CAPÍTULO 1. Diagramas de Flujo

También conocidos como flowchart, en estos se utilizan símbolos estándar, en el que cada paso para la elaboración del programa se representa con el símbolo y orden adecuados, unidos o conectados por flechas, también llamadas líneas de flujo, esto por que indican el sentido en el que se mueve el proceso.

En resumen el diagrama de flujo es un medio de presentación visual y gráfica del flujo de datos a través de un algoritmo.

ALGORITMO lo podemos definir como un conjunto de pasos, procedimientos o acciones que nos permiten alcanzar un resultado o resolver un problema.

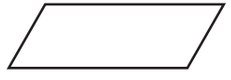
CARACTERÍSTICAS ELEMENTALES DE UN ALGORITMO

Debe ser exacto en el orden de la realización de cada uno de los pasos a seguir.
Debe obtenerse el mismo resultado después de seguirse dos veces el algoritmo.
Un algoritmo debe ser finito, debe tener una condición que le permita terminar.
Debe tener tres partes fundamentales como son información dada para el algoritmo (entrada), los cálculos necesarios para solucionar el problema (procesos) y respuestas o resultados finales de los cálculos (salida).

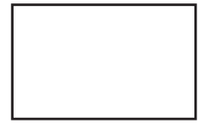
SÍMBOLOS PRINCIPALES EN LOS DIAGRAMAS DE FLUJO

Terminal: Representa el comienzo o final (fin) de un programa.

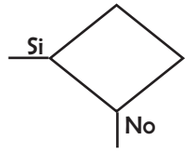
Entrada/Salida: Representa la entrada de datos en la memoria desde los periféricos (por ejemplo el teclado).



Proceso: Aquí se pueden dar las operaciones para originar un cambio de valor, formato o posición de la información almacenada en memoria; también se pueden realizar operaciones matemáticas como: suma, resta, multiplicación, división, exponenciación, o cualquier combinación entre estas.



Decisión: Es la operación lógica que se debe realizar dependiendo de un dato almacenado para tomar dos caminos alternativos (si o no). Permite comprobar condiciones particulares.



Conector: Sirve para enlazar cualquier parte del diagrama.

Impresora: Es un símbolo alterno al de Entrada/Salida de datos

Veamos a continuación un conjunto de reglas o normas que nos permiten construir un diagrama de flujo.

1. Todo diagrama de flujo debe tener un inicio y un fin.

2. Las líneas utilizadas para indicar la dirección del diagrama deben ser rectas, horizontales o verticales, nunca se deben cruzar entre si.

3. No deben haber líneas sin conexión a los demás elementos del diagrama de flujo.

4. Un diagrama de flujo se debe construir de arriba hacia abajo y de requerirse de izquierda a derecha.

5.La notación o símbolos utilizados en el diagrama de flujo son independientes del lenguajes de programación utilizado para la elaboración del programa o aplicación.

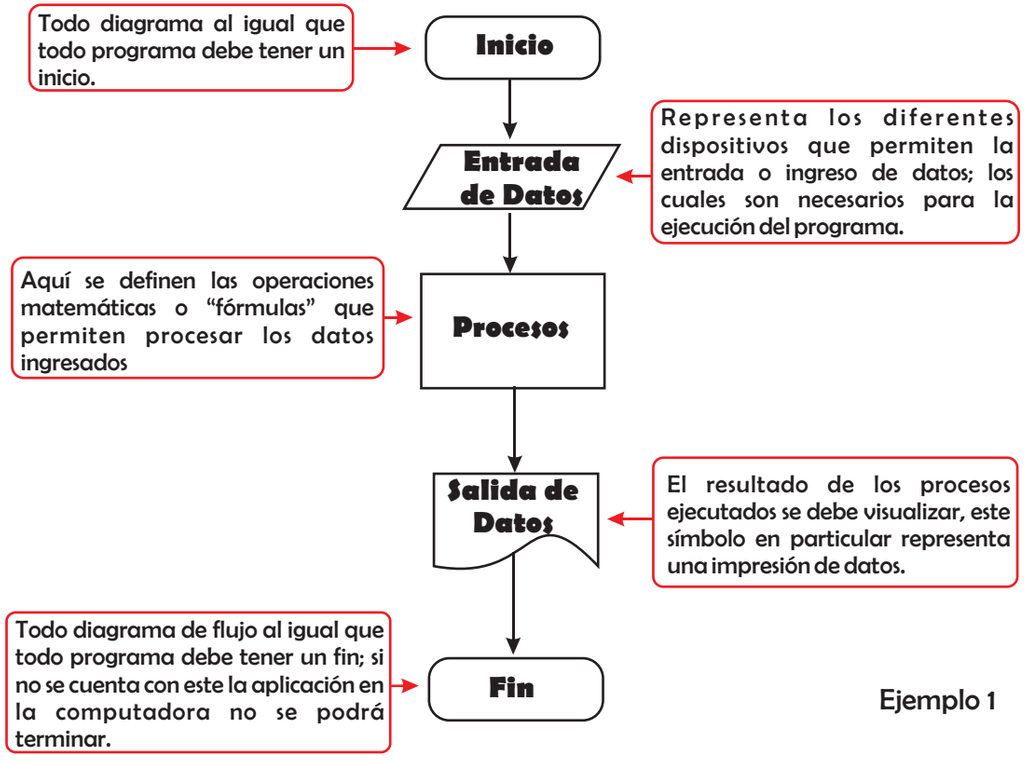
6. No puede llegar más de una línea de conexión a un símbolo.

Para tener en cuenta: En programación utilizamos las llamadas expresiones lógicas, que están constituidas por números, constantes o variables y operadores lógicos o relacionales, que pueden tomar un valor de falso o verdadero dependiendo del resultado de la evaluación para seguir por un camino determinado. Los operadores relacionales son aquellos que permiten la comparación y los operadores lógicos son los que permiten formular condiciones.

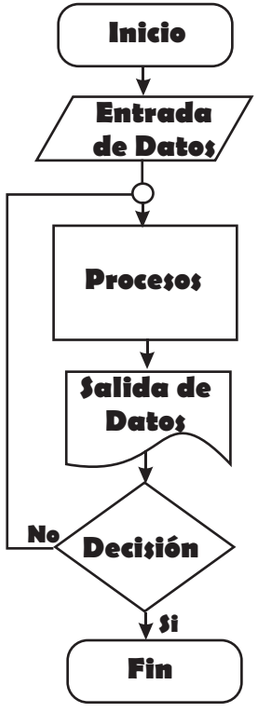
Operadores relacionales		Operadores Lógicos	
=	Igual que	NO	No es cierto que P
< >	Diferente a		
<	Menor que	Y	P sin embargo Q
>	Mayor que		
<=	Menor o igual que	O	o P o Q o Ambas
>=	Mayor o igual que		

ESTRUCTURAS BÁSICAS DE UN DIAGRAMA DE FLUJO

Esta es una de las estructuras más simples que se pueden dar para resolver un algoritmo, en donde se observa una entrada de datos, el procesamiento de estos y la salida de los datos.



