Práctica

6

GESTIÓN Y UTILIZACIÓN DE REDES LOCALES

Curso 2001/2002

Introducción a Visual Basic

Introducción

Entre las razones por las cuales la programación en el entorno Windows (3.1, 9x, NT) se diferencia de la programación en otros entornos está la dificultad añadida que supone para el programador el interfaz gráfico de la aplicación a desarrollar. Sin embargo, con la aparición de herramientas de programación específicas para el desarrollo de aplicaciones en entornos Windows se facilita enormemente la programación, proporcionándose las estructuras necesarias para el desarrollo de aplicaciones (menús desplegables, botones de comando, cuadros de opción, etc.). De esta manera, la creación de completas interfaces de usuario es más sencilla y eficaz, además de permitirse el desarrollo rápido de prototipos para su validación por el usuario.

Estas herramientas de programación se caracterizan, principalmente, porque el programador desarrolla su aplicación básicamente a partir del diseño de una interfaz: crea una ventana, introduce en ella diversos objetos que simbolizan datos o acciones que hay que llevar a cabo, establece propiedades de esos objetos y añade código donde es necesario. Una de estas herramientas de programación está basada en el lenguaje **Visual Basic**.

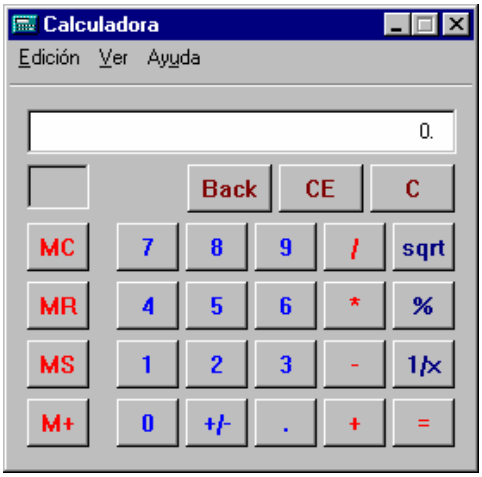
En esta práctica se pretende hacer una sencilla introducción a Visual Basic, de forma que en el resto de prácticas del curso podamos centrarnos en diversos aspectos relacionados con TCP/IP, y la programación de sencillas aplicaciones que utilizan este tipo de protocolos.

Existen otros lenguajes para la programación en Windows, como **C**, **C++**, **Visual C++**, **Pascal**, **Delphi**, etc. Cada uno de ellos tiene sus ventajas e inconvenientes y, en general, se trata de herramientas basadas en algún lenguaje de programación más o menos conocido. Hemos escogido Visual Basic por ser, a nuestro juicio, el más sencillo de usar, aunque no cabe duda que esta afirmación no será compartida por todos los lectores, y somos conscientes de que esta elección puede no satisfacer los gustos de personas familiarizadas con otros lenguajes de programación. Por otra parte, Visual Basic es empleado por numerosas empresas, además de ser muy similar al lenguaje de programación llamado **Access Basic**, utilizado en el entorno de bases de datos de Microsoft Access.

Visual Basic vs. programación clásica

En la programación tradicional, la ejecución de un programa comienza en la primera línea de código y sigue con la ejecución hasta la última línea, desviándose para ejecutar procedimientos y funciones a medida que son invocados. De esta forma, el programa básicamente se dedica a recoger la información que necesita y realiza los cálculos y operaciones para proporcionar, generalmente al final, un resultado. La aplicación misma controla las secciones de código que han de ejecutarse.

Sin embargo, una aplicación Windows suele estar fuertemente condicionada al empleo de un interfaz gráfico que permite al usuario la introducción de información, y que reacciona a diferentes eventos (o sucesos) que produce el usuario, como por ejemplo la pulsación de una tecla o un click con el ratón en un punto determinado. Las aplicaciones se centran en un funcionamiento guiado por los eventos que se van produciendo.



Como se puede ver, de una filosofía de programa secuencial se pasa a otra dirigida por eventos. De este modo nuestra aplicación básicamente especificará cómo reaccionar a cada uno de los diferentes eventos. Por ejemplo, si nuestro programa es una calculadora como la de la figura, compuesta por una serie de botones numéricos y otros de operaciones, el programa indicará cómo responder a la pulsación de cada uno de los botones.

