Unidad Didáctica 8

Creación de cuadros de diálogo

Contenido

1. Introducción

Ya sabe algo más sobre el manejo de macros e interactuar con los distintos elementos del libro de trabajo. En el uso cotidiano de *Excel,* se ha podido comprobar que el programa muestra multitud de ventanas y cuadros de diálogo para realizar diversas funciones (por ejemplo abrir un archivo, cambiar el formato de fuente, imprimir, etc.).

Todos estos cuadros de diálogo se podrán utilizar mediante macros o a través de instrucciones en *Visual Basic.*

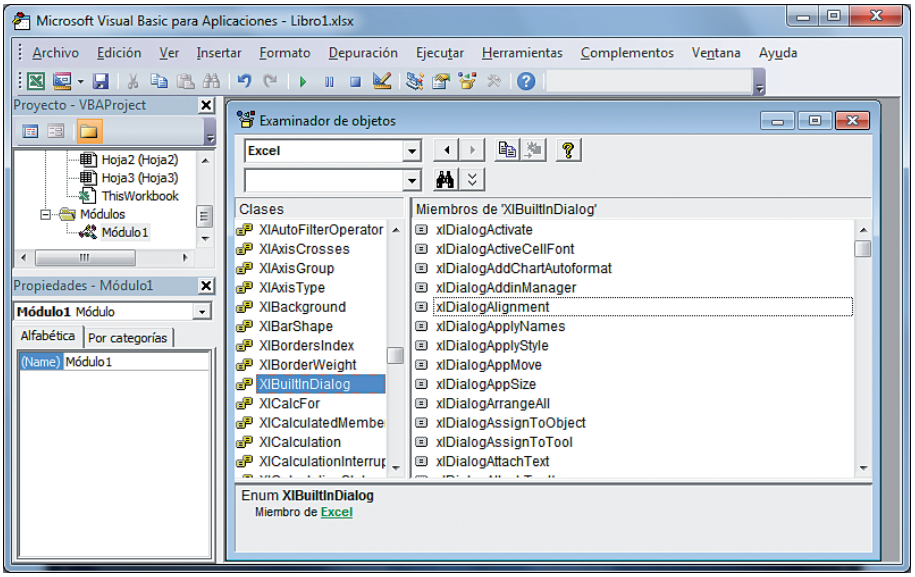
Aparte de los cuadros de diálogo, *Excel* ofrece la posibilidad de mostrar mensajes de advertencia, de información, de prohibición, etc., en pantalla.

También se podrán crear pequeños cuadros de diálogo en los que se soliciten datos al usuario para operar con ellos.

2. Utilización de los cuadros de diálogo prediseñados de Excel

El objeto que llama a los cuadros de diálogo prediseñados de *Excel* es el objeto **Application,** que dispone de una propiedad llamada **Dialogs,** que será la encargada de llamar a las distintas ventanas.

Para ver las instrucciones que abren los distintos cuadros de diálogo, se abrirá el **Examinador de objetos** de la ventana de *Visual Basic.* Para ello, se despliega el menú **Ver** y se selecciona esta opción o bien se pulsa la tecla [F2]. Cualquiera de estas acciones abrirá la siguiente ventana.



Cuadros de diálogo prediseñados

Para ver los miembros que abren las distintas ventanas, se debe seleccionar en la lista de la derecha la clase **XlBuiltInDialog,** que se podría traducir como "construyendo el cuadro de diálogo". Al seleccionar esta clase, aparecerá un listado en la zona de la izquierda con muchísimos elementos. Cada uno de estos elementos abrirá un cuadro de diálogo.

Nota

Cada miembro se corresponde con un cuadro de diálogo. Puede ser que algunos miembros abran el mismo cuadro de diálogo, pero en una pestaña distinta.

Se puede ver que la lista es bastante amplia. Algunos miembros de esta colección son:

- **xlDialogActiveCellFont:** abre la ventana **Fuentes.**
- **xlDialogAlignment:** abre la ventana **Formato de celdas** en la pestaña **Alineación.**
- **xlDialogApplyStyle:** abre la ventana **Estilo.**
- **xlDialogBorder:** abre la ventana **Formato de celdas** en la pestaña **Border.**
- **xlDialogOpen:** abre el cuadro de diálogo **Abrir.**

El método que se utilizará para ver el cuadro de diálogo será el método **Show.** Este método tiene una serie de parámetros para llamar a los cuadros de diálogo, pero no es frecuente utilizarlo, de hecho, quizá nunca se llegue a emplear.

