23)   ' Devuelve 4.79583152331272.
MiRaíz = Sqr(0)   ' Devuelve 0.
MiRaíz = Sqr(-4)   ' Genera un error en tiempo de ejecución.

---

**Str (Función)**

Devuelve en un tipo Variant (String) la representación de cadena de un número.

**Sintaxis**

```
Str(número)
```

El argumento número necesario es un tipo Long que contiene una expresión numérica válida.

**Comentarios**

Cuando los números se convierten a cadenas, siempre se reserva un espacio a la izquierda para el signo del número. Si número es positivo, la cadena devuelta contiene un espacio a la izquierda y el
Utilice la función Format para convertir valores numéricos en valores con formato de fecha, hora, moneda o en otros formatos definidos por el usuario. A diferencia de Str, la función Format no incluye un espacio a la izquierda para el signo del número.

Nota   La función Str sólo reconoce el punto (.) como separador decimal válido. Cuando exista la posibilidad de que se pueda utilizar un separador decimal diferente (por ejemplo, en aplicaciones internacionales), se debe utilizar CStr para convertir un número a una cadena.

**Ejemplo**

Dim MiCadena

MiCadena = Str(459)   ' Devuelve " 459".
MiCadena = Str(-459.65)   ' Devuelve "-459,65".
MiCadena = Str(459.001)   ' Devuelve " 459,001".

### StrComp (Función)

Devuelve un tipo Variant (Integer) que indica el resultado de una comparación de cadena.

**Sintaxis**

\[
\text{StrComp}(\text{string1}, \text{string2}[, \text{compare}])
\]

La sintaxis de la función StrComp tiene estos argumentos con nombre:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>string1</td>
<td>Requerido. Cualquier expresión de cadena válida.</td>
</tr>
<tr>
<td>string2</td>
<td>Requerido. Cualquier expresión de cadena válida.</td>
</tr>
<tr>
<td>compare</td>
<td>Opcional. Especifica el tipo de comparación de cadena. Si el argumento compare es nulo el valor de Option Compare determina el tipo de comparación.</td>
</tr>
</tbody>
</table>

**Valores**

Los valores de la instrucción compare son los siguientes:

<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>vbUseCompareOption</td>
<td>1</td>
<td>Realiza una comparación con los valores de la instrucción Option Compare.</td>
</tr>
<tr>
<td>vbBinaryCompare</td>
<td>0</td>
<td>Realiza una comparación binaria.</td>
</tr>
<tr>
<td>vbTextCompare</td>
<td>1</td>
<td>Realiza una comparación textual.</td>
</tr>
<tr>
<td>vbDatabaseCompare</td>
<td>2</td>
<td>Sólo para Microsoft Access. Realiza una comparación basada en la información en las bases de datos.</td>
</tr>
</tbody>
</table>

**Valores devueltos**

La función StrComp devuelve los siguientes valores:

<table>
<thead>
<tr>
<th>Si La función</th>
<th>StrComp devuelve</th>
</tr>
</thead>
<tbody>
<tr>
<td>string1 es menor que string2</td>
<td>1</td>
</tr>
<tr>
<td>string1 es igual a string2</td>
<td>0</td>
</tr>
</tbody>
</table>
string1 es mayor que string2 | 1  
string1 o string2 es Null | Null

**Ejemplo**

Dim MiCadena1, MiCadena2, MiComp  
MiCadena1 = "ABCD": MiCadena2 = "abcd"  
' Define las variables.  
MiComp = StrComp(MiCadena1, MiCadena2, 1)  
' Devuelve 0.  
MiComp = StrComp(MiCadena1, MiCadena2, 0)  
' Devuelve -1.  
MiComp = StrComp(MiCadena2, MiCadena1)  
' Devuelve 1.

---

**StrConv (Función)**

Devuelve un tipo Variant (String) que se convierte como se especifica.