Ejemplo 1



En esta estructura de Diagrama de Flujo encontramos los mismos componentes del ejemplo anterior, pero se utiliza un símbolo de "Decisión", el cual permite tomar uno y solo uno de los dos caminos disponibles, siempre y cuando se cumpla cierta condición, al dejar de cumplirse la condición inicial se continua por la otra alternativa.

A este tipo de estructuras cíclicas o repetitivas también se les llama bucles o loops.

Permiten la ejecución del mismo proceso todas las veces de forma idéntica mientras se de la condición u operación lógica planteada.

Ejemplo 2

ESTRUCTURAS REPETITIVAS EN UN DIAGRAMA DE FLUJO

Es común encontrar en la elaboración de un diagrama de flujo operaciones o procesos que se deben ejecutar un determinado número de veces, en donde las instrucciones son las mismas, pero los datos sobre los que operan varían, a esto lo llamamos un ciclo, según se observa en el ejemplo 2.

Estructura repetitiva FOR (Repetir): Es la adecuada para un ciclo que se ejecutará un número definido de veces.

Estructura repetitiva WHILE (Mientras): Es la estructura adecuada cuando no sabemos el número exacto de veces que se debe repetir un ciclo.

PROGRAMAS PARA COMPUTADORAS

Después de mirar y analizar los dos ejemplos anteriores podemos decir que un programa es un conjunto de instrucciones que sigue la computadora para alcanzar un resultado específico o expresar un resultado obtenido del procesamiento de unos datos dados. Pero para poder dar estas instrucciones a la computadora debemos utilizar un lenguaje de programación, estos están constituidos por unas reglas sintácticas y semánticas que son las que hacen posible la escritura correcta de las instrucciones dadas. Estos lenguajes básicamente pueden ser de bajo o alto nivel, en donde los últimos permiten el uso de ciertas palabras o frases compuestas que pueden ser fácilmente entendidas por los usuarios de dichos lenguajes o programadores.

•Lenguajes de Bajo Nivel: En su forma más simple son unos impulsos o señales electrónicas convertidas a unos y ceros, los cuales pueden ser interpretados directamente por un microprecesador.

•Lenguajes de Alto Nivel: Generalmente están compuestos por elementos del lenguaje natural usado por los seres humanos, sin embargo para poder interactuar con estos lenguajes y la computadora se requiere del manejo de ciertas estructuras de sintaxis que permitan una interacción acertada hombre - maquina , siendo el lenguaje el puente de comunicación entre estos dos.

Se requiere además de un lenguaje de alto nivel un interprete o compilador que es en realidad un traductor entre el usuario de la computadora y esta misma.

En resumen un lenguaje de programación básicamente es un lenguaje artificial utilizado para dar una serie de instrucciones a una computadora.

CAPÍTULO 2. Entorno de Visual Basic

¿QUÉ ES VISUAL BASIC?

Visual Basic es un sistema de programación gráfica rápido y sencillo, pensado para un ambiente de ventanas, de tal modo que se pueden dibujar objetos seleccionándolos de una caja de herramientas, arrastrándolos a un formulario o pantalla y asignándoles una programación que puede ser ejecutada de inmediato.

Visual Basic está diseñado como una plataforma de programación orientada a objetos y eventos; los objetos (ventanas y controles) se encuentran disponibles en una tabla de herramientas que permite arrastrarlos encima de una ventana, pantalla o formulario, y al dar clic sobre estos escribir los comandos necesarios para programar los eventos o acciones concernientes a cada elemento.

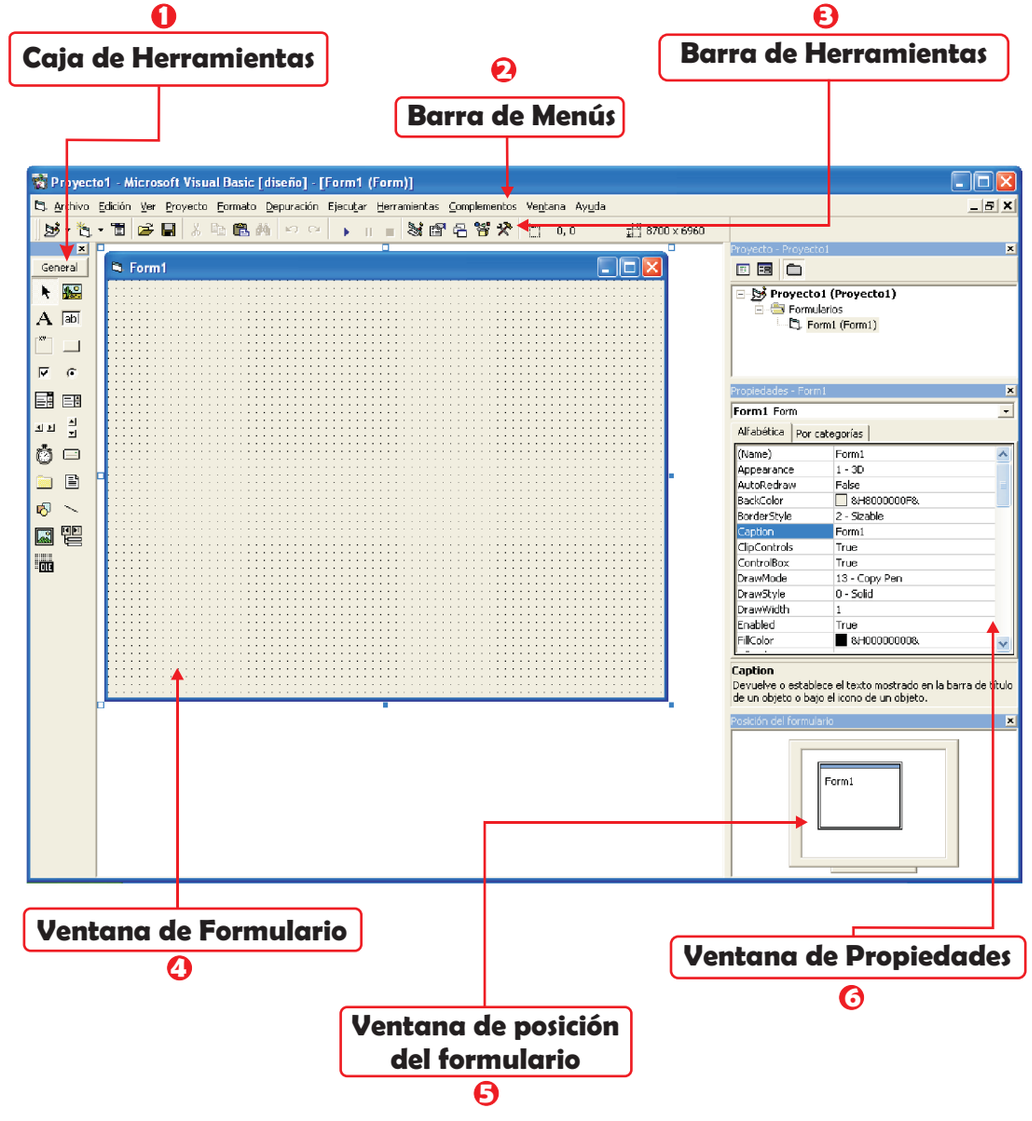
En Visual Basic un formulario es una ventana de fondo que sirve para situar en ella los controles gráficos requeridos. Como programador en Visual Basic, puedo utilizar tantos formularios como sea necesario. Los controles u objetos se dibujan sobre el formulario con el fin de permitir el ingreso de datos o visualizar gráficos.