Esta es la esencia de los interfaces gráficos de usuario y de la programación orientada a eventos: el usuario realiza acciones y la aplicación responde. Por todo lo anterior vemos que el enfoque de la programación orientada a eventos cambia radicalmente respecto a lo que hemos aprendido hasta ahora en otras asignaturas. Ahora bien, ante cada suceso nuestra aplicación ejecutará un código secuencial similar a la programación que ya conocemos.

Visual Basic

Visual Basic proporciona todos los elementos que se utilizan en la definición de interfaces para aplicaciones que usan ventanas. Estos elementos reciben el nombre de controles en la terminología que emplea Visual Basic. Ejemplos de controles son los menús desplegables, los botones de comando, los cuadros de opción, etc. Cada control en Visual Basic se caracteriza por sus eventos, métodos y propiedades:

- **Eventos:** Son sucesos reconocidos por un control (como hacer click sobre un botón o pulsar una tecla en una ventana de texto). Si se quiere que el control que ha recibido el evento ejecute alguna acción como respuesta, el programador deberá escribir el código asociado a esa acción. Los eventos los puede provocar el usuario (por ejemplo al pulsar una tecla), aunque también los puede provocar el sistema o incluso la misma aplicación.
- **Métodos:** Son funciones que operan sobre un control. Cada control tiene una lista de métodos asociados. Por ejemplo si a un control se le aplica el método `move`, el resultado será cambiar la posición del control dentro del formulario. El código para los métodos ya está escrito y por tanto podemos invocarlos sin necesidad de programarlos. Sin embargo, no podemos cambiar lo que hacen.
- **Propiedades:** Especifican los valores para ciertas características de los controles. Las propiedades pueden hacer referencia tanto a características relativas a la apariencia del objeto (tamaño, color, etc.) como a su comportamiento (activado/desactivado, etc.).

De acuerdo a lo mencionado anteriormente referente a la programación orientada a eventos, en la que el usuario realiza acciones y la aplicación responde, el funcionamiento típico de una aplicación Visual Basic es el siguiente:

- La aplicación comienza y muestra al usuario una ventana (también llamada formulario) conteniendo controles.
- El formulario o uno de los controles que contiene recibe un evento.
- Si el procedimiento de evento correspondiente tiene código asociado (es decir, no está vacío), se ejecuta el código.
- La aplicación se pone a la espera de otro evento.