**Sintaxis**

**StrConv(string, conversion, LCID)**

La sintaxis de la función StrConv tiene estos argumentos con nombre:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>Requerido. Expresión de cadena que se va a convertir.</td>
</tr>
<tr>
<td>Conversion</td>
<td>Requerido. Un tipo Integer. La suma de los valores que especifica el tipo de conversión que se va a realizar.</td>
</tr>
<tr>
<td>LCID</td>
<td>Opcional. La LocaleID, si es diferente de la LocaleID del sistema. (La LocaleID del sistema es la predeterminada.)</td>
</tr>
</tbody>
</table>

**Valores**

Los valores del argumento conversion son:

<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>vbUpperCase</td>
<td>1</td>
<td>Convierte la cadena a caracteres en mayúscula.</td>
</tr>
<tr>
<td>vbLowerCase</td>
<td>2</td>
<td>Convierte la cadena a caracteres en minúscula.</td>
</tr>
<tr>
<td>vbProperCase</td>
<td>3</td>
<td>Convierte a mayúscula la primera letra de cada palabra de la cadena.</td>
</tr>
<tr>
<td>vbWide*</td>
<td>4*</td>
<td>Convierte los caracteres cortos (un único byte) de la cadena a caracteres anchos (doble byte).</td>
</tr>
<tr>
<td>vbNarrow*</td>
<td>8*</td>
<td>Convierte los caracteres anchos (doble byte) de la cadena a caracteres cortos (un único byte).</td>
</tr>
<tr>
<td>vbKatakana**</td>
<td>16**</td>
<td>Convierte los caracteres Hiragana de la cadena a caracteres Katakana.</td>
</tr>
<tr>
<td>vbHiragana**</td>
<td>32**</td>
<td>Convierte los caracteres Katakana de la cadena a caracteres Hiragana.</td>
</tr>
<tr>
<td>vbUnicode</td>
<td>64</td>
<td>Convierte la cadena a Unicode utilizando la página de códigos predeterminada del sistema.</td>
</tr>
<tr>
<td>vbFromUnicode</td>
<td>128</td>
<td>Convierte la cadena de Unicode a la página de códigos predeterminada del sistema.</td>
</tr>
</tbody>
</table>

*Se aplica a las configuraciones regionales del Lejano Oriente.*
**Se aplica solamente a Japón.**

Nota Estas constantes están especificadas por Visual Basic for Applications. Como resultado, se pueden utilizar en cualquier parte de su código en lugar de los valores reales. La mayoría se pueden combinar, por ejemplo, vbUpperCase + vbWide, excepto cuando se excluyen mutuamente, por ejemplo, vbUnicode + vbFromUnicode. Las constantes vbWide, vbNarrow, vbKatakana y vbHiragana originan errores en tiempo de ejecución cuando utilizan configuraciones regionales donde no se aplican.

Los siguientes son separadores de palabras válidos para mayúsculas/minúsculas: Null (Chr$(0)), tabulador horizontal (Chr$(9)), avance de línea (Chr$(10)), tabulador vertical (Chr$(11)), avance de página (Chr$(12)), retorno de carro (Chr$(13)), espacio (SBCS) (Chr$(32)). El valor real de espacio varía por el país para DBCS.

Comentarios
Cuando se convierte una cadena de matriz Byte a formato ANSI, se debe utilizar la función StrConv. Cuando se convierte una matriz en formato Unicode, utilice una instrucción de asignación.

**Ejemplo**

### String (Función)

Devuelve un tipo Variant (String) que contiene una cadena de caracteres que se repite de la longitud especificada.

**Sintaxis**

```
String(number, character)
```

La sintaxis de la función String tiene estos argumentos con nombre:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>number</td>
<td>Requerido; un tipo Long. Longitud de la cadena devuelta. Si número contiene Null, se devuelve Null.</td>
</tr>
<tr>
<td>character</td>
<td>Requerido; un tipo Variant. Código de carácter que especifica el carácter o la expresión de cadena cuyo primer carácter se utiliza para construir la cadena devuelta. Si character contiene Null, se devuelve Null.</td>
</tr>
</tbody>
</table>

**Comentarios**

Si se especifica para character un número mayor que 255, String convierte el número en un código de carácter válido utilizando la fórmula:

```
character Mod 256
```

**Ejemplo**

Dim MiCadena

```
MiCadena = String(5, "*") ' Devuelve "*****".
MiCadena = String(5, 42) ' Devuelve "*****".
MiCadena = String(10, "ABC") ' Devuelve "AAAAAAAAAAA".
```
**StrReverse (Función)**

Descripción
Devuelve una cadena en la que se invierte el orden de carácter de una cadena especificada.