Consejo

La mejor manera de ver qué abre cada miembro es probándolo directamente.

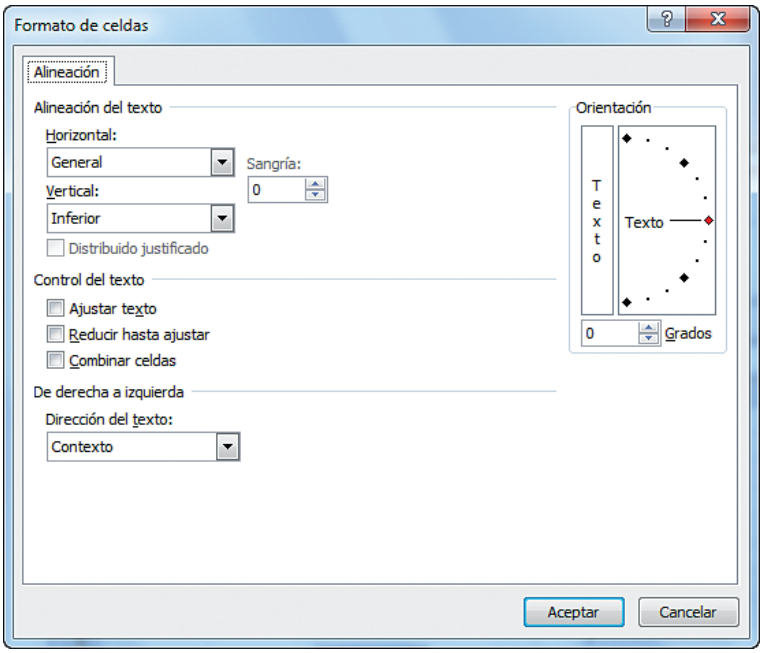
Un ejemplo para llamar a un cuadro de diálogo sería:

```
Application.Dialogs(xlDialogAlignment).Show
```

Esta instrucción, como ya se ha dicho, mostrará el cuadro de diálogo **Formato de celdas** con la ficha **Alineación.**

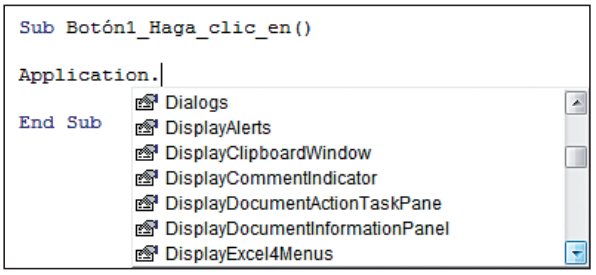
Nota

Al probar cuadros de diálogo, puede que algunos miembros den error. Esto puede ser debido a que necesiten algún parámetro o a que se deba realizar antes otra acción en la hoja de cálculo.

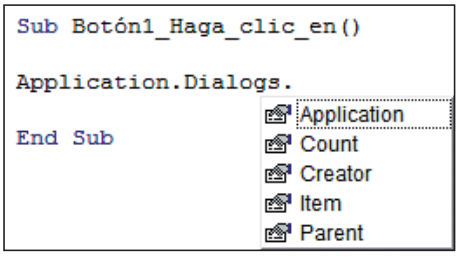


Alineación del Formato de celdas

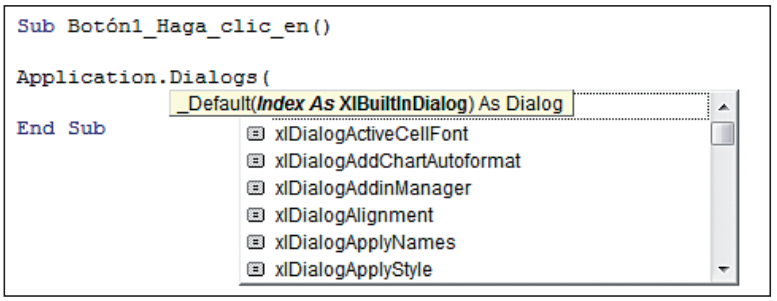
No será necesario saberse de memoria los miembros o las propiedades del objeto **Application,** ya que, a medida que se vaya escribiendo el código, irán apareciendo referencias. Para entenderlo mejor, obsérvese la siguiente secuencia de imágenes.