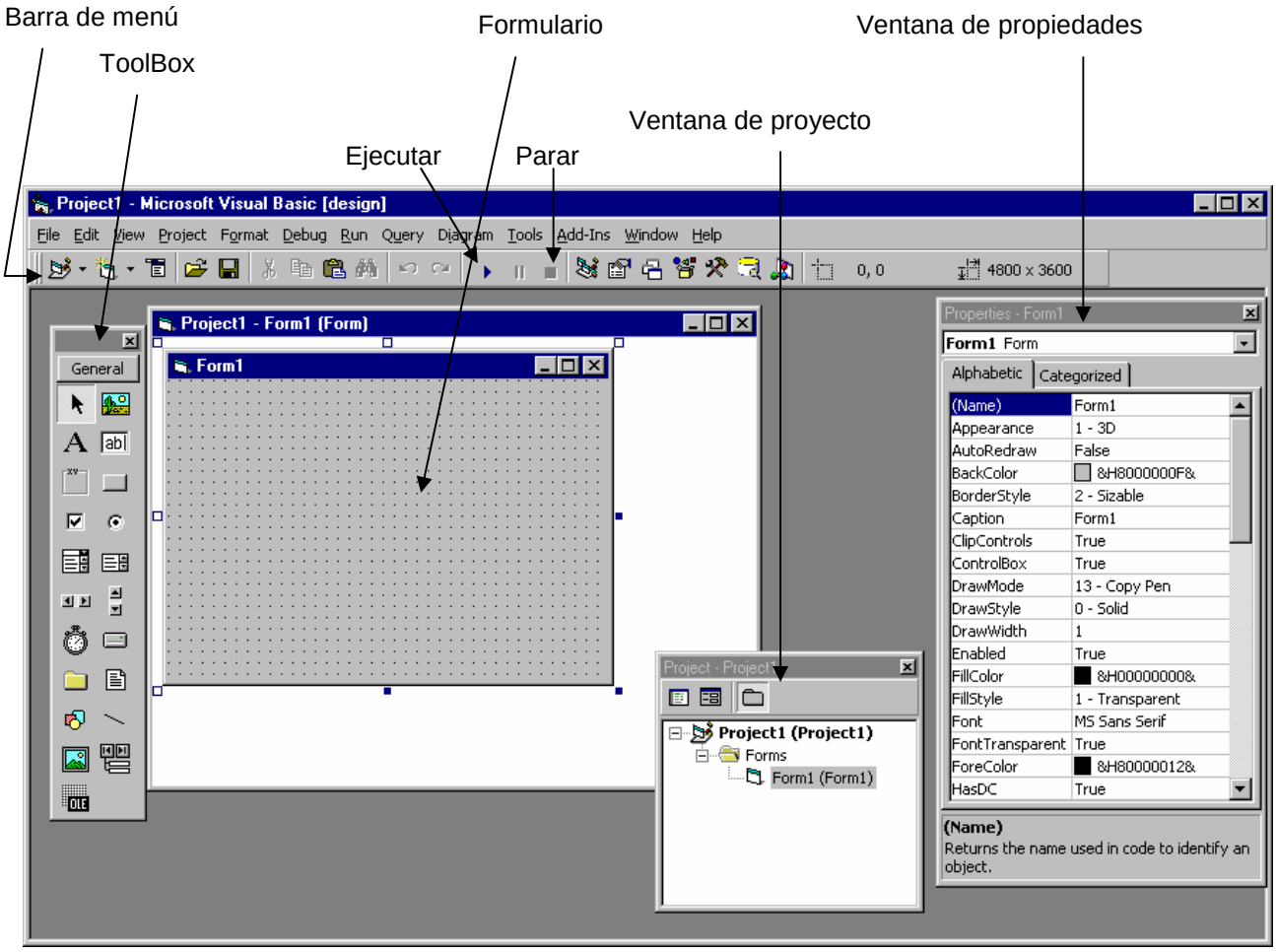
El entorno de programación Visual Basic

Visual Basic es una herramienta de programación de Microsoft. Su aspecto es algo distinto a otras aplicaciones como pueda ser una hoja de cálculo o un procesador de texto, ya que consta no sólo de una única ventana como es común, sino de múltiples ventanas (o marcos) conteniendo cada una diferentes elementos.

En la figura de la página siguiente se ven los principales elementos del entorno de desarrollo, que se describen a continuación.

La Barra de menú

El primero de estos elementos es la barra de menú, que permite realizar las operaciones de cargar y salvar proyectos, ejecutar y detener el programa, configurar multitud de opciones, etc.



La ToolBox

La ToolBox contiene los controles que Visual Basic proporciona para crear la interfaz de usuario de una aplicación. Los controles son objetos que podemos incorporar en una ventana del interfaz, como por ejemplo botones, cuadros de texto, las listas con o sin scroll, etc.

El procedimiento a seguir para insertar un control en un formulario es:

- Hacer click en la ToolBox sobre el control que se desea insertar.
- Mover el puntero del ratón al formulario. El puntero tomará la forma de una cruz.
- Posicionar el puntero sobre el lugar en el que se desea ubicar la esquina superior izquierda del control.
- Pulsar el botón izquierdo del ratón y, manteniéndolo pulsado, mover el ratón hasta que el control tenga el tamaño deseado.

También se puede insertar un control haciendo doble click en la ToolBox sobre el control que se desea insertar, de manera que este se situará en el centro del formulario.

Por otra parte, para ver la ayuda, donde se describen las propiedades, métodos y eventos de cada control, solamente hay que pulsar F1, teniendo el control seleccionado.

Los controles más habituales son:

Command: Este control nos permite insertar botones.

Label: Permite incluir etiquetas (no se pueden cambiar durante la ejecución).