Sintaxis

```
StrReverse(cadena1)
```

El argumento cadena1 es la cadena cuyos caracteres se invierten. Si cadena1 es una cadena de longitud cero (""), se devuelve una cadena de longitud cero. Si es Null, se produce un error.

**Tan (Función)**

Devuelve un tipo Double que especifica la tangente de un ángulo.

Sintaxis

```
Tan(número)
```

El argumento número es un tipo Double o cualquier expresión numérica válida que expresa un ángulo en radianes.

Comentarios
La función Tan toma un ángulo y devuelve la razón de dos lados de un triángulo rectángulo. La razón es la longitud del lado opuesto al ángulo, dividida por la longitud del lado adyacente al ángulo.

Para convertir grados en radianes, multiplique los grados por pi/180. Para convertir radianes en grados, multiplique los radianes por 180/pi.

**Ejemplo**

```
Dim MiÁngulo, MyCotangente
MiÁngulo = 1.3  ' Define el ángulo en radianes.
MyCotangente = 1 / Tan(MiÁngulo)  ' Calcula la cotangente.
```

**Time (Función)**

Devuelve un valor de tipo Variant (Date) indicando la hora actual del sistema.

Sintaxis

```
Time
```

Comentarios
Use la instrucción Time para establecer la hora del sistema.

**Ejemplo**

```
Dim MiTiempo
MiTiempo = Time  ' Devuelve la hora del sistema actual.
```
Timer (Función)
Devuelve un tipo Single que representa el número de segundos transcurridos desde la medianoche.

Sintaxis

Timer

Comentarios
En Microsoft Windows la función Timer devuelve partes fraccionales de un segundo.

Ejemplo
En este ejemplo se utiliza la función Timer para detener la aplicación. El ejemplo también utiliza DoEvents para pasar el control a otros procesos durante la pausa.

Dim TiempoPausa, Inicio, Final, TiempoTotal
If (MsgBox("Presione Sí para parar 5 segundos", 4)) = vbYes Then
    TiempoPausa = 5   ' Asigna hora de inicio.
    Inicio = Timer   ' Establece la hora de inicio.
    Do While Timer < Inicio + TiempoPausa
        DoEvents   ' Cambia a otros procesos.
    Loop
    Final = Time   ' Asigna hora de finalización.
    TiempoTotal = Final - Inicio   ' Calcula tiempo total.
    MsgBox "Detenida durante " & TiempoTotal & " segundos"
Else
    End
End If

TimeSerial (Función)
Devuelve un tipo Variant (Date) que contiene la hora actual para una hora, minuto y segundo específicos.

Sintaxis

TimeSerial(hora, minuto, segundo)

La sintaxis de la función TimeSerial consta de los siguientes argumentos con nombre:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>hora</td>
<td>Requerido; Variant (Integer). Número entre 0 (12:00 a.m.) y 23 (11:00 p.m.), inclusive o unaexpresión numérica.</td>
</tr>
<tr>
<td>minuto</td>
<td>Requerido; Variant (Integer). Cualquier expresión numérica.</td>
</tr>
<tr>
<td>segundo</td>
<td>Requerido; Variant (Integer). Cualquier expresión numérica.</td>
</tr>
</tbody>
</table>

Comentarios
Para especificar una hora, como 11:59:59, el intervalo de números para cada argumento de TimeSerial debería estar en el intervalo normalmente aceptado para la unidad; es decir, 0–23 para horas y 0–59 para minutos y segundos. Sin embargo, puede especificar horas relativas para cada argumento, usando cualquier expresión numérica que represente algunas horas, minutos o segundos antes o después de cierta hora. El ejemplo siguiente usaexpresiones en lugar de números de hora absoluta. La función TimeSerial devuelve una hora que es 15 (-15) minutos menos que seis horas antes del mediodía (12 - 6), ó 5:45:00 a.m.