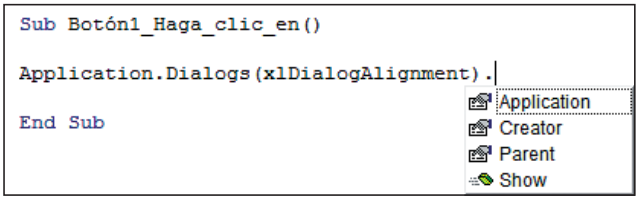
Referencias al objeto Application



Referencias al miembro Dialogs



Referencias de objetos del miembro Dialogs



Propiedades del miembro Dialogs

3. Cuadros de mensajes

Como ya se ha dicho anteriormente, *Excel* permite mostrar mensajes en pantalla para informar al usuario de un error o simplemente para preguntar si se quiere hacer una acción o no.

Para ello, se dispone de la función **MsgBox,** que muestra un mensaje en una ventana y puede mostrar distintas combinaciones de botones para que el usuario pulse uno. Cada botón devolverá un valor al programa.

La sintaxis de **MsgBox** es la siguiente:

```
MsgBox(Texto[, botones][, cabecera][, ficherodeAyuda, contexto])
```

Sus partes son:

- **Texto:** texto del mensaje.
- **Botones:** aquí se indicarán los botones que deben aparecer.
- **Cabecera:** al ser una ventana, se puede establecer el texto de la barra de título.
- **FicherodeAyuda:** esta parte es opcional y sirve para indicar un fichero de ayuda en el mensaje. Este fichero de ayuda sería como la ayuda del programa e irían todos los temas de ayuda integrados. Si se añade un fichero de ayuda, se deberá indicar un contexto.
- **Contexto:** al estar toda la ayuda integrada, con el contexto se indicará qué parte de la ayuda debe mostrar. Este parámetro es un número.

En la siguiente tabla, se pueden ver los valores que puede tener el argumento **Botones.**

Nota

Cada valor tiene un valor numérico asociado. Sumando los valores de los botones con el de los iconos, se podrán hacer combinaciones sin tener que escribir la expresión de valor.

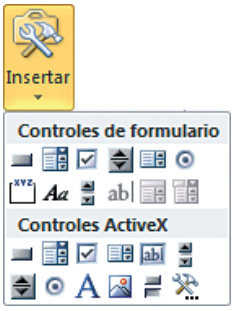
VALORES QUE PUEDE TENER EL ARGUMENTO BOTONES

Valor	Valor numérico	Descripción
vbOKOnly	0	Muestra solamente el botón Aceptar
vbOKCancel	1	Muestra los botones Aceptar y Cancelar
vbAbortRetryIgnore	2	Muestra los botones Anular, Reintentar e Ignorar
vbYesNoCancel	3	Muestra los botones Sí, No y Cancelar
vbYesNoCancel	4	Muestra los botones Sí y No
vbRetryCancel	5	Muestra los botones Reintentar y Cancelar
vbCritical	16	Muestra el icono de mensaje crítico
vbQuestion	32	Muestra el icono de pregunta de advertencia
vbExclamation	48	Muestra el icono de mensaje de advertencia
vbInformation	64	Muestra el icono de mensaje de información
vbDefaultButton1	0	El primer botón es el predeterminado
vbDefaultButton2	256	El segundo botón es el predeterminado
vbDefaultButton3	512	El tercer botón es el predeterminado
vbDefaultButton4	768	El cuarto botón es el predeterminado
vbApplicationModal	0	Aplicación modal: el usuario debe responder al cuadro de mensajes antes de poder seguir trabajando en la aplicación actual
vbSystemModal	4096	Sistema modal: se suspenden todas las aplicaciones hasta que el usuario responda al cuadro de mensajes

Y los valores que puede devolver son:

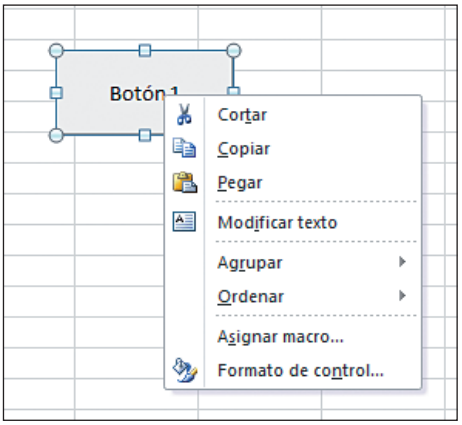
VALORES QUE PUEDE DEVOLVER EL ARGUMENTO BOTONES		
Valor	**Valor numérico**	**Descripción**
vbOK	1	Se ha pulsado el botón aceptar
vbCancel	2	Seha pulsado el botón cancelar
vbAbort	3	Se ha pulsado el botón anular
vbRetry	4	Se ha pulsado el botón Reintentar
vbIgnore	5	Se ha pulsado el botón Ignorar
vbYes	6	Se ha pulsado el botón Sí
vbNo	7	Se ha pulsado el botón No

Se va a ver cómo funcionan estos mensajes mediante ejemplos. Se meterá el código dentro del evento clic de un botón. Para ello, en la ficha **Programador,** se despliega el botón **Insertar** del grupo **Controles** y se selecciona el primer icono correspondiente a la herramienta **Botón.**



Icono de Botón en el grupo Controles del botón Insertar

Tan solo habrá que dibujarlo en la página. Para asignarle una macro, se hará clic con el botón derecho del ratón sobre él y se seleccionará la opción **Asignar macro.** Se abrirá el cuadro de diálogo **Macro,** que ya se conoce.



Asignar macro

Para comenzar con los ejemplos, hay que fijarse en el código y el mensaje que genera:

```
Sub Botón1_Haga_clic_en()
   MsgBox "¿Desea Salir de Excel?"
End Sub
```

El mensaje que generaría sería el siguiente:

Mensaje generado

Al no especificar ningún parámetro, aparece por defecto el botón **Aceptar.**

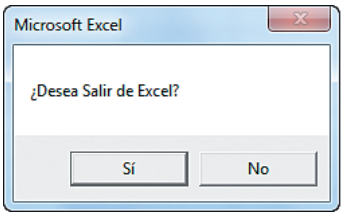
Nota

Lógicamente, si se pulsa el botón Aceptar, no se cerrará el programa, ya que no se han escrito las instrucciones adecuadas para que se cierre.

Se va a mostrar el mismo mensaje, pero con los botones **Sí** y **No.**

```
Sub Botón1_Haga_clic_en()
    MsgBox "¿Desea Salir de Excel?", vbYesNo
End Sub
```

El mensaje que generaría sería el siguiente:



Mensaje generado

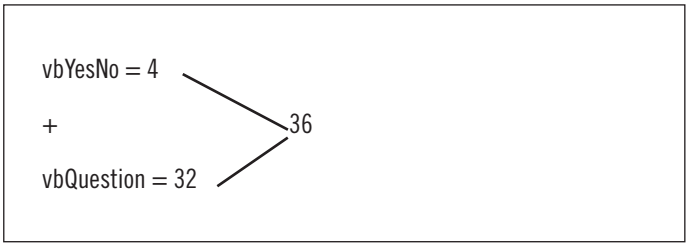
Aparecen los dos botones. Ahora se añadirá también el icono de interrogación, para enfatizar que el programa está preguntando algo.

```
Sub Botón1_Haga_clic_en()
    MsgBox "¿Desea Salir de Excel?", vbYesNo + vbQuestion
End Sub
```

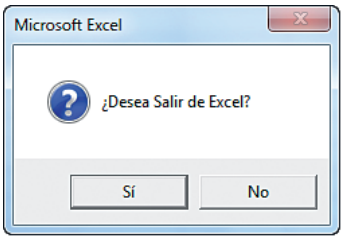
Esta misma expresión se podría escribir de la siguiente forma resumida:

```
Sub Botón1_Haga_clic_en()
   MsgBox "¿Desea Salir de Excel?", 36
End Sub
```

Se han cambiado los botones y el icono por un número. Este número es el resultado de sumar sus valores numéricos.