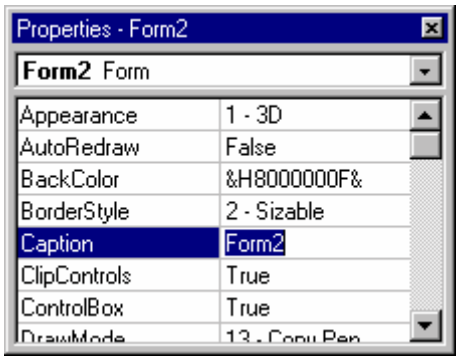
Text: Campo de texto (su contenido se puede cambiar durante la ejecución).

Listbox: Lista de elementos

Formulario

El formulario en Visual Basic es el punto central de cualquier aplicación, pudiendo ésta constar de uno o más formularios, según las necesidades. Un formulario es una ventana Windows en la que colocamos los controles necesarios para crear nuestra interfaz con el usuario de la aplicación.

Ventana de Propiedades



En la ventana de propiedades se establecen los valores para las características de los controles y de los formularios. Cada control viene caracterizado por su lista de propiedades. La columna de la izquierda muestra las propiedades del control seleccionado, y la columna de la derecha el valor establecido actualmente para esas propiedades.

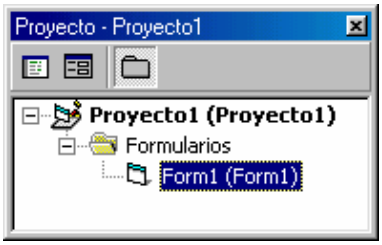
Algunas propiedades sólo pueden ser establecidas o modificadas en tiempo de diseño, mientras que otras sólo permiten el acceso en tiempo de ejecución. Por último, a otras propiedades se les puede cambiar su valor en cualquier momento.

Se puede acceder a la **Ventana de propiedades** de un control o formulario seleccionando el control o formulario y pulsando **F4**, o bien mediante el comando **Ventana Propiedades** del menú **Ver**. También se puede pulsar el icono correspondiente de la Barra de menú.

Ventana de Proyecto

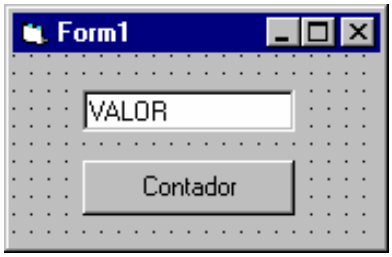
Un proyecto es la colección de ficheros que se usan para construir una aplicación. Un proyecto consta, como mínimo, de:

- Un fichero por cada uno de los formularios del proyecto. Tienen extensión **.FRM**
- Un fichero que contiene una lista con todos los ficheros del proyecto, además de la información de entorno. Tendrá extensión **.VBP**



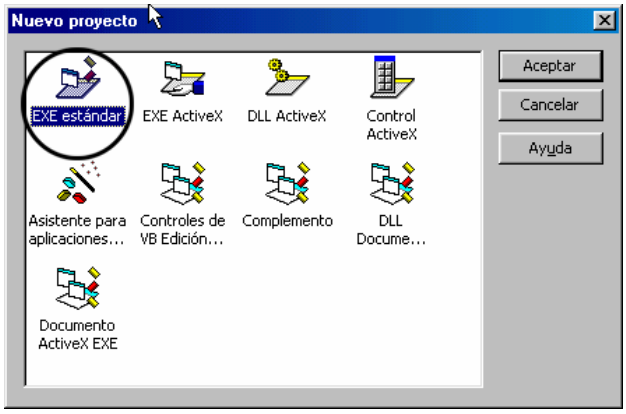
Para visualizar la ventana de proyectos, en el menú **Ver** seleccionar **Explorador de Proyectos**. Por otra parte, en el menú **Proyecto** encontramos las distintas operaciones de manejo de proyectos (**Agregar Formulario**, etc.).

Primeros pasos en Visual Basic



El siguiente paso, una vez revisado el entorno, es realizar una serie de ejercicios introductorios para familiarizarnos con la construcción de pequeñas y sencillas aplicaciones en Visual Basic. La primera aplicación que vamos a realizar va a ser un programa que cuente las veces que se hace click sobre un determinado botón. Esta aplicación tendrá un aspecto similar a la figura de la izquierda.