    TimeSerial(12 - 6, -15, 0)

Cuando algún argumento supera el intervalo normalmente aceptado para él, se incrementa hasta la siguiente unidad mayor. Por ejemplo, si especifica 75 minutos, se evalúa como una hora y 15 minutos. No obstante, si un solo argumento está fuera del intervalo -32.768 a 32.767 o si la hora especificada por los tres argumentos, directamente o mediante expresión, hace que la fecha quede fuera del intervalo aceptable de valores, se produce un error.

**Ejemplo**

Dim MiTiempo

MiTiempo = TimeSerial(16, 35, 17) ' MiTiempo contiene la hora

    ' correspondiente a 4:35:17 PM.

---

### TimeValue (Función)

Devuelve un tipo Variant (Date) que contiene la hora.

**Sintaxis**

    TimeValue(hora)

El argumento hora es normalmente unaexpresión de tipo cadena que representa una hora desde las 0:00:00 (12:00:00 a.m.) a las 23:59:59 (11:59:59 p.m.), inclusive. Sin embargo, hora también puede ser cualquierexpresión que pueda representar una hora en ese intervalo. Si hora contieneNull, la función devolverá Null.

**Comentarios**

Puede introducir horas válidas usando un reloj de 12 horas o de 24 horas. Por ejemplo, "2:24 p.m." y "14:24" son argumentos válidos de hora.

Si el argumento hora contiene información de fecha, TimeValue no devuelve esta información. Sin embargo, si hora incluye información no válida de fecha, ocurrirá un

**Ejemplo**

Dim MiHora

MiHora = TimeValue("4:35:17 PM") ' Devuelve la hora.

---

### TypeName (Función)

Devuelve una cadena (String) que proporciona información acerca de unvariable.

**Sintaxis**

    TypeName(nombrevariable)
El argumento `nombrevariable` requerido, es un tipo de datos `Variant` que contiene cualquier variable excepto una variable de `tipo` definido por el usuario.

Comentarios

La cadena devuelta por la función `TypeName` puede ser cualquiera de las siguientes:

<table>
<thead>
<tr>
<th>Cadena devuelta</th>
<th>La variable contiene</th>
</tr>
</thead>
<tbody>
<tr>
<td>tipo de objeto</td>
<td>Un objeto cuyo tipo es <code>tipoobjeto</code></td>
</tr>
<tr>
<td>Byte</td>
<td>Un valor byte</td>
</tr>
<tr>
<td>Integer</td>
<td>Un entero</td>
</tr>
<tr>
<td>Long</td>
<td>Un entero largo</td>
</tr>
<tr>
<td>Single</td>
<td>Un número de punto flotante de precisión simple</td>
</tr>
<tr>
<td>Double</td>
<td>Un número de punto flotante de precisión doble</td>
</tr>
<tr>
<td>Currency</td>
<td>Un valor de moneda</td>
</tr>
<tr>
<td>Decimal</td>
<td>Un valor decimal</td>
</tr>
<tr>
<td>Date</td>
<td>Un valor fecha</td>
</tr>
<tr>
<td>String</td>
<td>Una cadena</td>
</tr>
<tr>
<td>Boolean</td>
<td>Un valor booleanano</td>
</tr>
<tr>
<td>Error</td>
<td>Un valor de error</td>
</tr>
<tr>
<td>Empty</td>
<td>No inicializado</td>
</tr>
<tr>
<td>Null</td>
<td>Datos no válidos</td>
</tr>
<tr>
<td>Object</td>
<td>Un objeto</td>
</tr>
<tr>
<td>Desconocido</td>
<td>Un objeto cuyo tipo es desconocido</td>
</tr>
<tr>
<td>Nothing</td>
<td>Una variable de que no hace referencia a un objeto</td>
</tr>
</tbody>
</table>

Si `nombrevariable` es una matriz, la cadena devuelta puede ser cualquiera de las posibles (o `Variant`) con un paréntesis vacío añadido. Por ejemplo, si `nombrevariable` es una matriz de números enteros, la función `TypeName` devuelve "Integer()".