El mensaje quedaría de la siguiente manera:

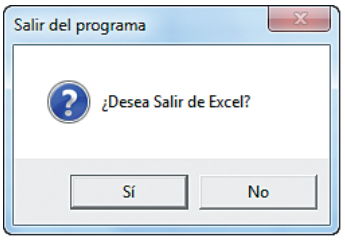


Mensaje generado

Se le va a cambiar el título a la ventana:

```
Sub Botón1_Haga_clic_en()
   MsgBox "¿Desea Salir de Excel?", 36, "Salir del programa"
End Sub
```

El mensaje quedaría así:



Mensaje generado

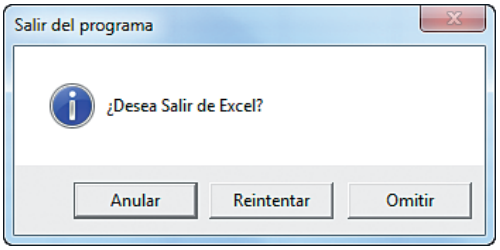
Se va a ver un último ejemplo en el que se mostrarán tres botones y un icono de información. Se ve que **vbAbrotRetryIgnore** vale **2** y el icono de información **64.** El código sería el siguiente:

```
Sub Botón1_Haga_clic_en()

    MsgBox "¿Desea Salir de Excel?", 66, "Salir del programa"

End Sub
```

El mensaje quedaría de la siguiente manera:



Mensaje generado

En estos momentos, ya se debe saber construir un mensaje, pero lo realmente interesante es el valor que devuelve y lo que uno hará según el valor devuelto al pulsar un botón.

Siguiendo con los ejemplos anteriores, se va a mostrar un cuadro de diálogo con los botones **Sí** y **No** y, dependiendo del botón que se pulse, se cerrará el programa o no:

```
Sub Botón1_Haga_clic_en()

respuesta=MsgBox(("¿Desea Salir de Excel?"), 36, "Salir del
programa")

If respuesta = vbYes Then ActiveWorkbook.Close

End Sub
```

En este caso, al pulsar el botón, la variable respuesta toma un valor y, si ese valor es igual a **vbYes,** cerrará el programa. Recuérdese la tabla de valores devueltos que se vio con anterioridad.

Recuerde

Si, en lugar de vbYes, se hubiese escrito el valor numérico 6, el resultado hubiese sido el mismo. Véanse los valores de la tabla.

4. Cuadros de introducción de datos

Existe otra función muy parecida a **MsgBox,** pero, en lugar de mostrar un texto en pantalla y unos botones, se podrán realizar preguntas al usuario y este deberá introducir, a través del teclado, dicha información. Se trata de la función **InputBox.**

Esta función guardará la respuesta en una variable, siendo su sintaxis la siguiente:

InputBox (mensaje [, titulo][, pordefecto][, xpos][, ypos][, helpfile, contexto])

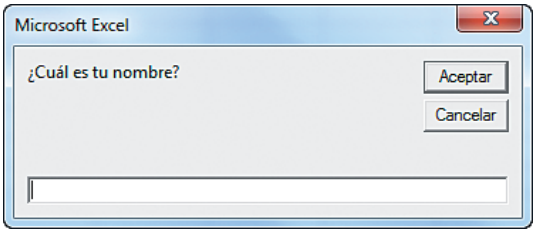
- **mensaje:** mensaje que aparecerá en el cuadro de diálogo. El mensaje suele estar limitado a 1.024 caracteres.
- **título:** lo que muestra la barra de título del cuadro de diálogo.
- **pordefecto:** texto que aparecerá en el cuadro de texto del mensaje. Si se omite, este cuadro de texto aparecerá vacío.
- **Xpos,Ypos:** utilizando estos valores, se podrá situar el cuadro de diálogo en cualquier parte de la pantalla.

- **Helpfile:** al igual que en los **MsgBox,** también se puede llamar a un fichero de ayuda y proporcionar ayuda interactiva para el cuadro de diálogo.
- **Contexto:** expresión numérica que es el número de contexto de **Ayuda.**

Puede verse a través de un ejemplo la utilización de esta función:

```
Sub Botón1_Haga_clic_en()
   respuesta = InputBox("¿Cuál es tu nombre?")
   If respuesta <> "" Then
      ActiveCell = respuesta
   End If
End Sub
```

El mensaje que aparecería sería el siguiente:



Ejemplo de mensaje

Como se puede observar, primeramente, con la sentencia **If,** se está comprobando que no se ha dejado el cuadro de texto vacío. A continuación, se le asigna a la celda activa el valor que se ha introducido.