El primer paso es diseñar la presentación de nuestro programa. Para ello empezamos un nuevo proyecto (menú **Archivo**). Aparecerá la ventana de la derecha, preguntando qué tipo de proyecto queremos hacer. Escogeremos el tipo "**EXE estándar**".

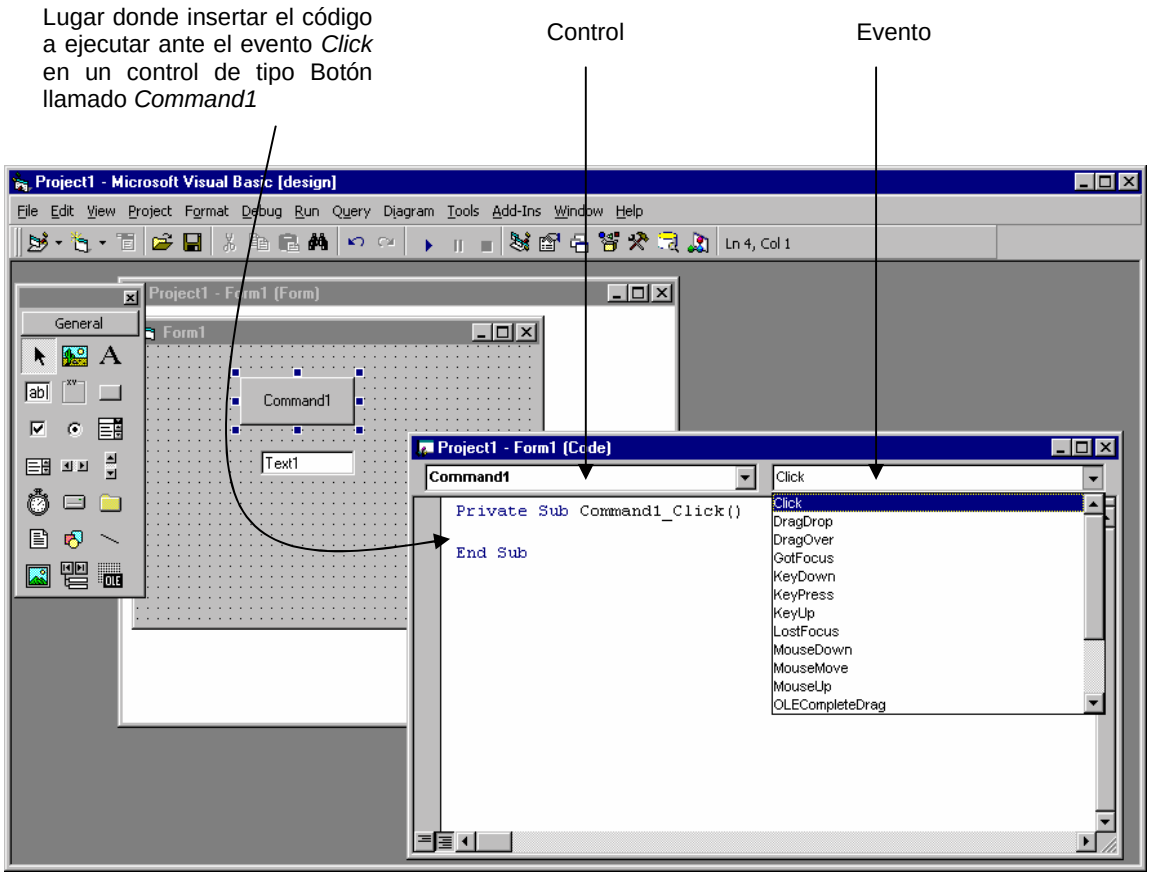


Posteriormente incorporamos a la hoja vacía (Form1) un campo de texto (cogiéndolo de la toolbox o ventana de controles). A continuación añadimos el botón (por el mismo procedimiento) y cambiamos la propiedad **text** del campo de texto a "VALOR" y la propiedad **caption** del botón a "Contador". Finalmente introducimos el código de la aplicación, que se detalla a continuación:

```
Private Sub Form_Load()
    Text1.Text = 0
End Sub

Private Sub Command1_Click()
    Text1.Text = Text1.Text + 1
End Sub
```

En la siguiente figura se muestra dónde se debe insertar el código.