Con Visual Basic puedo crear aplicaciones robustas como las más conocidas comercialmente (por ejemplo Microsoft Word), con barras de herramientas y barras de menús, también crear archivos instaladores que me permitan instalar la aplicación en cualquier PC. compatible.

También contamos en Visual Basic con una plataforma que maneja un lenguaje de alto nivel, lo cual me permite:

- Verificar la sintaxis de cada una de las líneas de programación.
- Contar con un depurador de código interactivo.
- Manejar funciones lógicas, matemáticas y de cadenas de caracteres.

Principales Componentes de la Interfaz Gráfica de Visual Basic



1. La Caja de Herramientas: ésta se encuentra conformada por un grupo de botones con los que podemos dibujar diferentes objetos o controles sobre el formulario para construir la aplicación.

2. Barra de Menús: es la típica serie de menús desplegables de Windows en donde encontramos todas las herramientas necesarias para manipular los controles y las ventanas durante el desarrollo de la aplicación.

3. Barra de Herramientas: presenta unos botones que representan las utilidades mas importantes contenidas también en la barra de menús.

Cabe destacar los botones de Inicio, Interrupción y Terminar, ya que son los que permiten correr una simulación de la aplicación antes de compilarla o hacer el ejecutable correspondiente a dicha aplicación

4. Ventana del Formulario: es en esta ventana en donde se colocan o pegan los controles que nos permiten definir el diseño o aspecto de la aplicación.

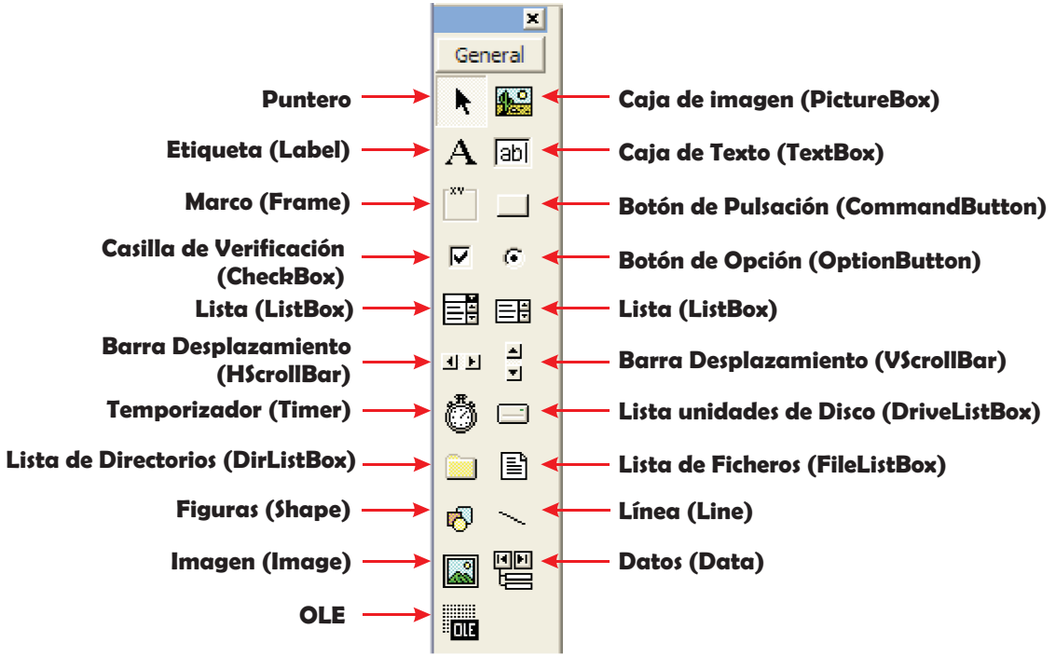
5. Ventana de posición del formulario: en esta visualizamos la ubicación del formulario al correr la aplicación, también desde aquí podemos modificar dicha posición arrastrando el formulario con un clic del mouse.

6. Ventana de propiedades: en esta encontramos el conjunto de propiedades asociadas a cada uno de los controles colocados en el formulario.

Para recordar:

• Los controles colocados en cada aplicación permiten el ingreso de datos o la visualización de estos al interactuar con el programa.
• Las propiedades de los controles son atributos o características tales como color, tamaño, ubicación, tipo de fuente (Font), etc.
• Para ver las propiedades correspondientes a cada control primero debe seleccionar dicho control.
• Algunas propiedades dentro de la ventana de propiedades despliegan un cuadro de dialogo, ya sea para ingresar o seleccionar un valor.
• El nombre de los objetos lo asigna Visual Basic y este lo encontramos en la propiedad "Name" de cada objeto.

Conozcamos la Caja de Herramientas en Visual Basic



• Puntero: se utiliza para manipular los controles existentes sobre el formulario. Con el puntero se puede seleccionar , mover y ajustar el tamaño de los objetos.

• Caja de Imagen: se utiliza para visualizar una imagen importada desde un fichero o archivo.

• Etiqueta: se utiliza para colocar un texto de una o más líneas que no queremos sea modificado por el usuario.

• Caja de Texto: en esta se puede escribir o visualizar un texto, también permite el ingreso de datos.

• Marco: se utiliza para agrupar objetos relacionados entre sí, o para mejorar el aspecto del formulario.

• Botón de Pulsación: siempre lleva asociada una orden la cual se llevará a cabo cuando el usuario pulse clic sobre el.

• Casilla de Verificación: permite seleccionar una opción.

• Botón de Opción: se utiliza para seleccionar una opción de entre varias.

• Lista Desplegable: es una combinación de caja de texto y lista.

• Lista: contiene una lista de elementos de los que el usuario puede seleccionar uno.

• Barra de Desplazamiento horizontal y barra de desplazamiento vertical: permiten seleccionar un valor dentro de un rango de valores. Estos controles no son lo mismo que las barras de desplazamiento de una ventana.

• Temporizador: permite activar procesos a intervalos determinados de tiempo.

•Lista de Unidades de Disco: permite visualizar la lista de unidades de disco disponibles y seleccionar una.

• Lista de Directorios: permite ver los directorios a los que el usuario puede dirigirse.

• Lista de Ficheros: permite al usuario dirigirse a unos ficheros específicos.

• Figuras: permite dibujar en el formulario rectángulos, cuadrados, elipses o círculos.

• Línea: permite dibujar líneas rectas en un formulario.