5. Creación de cuadros de diálogo propios

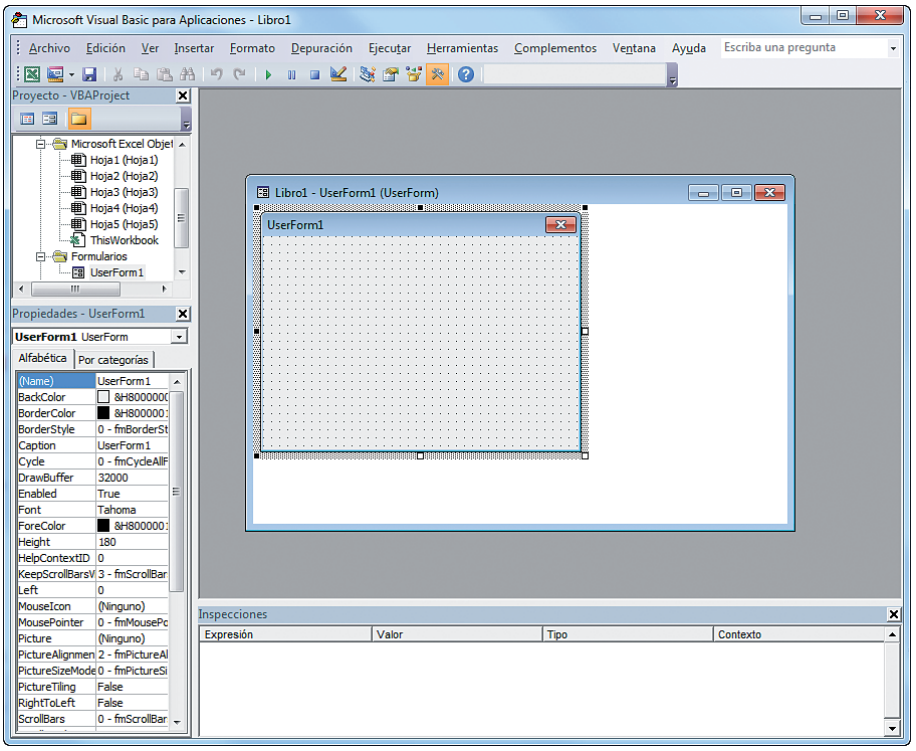
Ya se conocen los cuadros de dialogo prediseñados de *Excel.* Puede ocurrir que no cumplan con las expectativas y se necesite construir los cuadros propios. *Excel* permite construirlos y se les llama **Formularios.**

Nota

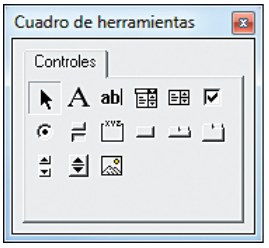
El diseño de los formularios puede ser una tarea muy simple o, por el contrario, si se quiere una mayor funcionalidad, se puede convertir en una tarea muy compleja.

Para crear un formulario propio, hay que situarse en la ventana de *Visual Basic* y seleccionar la opción **UserForm** del menú **Insertar.** De este modo, aparecerá una ventana con un formulario vacío y un cuadro con herramientas para la elaboración del formulario.

El formulario por si solo sirve de poco. Hay que añadirle elementos que realicen las tareas y se apliquen a la hoja de cálculo. Los elementos que se pueden incluir se encuentran en el cuadro de diálogo **Cuadro de herramientas,** que se abrirá desde el menú **Ver.**



Ventana para la elaboración del formulario



Cuadro de herramientas

Las principales herramientas son:

- **Etiquetas:** permite presentar un texto. Se suele utilizar cuando el texto a presentar tiene un carácter fijo (por ejemplo, una información que no va a cambiar o una serie de datos). Bastará con seleccionarla y podrá

dibujarse dentro del formulario. Con la propiedad **Caption** de la ventana **Propiedades** se establecerá el texto a mostrar.

Etiquetas

- **Cuadros de texto:** cajas de texto como las que se está acostumbrado a ver en cualquier cuadro de diálogo. Esta herramienta normalmente se utiliza para introducir texto o para mostrar textos que van a ser cambiados.

Cuadros de texto

- **Cuadros de lista y cuadros combinados:** los primeros son los elementos que muestran una lista directamente, para seleccionar una opción de forma directa. Los cuadros combinados fusionan las características de los cuadros de lista y los campos de texto. En ellos, también aparecerá una lista, pero se debe pulsar antes el botón que los acompaña.

Cuadros de lista y cuadros combinados

Los cuadros combinados y los de lista tienen, entre otras, las siguientes propiedades:

- **ListCount:** indica el número de elementos que tiene la lista
- **ListIndex:** indica el número de orden del elemento seleccionado dentro de la lista.
- **AddItem:** añade un elemento a la lista.
- **RemoveItem:** elimina un elemento de la lista.