**Ejemplo**

Declara variables.

```vba
Dim VarNull, MiTipo, VarCad As String, VarEnt As Integer, VarMon As Currency
Dim VarMatriz (1 To 5) As Integer

VarNull = Null  ' Asigna un valor Null.
MiTipo = TypeName(VarCad)  ' Devuelve "String".
MiTipo = TypeName(VarEnt)  ' Devuelve "Integer".
MiTipo = TypeName(VarMon)  ' Devuelve "Currency".
MiTipo = TypeName(VarNull)  ' Devuelve "Null".
MiTipo = TypeName(VarMatriz)  ' Devuelve "Integer()".
```
**UCase (Función)**

Devuelve un tipo Variant (String) que contiene una cadena especificada que se ha convertido a mayúsculas.

**Sintaxis**

\[
\text{UCase(cadena)}
\]

El argumento `cadena` necesario es cualquier expresión de cadena válida. Si `cadena` contiene `Null`, se devuelve `Null`.

**Comentarios**

Sólo se convierten a mayúsculas las letras minúsculas. Las letras mayúsculas o los caracteres que no son letras no sufren cambios.

**Ejemplo**

Dim LowerCase, UpperCase

LowerCase = "Hola Mundo 1234"  ' Cadena a convertir.

UpperCase = UCase(LowerCase)  ' Devuelve "HOLA MUNDO 1234".

---

**Val (Función)**

Devuelve los números contenidos en una cadena como un valor numérico del tipo adecuado.

**Sintaxis**

\[
\text{Val(cadena)}
\]

El argumento obligatorio `cadena` es cualquier expresión de cadena válida.

**Comentarios**

La función `Val` deja de leer la cadena en el primer carácter que no puede reconocer como parte de un número. Los símbolos y caracteres que se consideran a menudo parte de valores numéricos, como signos de moneda y comas, no se reconocen. Sin embargo, la función reconoce los prefijos de base &O (para octal) y &H (para hexadecimal). Los espacios en blanco, los tabuladores y los avances de línea se eliminan del argumento.

Lo siguiente devuelve el valor 1615198:

\[
\text{Val(" 1615 198 Calle N.E.")}
\]

En el código que se muestra a continuación, `Val` devuelve el valor decimal -1 correspondiente al valor hexadecimal entre paréntesis:

\[
\text{Val("&HFFFF")}
\]

Nota. La función `Val` sólo reconoce el punto (.) como separador decimal válido. Cuando se utilizan separadores decimales diferentes, como en aplicaciones internacionales, debe utilizar `CDbl` para convertir una cadena a un número.

**Ejemplo**

Dim MiValor

MiValor = Val("2457")  ' Devuelve 2457.
MiValor = Val("2 45 7") ' Devuelve 2457.
MiValor = Val("24 y 57") ' Devuelve 24.

**VarType (Función)**

Devuelve un entero (Integer) que indica el subtipo de una variable.

**Sintaxis**

`VarType(nombrevariable)`

El argumento `nombrevariable` requerido, es un tipoVariant que contiene cualquier variable excepto una variable de un tipo definido por el usuario.