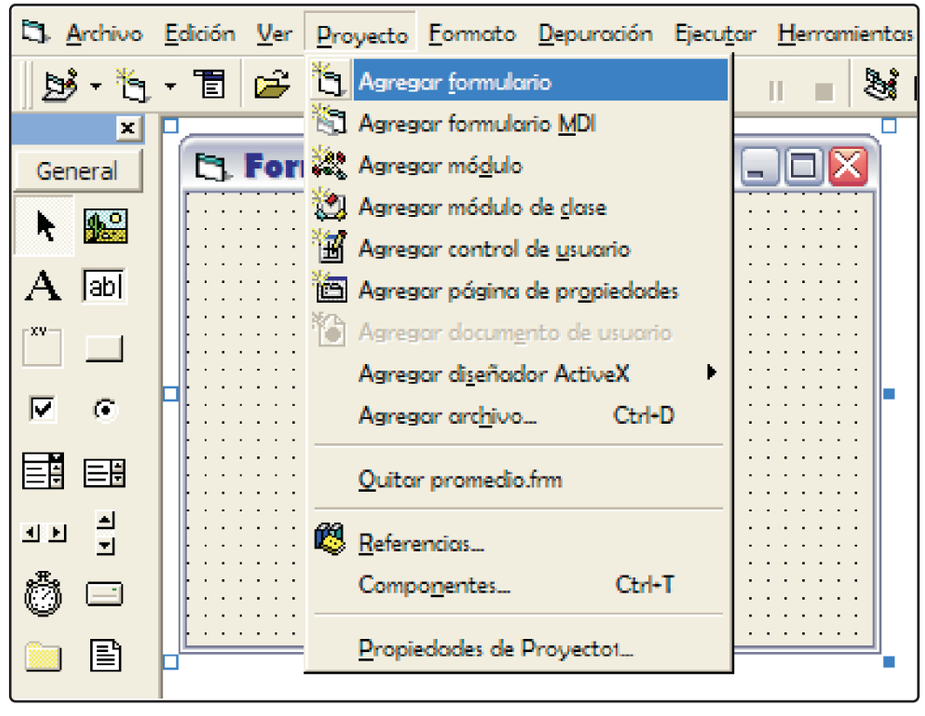
• Imagen: permite visualizar una imagen importada desde un archivo.

• Datos: permite conectarse a una base de datos existente.

• OLE: permite incrustar datos desde otra aplicación tal como Word, Excel, etc.

Para recordar:

En la elaboración de una aplicación se pueden utilizar uno o más formularios, para ingresar un nuevo formulario vamos a la opción Proyecto del menu y allí encontramos la opción agregar formulario.



• El conjunto de formularios, con sus controles y módulos conforman en Visual Basic lo que denominamos proyecto.

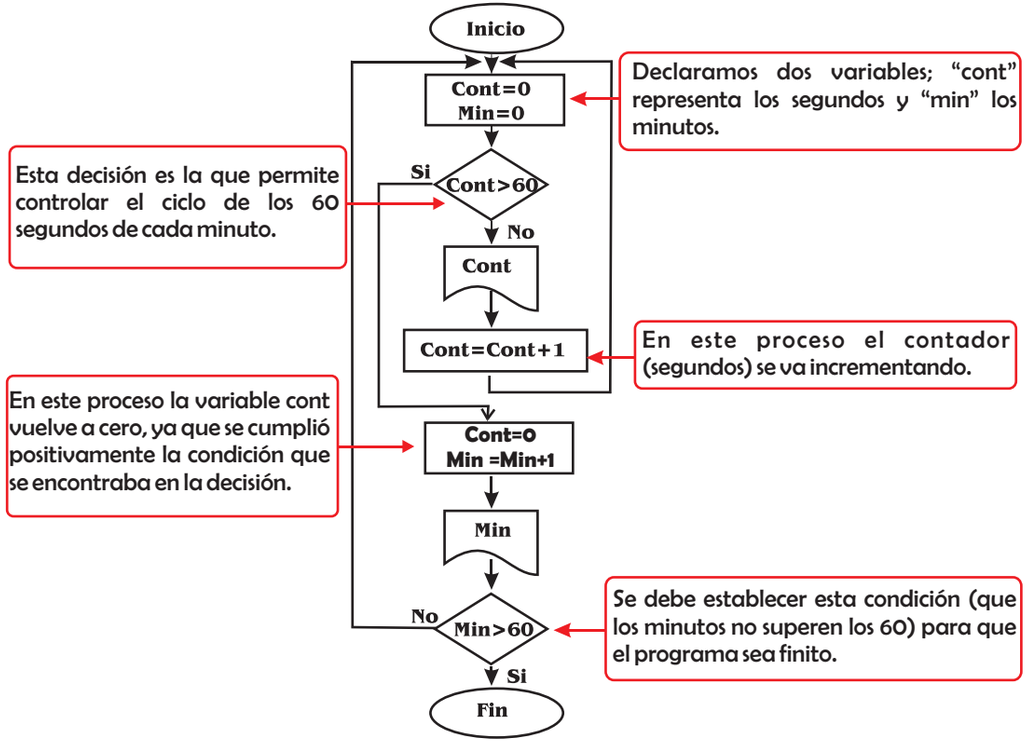
• Todo formulario es guardado como archivo con la extensión .frm

CAPÍTULO 3. Diagramas de Flujo y Visual Basic

Después de reconocer los principales elementos de la interfaz gráfica de Visual Basic, vamos a desarrollar en este capítulo dos ejercicios que nos permitirán entender la relación existente entre el Diagrama de flujo y el código de programación que nos permitirá desarrollar la aplicación.

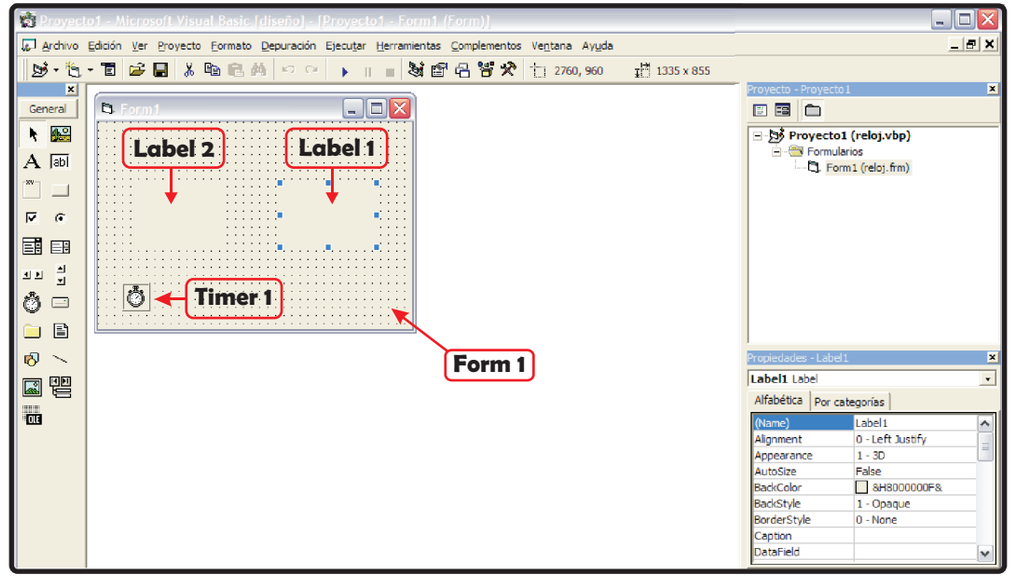
Ejemplo 1: Construya un diagrama de flujo que represente la función de un cronómetro, en donde se impriman los segundos hasta llegar a sesenta y a partir de allí también imprima los minutos.