Como vemos, la variable que utilizaremos en el contador para guardar su valor es una de las propiedades del objeto Text1, que es el cuadro de texto. La acción asociada a la pulsación del botón es incrementar ese valor. La inicialización del contador se produce cuando se abre la ventana Form1, pues en ese momento se ejecuta el código asociado al procedimiento "**Form Load**" (que es el primer procedimiento de la aplicación que se ejecuta). Por otra parte, el programa terminará cuando pulsemos el aspa de cerrar ventana.

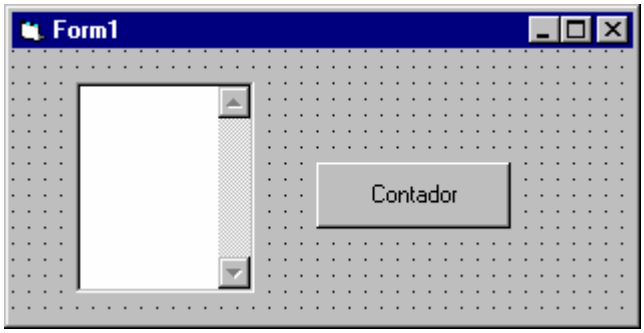
EJERCICIO: Modificar el programa anterior para que, además, se decremente en 10 unidades la cuenta del contador al realizar un doble click sobre el cuadro de texto.

EJERCICIO: Repetir el programa anterior empleando para visualizar el contador un control de tipo etiqueta (Label) en vez del cuadro de texto. En este caso la propiedad que almacena el valor de la cuenta debe ser "Caption", puesto que la propiedad "Text" no está disponible en las etiquetas.

Listas de elementos

El siguiente control que vamos a introducir nos va a permitir presentar y manejar listas de elementos. Si deseamos presentar una información larga, compuesta por una lista de diferentes items, la forma más cómoda de hacerlo es usando un control de tipo **ListBox**.

Comenzaremos con un programa contador, similar al anterior, que añade una línea a un cuadro de texto cada vez que pulsamos un botón.



El código asociado al evento click del botón será el siguiente:

```
Private Sub Contador_Click()
    valor = valor + 1
    List1.AddItem valor
End Sub
```

Hay que fijarse en un detalle y es que, a diferencia de los casos anteriores, el contenido de la variable **`valor`** no se asigna mediante un símbolo igual (como sucedería en el caso de que se tratara de una propiedad) sino que aparece separado por un espacio en blanco. Podemos decir que en este caso esta variable actúa como un parámetro del procedimiento **`AddItem`**, que no es una propiedad sino que es un método, común en aquellos objetos que están compuestos por una colección de ítems.

El resto del código del programa sería:

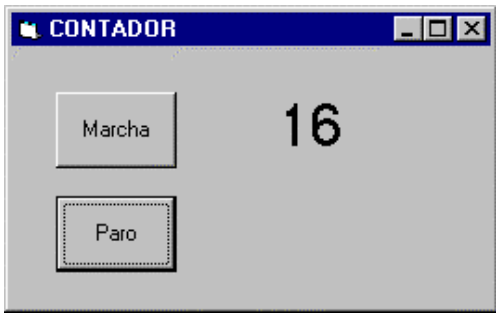
```
Dim valor As Integer
Private Sub Form_Load()
        valor = 0
End Sub
```

Como en el caso anterior, en el procedimiento asociado al evento Form_Load (que es el primero que se ejecuta) se inicializa la variable **`valor`**. Sin embargo, a diferencia del código del programa anterior, ahora aparece la línea "**`Dim valor As Integer`**". Esta línea declara la variable **`valor`** como global a todo el programa. Esta declaración debe situarse en la parte "General", al principio de todo el código.

CUESTIÓN: ¿Qué sucede si la declaración "**`Dim valor As Integer`**" se elimina del código?.

Temporizadores

En ocasiones necesitamos que un programa realice una tarea de forma repetitiva cada cierto tiempo. Esto lo podemos conseguir con el empleo de temporizadores. En Visual Basic disponemos de un control que automatiza la creación de temporizadores. En la siguiente figura tenemos un programa ejemplo que incorpora un temporizador.