**Valores devueltos**

<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>vbEmpty</td>
<td>0</td>
<td>Empty (no inicializado)</td>
</tr>
<tr>
<td>vbNull</td>
<td>1</td>
<td>Null (datos no válidos)</td>
</tr>
<tr>
<td>vbInteger</td>
<td>2</td>
<td>Entero</td>
</tr>
<tr>
<td>vbLong</td>
<td>3</td>
<td>Entero largo</td>
</tr>
<tr>
<td>vbSingle</td>
<td>4</td>
<td>Un número de coma flotante de precisión simple</td>
</tr>
<tr>
<td>vbDouble</td>
<td>5</td>
<td>Un número de coma flotante de precisión doble</td>
</tr>
<tr>
<td>vbCurrency</td>
<td>6</td>
<td>Valor de moneda</td>
</tr>
<tr>
<td>vbDate</td>
<td>7</td>
<td>Valor de fecha</td>
</tr>
<tr>
<td>vbString</td>
<td>8</td>
<td>Cadena</td>
</tr>
<tr>
<td>vbObject</td>
<td>9</td>
<td>Objeto</td>
</tr>
<tr>
<td>vbError</td>
<td>10</td>
<td>Valor de error</td>
</tr>
<tr>
<td>vbBoolean</td>
<td>11</td>
<td>Valor booleano</td>
</tr>
<tr>
<td>vbVariant</td>
<td>12</td>
<td>Variant (utilizada solamente con matrices de variantes)</td>
</tr>
<tr>
<td>vbDataObject</td>
<td>13</td>
<td>Un objeto de acceso a datos</td>
</tr>
<tr>
<td>vbDecimal</td>
<td>14</td>
<td>Valor decimal</td>
</tr>
<tr>
<td>vbByte</td>
<td>17</td>
<td>Valor de byte</td>
</tr>
<tr>
<td>vbUserDefinedType</td>
<td>36</td>
<td>Variantes que contienen tipos definidos por el usuario</td>
</tr>
<tr>
<td>vbArray</td>
<td>8192</td>
<td>Matriz</td>
</tr>
</tbody>
</table>

Nota: Estas constantes están especificadas por Visual Basic para aplicaciones. Los nombres se puede utilizar en cualquier posición en su código en lugar de los valores actuales.

**Comentarios**

La función VarType nunca devuelve por sí misma el valor para vbArray. Siempre se agrega a algún otro valor para indicar una matriz de un tipo particular. La constante vbVariant solamente se devuelve junto con vbArray para indicar que el argumento de la función VarType es una matriz del tipo Variant. Por ejemplo, el valor devuelto por una matriz de enteros se calcula como vbInteger + vbArray, o 8194. Si un objeto tiene una propiedad predeterminada, VarType (objeto) devuelve el tipo de la propiedad predeterminada del objeto.

**Ejemplo**
Dim VarEnt, VarCad, VarFecha, MiPrueba
' Inicializa las variables.
VarEnt = 459: VarCad = "Hola a todos ": VarFecha = #2/12/69#
MiPrueba = VarType(VarEnt)   ' Devuelve 2.
MiPrueba = VarType(VarFecha)   ' Devuelve 7.
MiPrueba = VarType(VarCad)   ' Devuelve 8.

**Weekday (Función)**

Devuelve un valor de tipo Variant (Integer) que contiene un número entero que representa el día de la semana.

Sintaxis

```vba
Weekday(fecha, [primerdías]semana)
```

La sintaxis de la función Weekday consta de los siguientes argumentos con nombre:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>fecha</td>
<td>Requerido. Variant, expresión numérica, expresión de tipo cadena o cualquier combinación que puede representar una fecha. Si fecha contiene Null, la función devolverá Null.</td>
</tr>
<tr>
<td>primerdías semana</td>
<td>Opcional. Una constante que especifica el primer día de la semana. Si no se especifica, se asume el valor vbSunday.</td>
</tr>
</tbody>
</table>

**Valores**

El argumento primerdías semana tiene estos valores:

<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>vbUseSystem</td>
<td>0</td>
<td>Utilice configuración de la API de NLS.</td>
</tr>
<tr>
<td>vbSunday</td>
<td>1</td>
<td>Domingo (predeterminado)</td>
</tr>
<tr>
<td>vbMonday</td>
<td>2</td>
<td>Lunes</td>
</tr>
<tr>
<td>vbTuesday</td>
<td>3</td>
<td>Martes</td>
</tr>
<tr>
<td>vbWednesday</td>
<td>4</td>
<td>Miércoles</td>
</tr>
<tr>
<td>vbThursday</td>
<td>5</td>
<td>Jueves</td>
</tr>
<tr>
<td>vbFriday</td>
<td>6</td>
<td>Viernes</td>
</tr>
<tr>
<td>vbSaturday</td>
<td>7</td>
<td>Sábado</td>
</tr>
</tbody>
</table>