El diagrama de flujo sería el siguiente:



Ahora veamos como sería la programación en Visual Basic

Identifiquemos los componentes necesarios para la realización de nuestra aplicación



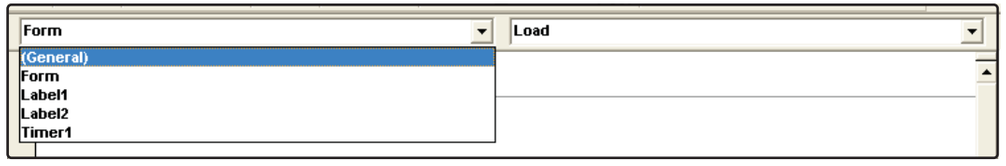
Label1 (Etiqueta 1): en esta se visualizará la variable "cont" o los segundos que van corriendo. Las propiedades que debo modificar de este objeto son: Font (tipo de fuente (letra)y tamaño) y Alignment (Center).

Label2 (Etiqueta 2): en esta se visualizará la variable "min" o los minutos que van corriendo. Las propiedades que debo modificar de este objeto son: Font (tipo de fuente (letra)y tamaño) y Alignment (Center).

Timer1 (Temporizador 1): es un objeto no visible en la ejecución del programa, en este se escribe todo el código necesario para que el cronómetro inicie. Las propiedades que debo modificar son: Interval, que se debe colocar en mil.

Form1 (Formulario 1): damos clic sobre el Form1, el cual me despliega un menú, del que voy a seleccionar (General / Declaraciones), y escribo lo siguiente:

```
Dim contador
Dim min
```



Damos doble clic sobre el Timer1 y escribimos el siguiente código, esta nueva ventana que se abre la llamamos ventana de código:

```
Private Sub Timer1_Timer( )

If contador < 60 Then
Label1.Caption = Str$(contador)
contador = contador + 1
Else
contador = 0
min = min + 1
Label2.Caption = Str$(min)
End If

End Sub
```

Inicio de Subrutina

Utilizamos la estructura Si entonces, si no, fin del si; la cual nos permite mantener el ciclo de los segundos de 1 a 60.

Fin de Subrutina

Para tener en cuenta:

Las líneas de inicio y fin de subrutina ya se encuentran escritas, el código lo debemos escribir entre estas tal como lo muestra este ejercicio.

Str$ hace referencia a una cadena de caracteres, ya sea que se toma un valor de esta o se convierte una variable (valor) en texto.

Ejemplo 2: Dadas 4 calificaciones de un alumno construya un diagrama de flujo que saque el promedio de estas e imprima el resultado.

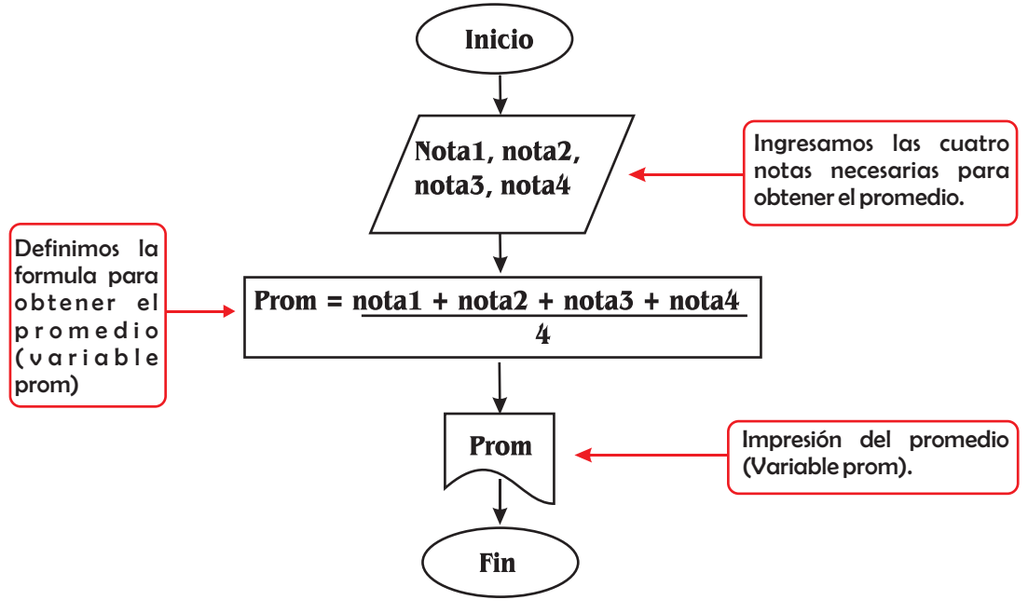
El diagrama de flujo sería el siguiente:

Los datos necesarios para elaborar este programa son los siguientes: Prom, nota1, nota2, nota3, nota4.

Explicación de las variables:

Prom: Variable que almacena el promedio de las notas obtenidas por el alumno y lo divide por cuatro.

Nota1, Nota2, Nota3, Nota4: Notas obtenidas por el estudiante, datos reales.



Ahora veamos como sería la programación en Visual Basic:

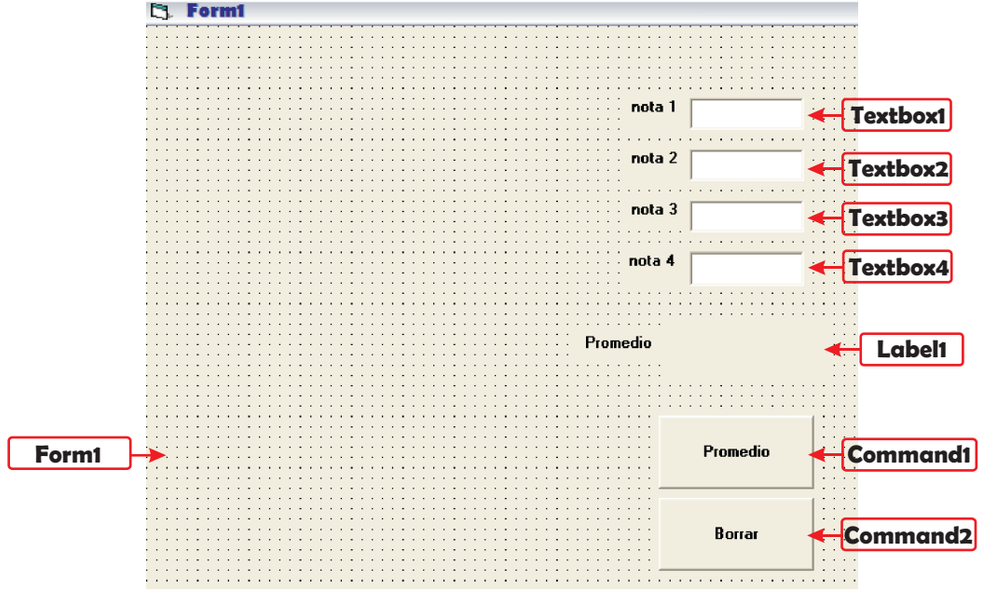
Antes de ver el código de programación es necesario familiarizarnos con los siguientes términos:

Keypress: Es un evento útil para interceptar pulsaciones de teclas realizadas en un control TextBox o ComboBox. Esto le permite comprobar inmediatamente la validez de las pulsaciones o el formato de los caracteres a medida que se escriben

SetFocus: método para establecer el enfoque en el control.