- **Text:** obtiene el elemento seleccionado.
- **List (n):** obtiene el elemento cuyo orden dentro de la lista es n.

Tan solo habrá que seleccionar un elemento de la lista para que se resalte en azul y programar las instrucciones adecuadas para su utilización.

- **Casilla de verificación y botón de opción:** la casilla de verificación, más conocida como **ChekBox,** permitirá activar o desactivar una opción. Si la opción está activa, tomará el valor **True** y, si está desactivada, el valor **False.** Los botones de opción u **OptionButton** permitirán seleccionar solo una opción de todas las que se muestren.

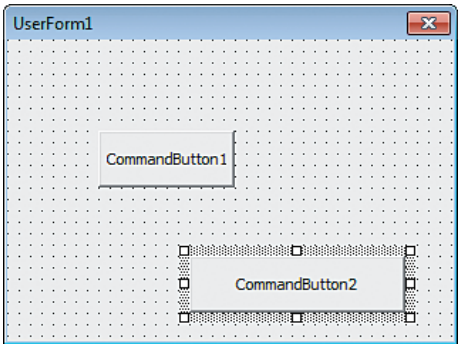
Casilla de verificación y botón de opción

- **Marco:** este control sirve para agrupar controles. En el caso de los **OptionButton,** solo se podrá seleccionar uno de los que el marco contenga, pudiendo existir múltiples marcos.

Marco

- **Botones de comando:** con esta herramienta se podrán dibujar botones en el formulario. Son muy útiles, ya que, dentro del evento clic del que disponen, se pueden escribir las instrucciones necesarias para que, al pulsarlo, se ejecuten.

Herramienta botón de comando



Botones en el formulario

Este control permite mostrar imágenes en el formulario.

Herramienta Picture

Para cargar una imagen, se podrá hacer desde la propiedad **Picture,** en la que se abrirá un cuadro de diálogo en el que hay que localizar la imagen que se quiere insertar.

Nota

Todas estas herramientas podrán utilizarse también en la hoja de cálculo directamente. Para ello, hay que situarse en la ficha Programador y, desde el grupo Controles, seleccionar el que se quiera y dibujarlo directamente en la hoja.

5.1. Propiedades de los controles

Son muchas las propiedades que abarca cada control en *Visual Basic.* Muchas de ellas no se utilizarán nunca, pero hay otras que son esenciales y que se utilizarán con bastante frecuencia.

Cuando se selecciona un control, aparecen las propiedades de este en el panel de **Propiedades** y se podrán modificar directamente en modo de diseño. Pero también se accederá a las propiedades de los controles cuando se esté ejecutando una macro.

Muchas de las propiedades son comunes a todos o parte de los controles y otras son específicas. Las principales son:

- **Name:** con ella, se establece el nombre del control para poder hacer referencia a él. Por defecto, siempre que se cree un control, aparecerá un nombre que se puede cambiar.
- **Caption:** especifica el texto que se verá en el control.
  Ejemplo: el texto que llevará un botón, el texto que aparecerá en una etiqueta, etc. El texto siempre debe aparecer entre comillas.

```
Etiqueta.Caption = "Hola Mundo"
```

- **Value:** con ella, se obtendrá el valor del control. Cuando el control es de texto, como un **Textbox** o caja de texto, con esta propiedad se obtendrá lo que haya escrito en la caja de texto.
  Ejemplo:

```
Variable = Textbox1.value
```

De esta forma, se asigna a una variable el valor que tenga la caja de texto.

Si el control es un **Optionbutton** o un **Commandbutton, Value** indicará solamente si está seleccionado o no. Devolverá el valor **True** en caso de estar seleccionado. Normalmente, se hará referencia a ella con alguna sentencia condicional o de repetición.

Ejemplo:

```
If optionbutton.value = true then
   Variable1 = variable2 * 10
End If
```

- **Clear:** borrará el contenido de una caja de texto, de un **Combobox** y, en definitiva, de cualquier componente que admita texto.
- **Enabled:** activará o desactivará un control. Seguro que se ha visto en muchos programas algún botón cuyo texto está gris y no puede pulsarse, pues bien, esto es porque tiene la propiedad **Enabled** a **False.** Por defecto, al crear un control, está a **True** y se podrá cambiar la propiedad tanto en modo diseño como en ejecución.