El programa dispone de dos botones: uno para iniciar la cuenta y otro para detenerla. La frecuencia de conteo depende del valor de la propiedad **Interval** del temporizador. Una vez completado el intervalo prefijado, el temporizador provoca un evento **Timer**, en cuyo código asociado pondremos la tarea periódica que deseemos (en nuestro caso incrementar el valor de la etiqueta del contador).

Un detalle importante es que el control del temporizador no aparece cuando ejecutamos el programa, puesto que se trata de un objeto invisible. En algunos controles encontraremos una propiedad **Visible** que nos permitirá indicar si ese control es o no visible durante la ejecución. Como podemos comprobar, esa propiedad no existe para el control temporizador, que siempre es invisible en ejecución aunque aparece en el diseño del formulario en Visual Basic.

El código del programa anterior se detalla a continuación:

```
Private Sub Marcha_Click()
    Timer1.Interval = 100 'milisegundos
    Timer1.Enabled = True
End Sub

Private Sub Paro_Click()
    Timer1.Enabled = False
End Sub

Private Sub Timer1_Timer()
     contador.Caption = contador.Caption + 1
End Sub
```

El temporizador se ha establecido en **100 milisegundos**, con lo que la cuenta se produce a razón de diez unidades cada segundo. La propiedad **Enable** del temporizador

permite activar o desactivar el mismo. Si el temporizador es desactivado ya no podrá generar más sucesos **Timer**.

EJERCICIO: Construid un programa, similar al anterior, que incremente la cuenta dos veces por segundo.

Estructuras de control

En Visual Basic tenemos estructuras de control similares a las de otros lenguajes, como son el bucle **for**, o la instrucción condicional **if**. Vamos a comentar las más sencillas, pues las usaremos posteriormente en otras prácticas.

El bucle **For** es muy habitual. Su sintaxis es muy diferente a la del lenguaje C, como se muestra en el cuadro que sigue:

```
For variable = valor1 To valor2 [step incremento]
    ....
    ....
Next variable
```

La instrucción **for** anterior realiza el bucle tantas veces como sea necesario para que la **variable** alcance el **valor2** comenzando desde el **valor1**. El incremento por defecto es uno, excepto que indiquemos otro valor con la opción **step**. El bucle ejecuta todas las instrucciones que se escriban entre la palabra **for** y la palabra **next**.

La estructura condicional más típica es la orden **If_then_else,** que tiene dos formas:

```
'primera forma:
If condición Then acción


'segunda forma:
If condición Then
    acción1
    acción2
    ...
Else      'el Else es opcional
    acción3
    acción4
    ...
Endif
```

En la primera forma no existe **else**. Por este motivo la acción que se ejecuta tras el **Then** hay que ponerla justo detrás de éste, en la misma línea. Además, únicamente se puede poner una instrucción. Si necesitamos más instrucciones, habría que utilizar la segunda forma, que no tiene por qué tener necesariamente la parte del **else**. Hay que hacer notar que es necesaria la palabra **Endif** al final del **if**.

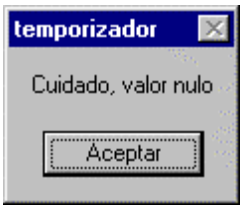
EJERCICIO: Construid un programa que cuente del 1 al 100. Cuando la cuenta pase por los valores 25, 50, y 75 debe añadir a una ListBox el número correspondiente, expresado con letras.

Cuadros de aviso

Para algunas aplicaciones es conveniente presentar una ventana que avise de una determinada contingencia. Este es el caso, por ejemplo, al final de una cuenta, o tras un error en los datos que introduce el usuario. Esto puede realizarse mediante una ventana que aparece superpuesta a la aplicación. Esta es la función de la orden **MsgBox**. La forma más simple de emplear esta orden es especificando tan sólo la cadena de caracteres a representar, como se muestra en el cuadro de la derecha.

```
MsgBox "Cuidado, valor nulo"
```

En este caso visualizará la cadena de caracteres en una ventana con un botón de "**Aceptar**".



EJERCICIO: Construid un programa que realice una temporización cuyo valor en segundos se introduce en un cuadro de texto. Al pulsar el botón de "comenzar" empieza a contar el tiempo y al finalizar la cuenta, el programa hace aparecer un cuadro de aviso.