Keyascii: se utiliza para realizar operaciones sobre cadenas de caracteres (texto) y traducir estos a un número que el control pueda interpretar.

Identifiquemos los componentes necesarios para la realización de nuestra aplicación



Para el código de nuestra aplicación damos doble clic sobre alguno de los componentes que se encuentran sobre el formulario o sobre este mismo, a continuación aparece la ventana de código y escribimos:

```
Private Sub Command1_Click()
Dim prom As Double
Label1.Caption = Str$(Val(Text1.Text) + Val(Text2.Text)
+ Val(Text3.Text) + Val(Text4.Text)) / 4
End Sub
```

Evento Clic del botón de comando1 (Promedio). Aquí escribimos la fórmula que nos permitirá promediar las cuatro notas

---

```
Private Sub Command2_Click()
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Text4.Text = ""
Label1.Caption = ""
End Sub
```

Evento Clic del botón de comando 2 (Borrar). Aquí escribimos el código que nos permite borrar los datos ingresados y el promedio obtenido para empezar nuevamente.

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
Dim nota1 As Double
Text1.SetFocus
nota1 = Val(Text1.Text)
End Sub
```

Evento Key press del textbox1 (Nota1). Aquí escribimos el código que nos permite tomar el valor que el usuario ingresa por el teclado y almacenarlo en la variable nota1.

```
Private Sub Text2_KeyPress(KeyAscii As Integer)
Dim nota2 As Double
Text2.SetFocus
nota2 = Val(Text2.Text)
End Sub
```

Evento Key press del textbox2 (Nota2). Aquí escribimos el código que nos permite tomar el valor que el usuario ingresa por el teclado y almacenarlo en la variable nota2.

```
Private Sub Text3_KeyPress(KeyAscii As Integer)
Dim nota3 As Double
Text3.SetFocus
nota3 = Val(Text3.Text)
End Sub
```

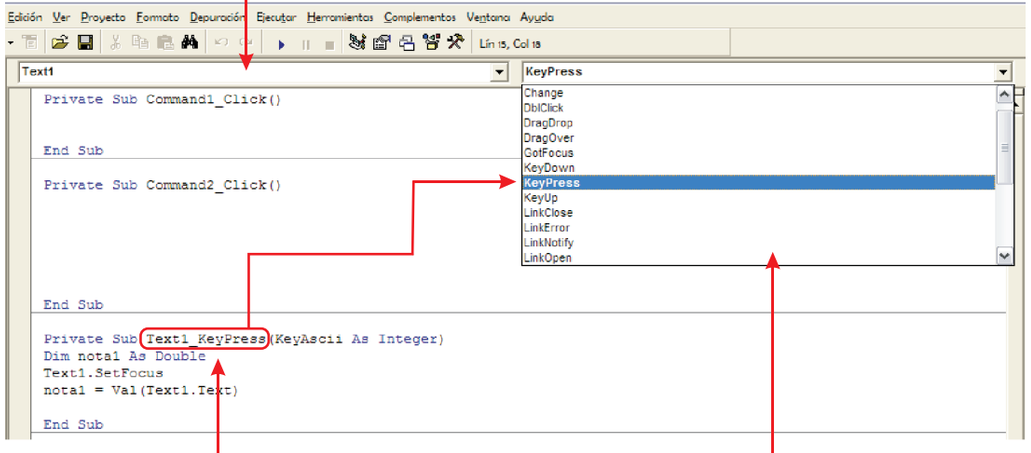
Evento Key press del textbox3 (Nota3). Aquí escribimos el código que nos permite tomar el valor que el usuario ingresa por el teclado y almacenarlo en la variable nota3.