**Valores devueltos**

La función Weekday puede devolver cualquiera de estos valores:

<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>vbSunday</td>
<td>1</td>
<td>Domingo</td>
</tr>
<tr>
<td>vbMonday</td>
<td>2</td>
<td>Lunes</td>
</tr>
<tr>
<td>vbTuesday</td>
<td>3</td>
<td>Martes</td>
</tr>
<tr>
<td>vbWednesday</td>
<td>4</td>
<td>Miércoles</td>
</tr>
<tr>
<td>vbThursday</td>
<td>5</td>
<td>Jueves</td>
</tr>
</tbody>
</table>
Ejemplo
Dim MiFecha, MiDíaSemana
MiFecha = #12 febrero 1969# ; Asigna una fecha.
MiDíaSemana = Weekday(MiFecha) ; MiDíaSemana contiene 4 ya que
; MiFecha es un miércoles.

WeekdayName (Función)
Devuelve una cadena que indica el día de la semana especificado.

Sintaxis

WeekdayName(díaDeLaSemana, abreviar, primerDíaDeLaSemana)

La sintaxis de la función WeekdayName consta de las siguientes partes:

<table>
<thead>
<tr>
<th>Parte</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>díaDeLaSemana</td>
<td>Requerido. La designación numérica para el día de la semana. El valor numérico de cada día depende de la configuración del valor primerDíaDeLaSemana.</td>
</tr>
<tr>
<td>abreviar</td>
<td>Opcional. Un valor Boolean que indica si se abrevia el nombre del día de la semana. Si se omite, el valor predeterminado es False, que significa que el nombre del día de la semana no se abrevia.</td>
</tr>
<tr>
<td>primerDíaDeLaSemana</td>
<td>Opcional. Un valor numérico que indica el primer día de la semana. Vea la sección Valores para consultar los valores posibles.</td>
</tr>
</tbody>
</table>

Valores
El argumento primerDíaDeLaSemana puede tener los valores siguientes:

<table>
<thead>
<tr>
<th>Constante</th>
<th>Valor</th>
<th>Descripción</th>
</tr>
</thead>
<tbody>
<tr>
<td>vbUseSystem</td>
<td>0</td>
<td>Utilizar la configuración de la API National Language Support (NLS).</td>
</tr>
<tr>
<td>vbSunday</td>
<td>1</td>
<td>Domingo (predeterminado).</td>
</tr>
<tr>
<td>vbMonday</td>
<td>2</td>
<td>Lunes.</td>
</tr>
<tr>
<td>vbTuesday</td>
<td>3</td>
<td>Martes.</td>
</tr>
<tr>
<td>vbWednesday</td>
<td>4</td>
<td>Miércoles.</td>
</tr>
<tr>
<td>vbThursday</td>
<td>5</td>
<td>Jueves.</td>
</tr>
<tr>
<td>vbFriday</td>
<td>6</td>
<td>Viernes.</td>
</tr>
<tr>
<td>vbSaturday</td>
<td>7</td>
<td>Sábado.</td>
</tr>
</tbody>
</table>

.Year (Función)
Devuelve un valor de tipo Variant (Integer) que contiene un número entero que representa el año.

Sintaxis
**Year(fecha)**

El argumento fecha es cualquier valor de tipo Variant, expresión numérica, expresión de tipo cadena o cualquier combinación que puede representar una fecha. Si fecha contiene Null, la función devolverá Null.

**Ejemplo**

Dim MiFecha, MiAño

MiFecha = #12 febrero 1969# Asigna una fecha.

MiAño = Year(MiFecha) ' MiAño contiene 1969.