```
Private Sub Text4_KeyPress(KeyAscii As
Integer)Dim nota4 As Double
Text4.SetFocus
nota4 = Val(Text4.Text)
End Sub
```

Evento Key press del textbox4 (Nota4).
Aquí escribimos el código que nos permite tomar el valor que el usuario ingresa por el teclado y almacenarlo en la variable nota4.

Debemos tener en cuenta que en la ventana de código vamos a encontrar siempre el listado de componentes que conforman la aplicación a la izquierda y los diferentes eventos de cada componente a la derecha de la pantalla

Listado de componentes utilizados en el formulario como por ejemplo labels, textbox, etc.



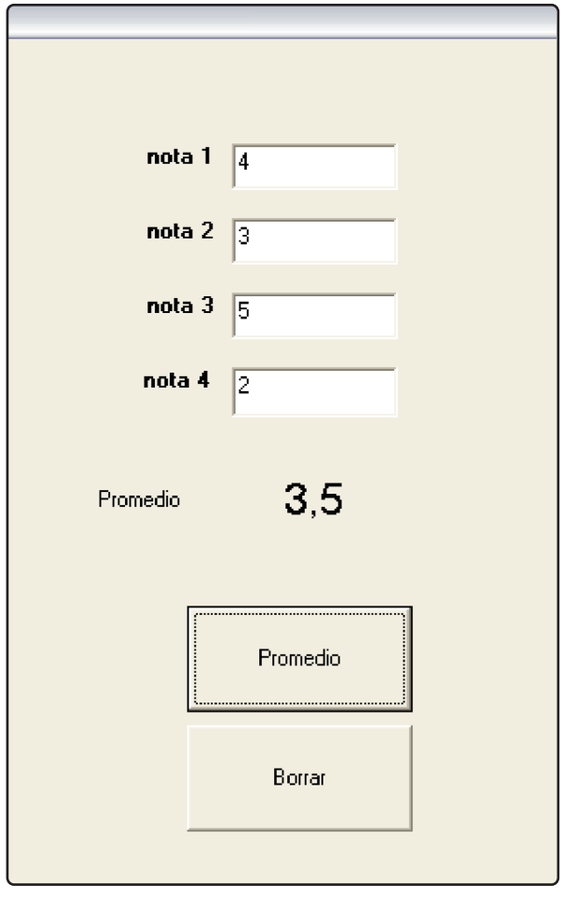
Evento Keypress correspondiente al text1

Listado de eventos correspondientes al componente o herramienta seleccionada

Para tener en cuenta:

En el evento Keypress de cada uno de los Textbox se utiliza el código: Dim nota1 As Double, para crear la variable en la cual se van a almacenar los datos correspondientes a cada una de las notas(nota1, nota2, nota3, nota4).

Usando controles adicionales como label, y propiedades como aligment, font, podemos mejorar la presentación de la Interfaz como se observa en la siguiente imagen de la aplicación ya terminada.

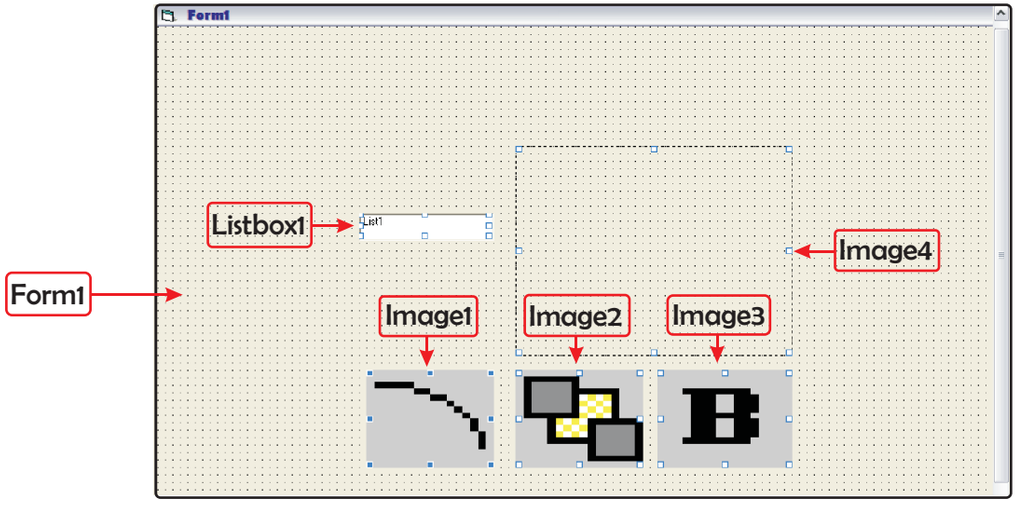


CAPÍTULO 4. Ejercicios en Visual Basic

En este capítulo realizaremos algunos ejercicios que nos permitan conocer mejor algunas estructuras de programación y herramientas de Visual Basic.

Ejercicio 1

Seleccionar de una lista (Listbox) una imagen y poder visualizarla. Para esto utilizaremos las siguientes herramientas u objetos: form1, image1, image2, image3, image4 y Listbox1.



Después de tener organizado el form1 con todos sus elementos damos doble clic sobre este mismo para escribir el siguiente código:

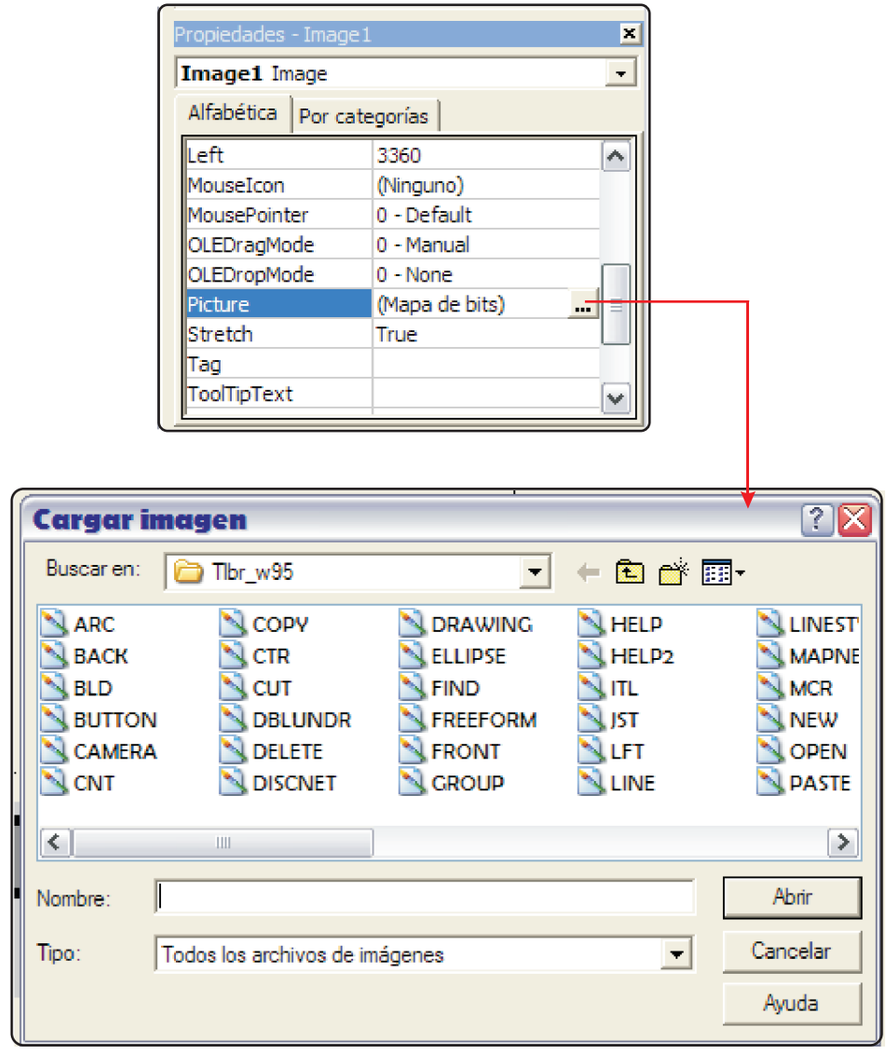
```
Private Sub Form_Load()
List1.AddItem "dibujo1"
List1.AddItem "dibujo2"
List1.AddItem "dibujo3"
End Sub
```

Inicio de subrutina

El código Additem permite que en el listbox1 aparezcan los elementos de la lista que nos permitirán seleccionar una imagen.

Fin de subrutina

A continuación debemos insertar las imagenes en los image1, image2 e image3, para lograr esto seleccionamos por ejemplo el image1 y buscamos dentro de la ventana de propiedades picture; esta opción abre una caja de dialogo para seleccionar el origen de la imagen como se observa en las dos siguientes gráficas.



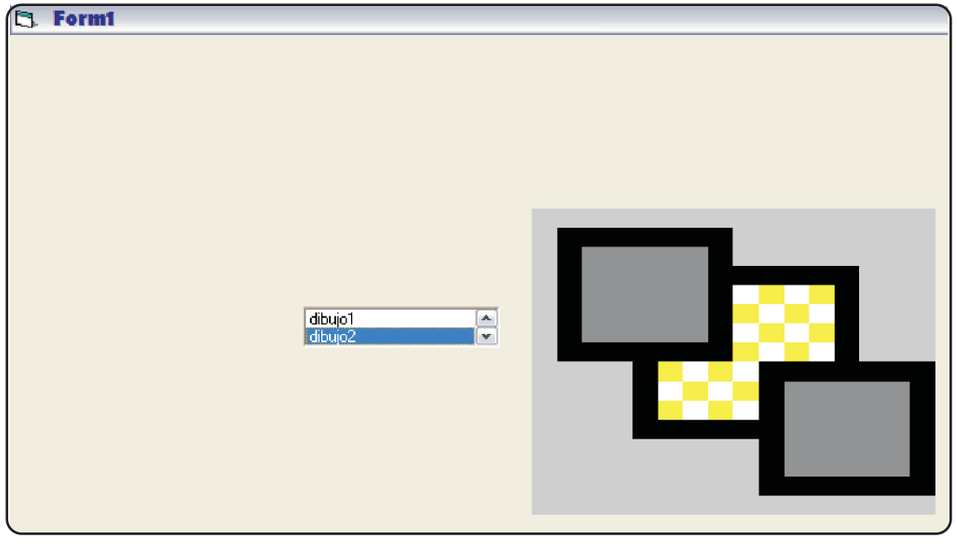
A continuación damos doble clic sobre el listbox1 y escribimos el siguiente código:

```
Private Sub List1_Click()
Select Case List1.ListIndex
Case 0
Image4.Picture = Image1.Picture
Case 1
Image4.Picture = Image2.Picture
Case 2
Image4.Picture = Image3.Picture
End Select
End Sub
```

Inicio de subrutina

Para este ejercicio utilizamos la estructura "Select Case" o selector de casos, ya que esta nos permite cambiar las propiedades picture de los image para visualizar la imagen seleccionada por el usuario

Fin de subrutina

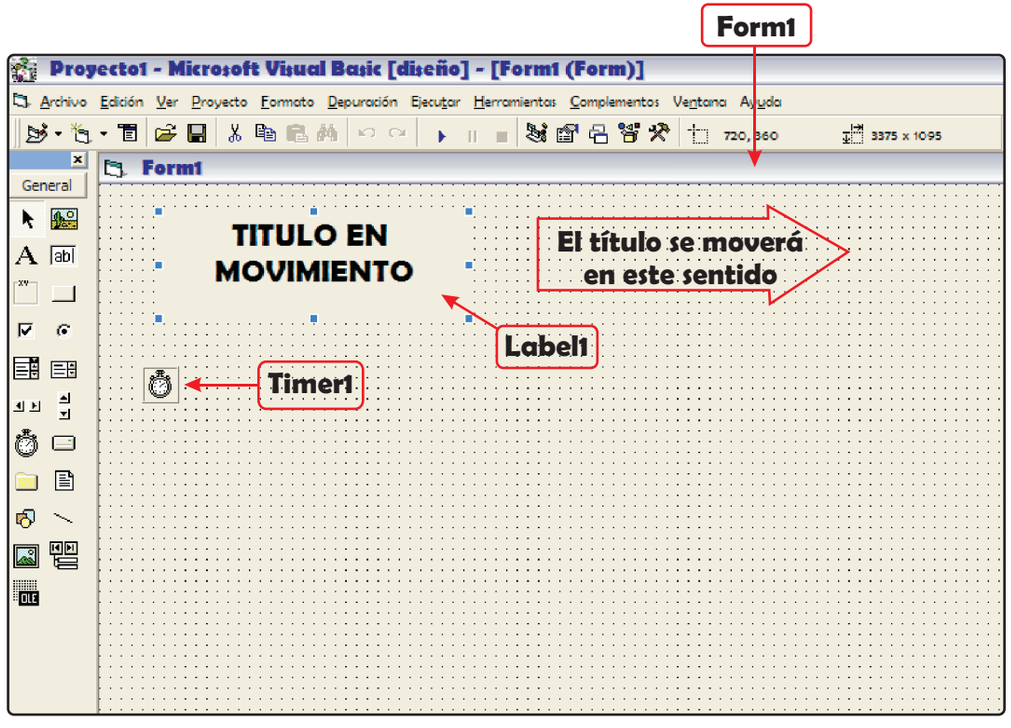


Esta sería la aplicación final del programa; como nos podemos dar cuenta los image1, image2 e image3 no son visibles al usuario, ya que a la propiedad visible de cada una de ellos se encuentra seleccionada en "false".

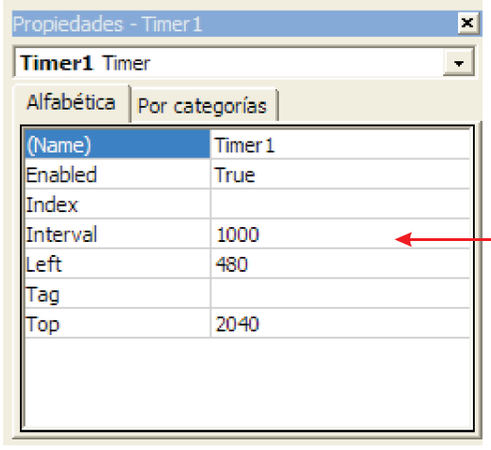
Ejercicio 2

A continuación programaremos movimientos horizontales y verticales de un objeto, para nuestro ejemplo lo haremos con un label.

Los componentes o herramientas a utilizar para este ejercicio son: Form1, Label1 y Timer1.



En este ejercicio la programación la vamos a hacer sobre el timer1, la idea básica es que en un intervalo de tiempo el objeto realice un movimiento progresivo hacia la derecha (+800), y cuando llegue a un lugar específico (9000) en pantalla vuelva e inicie su recorrido horizontal desde 0. Pero no hay que olvidar que debemos seleccionar la propiedad "Interval" del timer1 y colocar un valor para nuestro caso este será de 1000. Observemos esto en la siguiente gráfica.



El valor en la propiedad interval puede cambiar; con este valor ya sea menor o mayor que mil el movimiento puede ser mas rápido o lento

Para escribir el siguiente código damos doble clic sobre el timer1:

```
Private Sub Timer1_Timer()
If Label1.Left < 9000 Then
Label1.Left = Label1.Left + 800
Else
Label1.Left = 0
End If
End Sub
```

Para lograr un movimiento vertical ya no usamos la propiedad "left" sino "Top" y el código sería el siguiente:

```
Private Sub Timer1_Timer()
If Label1.Top < 9000 Then
Label1.Top = Label1.Top + 200
Else
Label1.Top = 0
End If
End Sub
```

Puedo combinar estos códigos para lograr un movimiento combinado (en diagonal)

Ejercicio 3

En este ejercicio utilizaremos una matriz o un array de controles, que básicamente son una serie de controles que comparten los mismos procedimientos o códigos de programación, se distingue uno de otro por el index (list1(1), list1(2)) o un número entre paréntesis que encontramos después del nombre del control en la propiedad "name".

La aplicación que haremos en Visual Basic para entender el procedimiento de un array de controles o matriz es una simulación de una maquina de escribir o un procesado de texto muy sencillo.

Los componentes o herramientas a utilizar serán las siguientes: form1 ,Textbox1, botón de comando 1, botón de comando 2 y botón de comando 3.

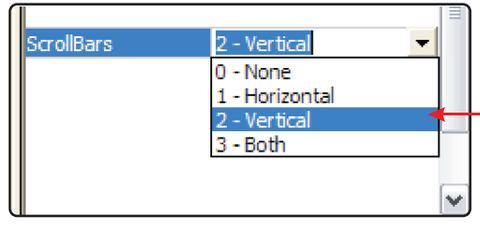
Para tener en cuenta:

•En el textbox1 es en donde vamos a visualizar lo que se escriba, por tanto debemos utilizar las propiedades:

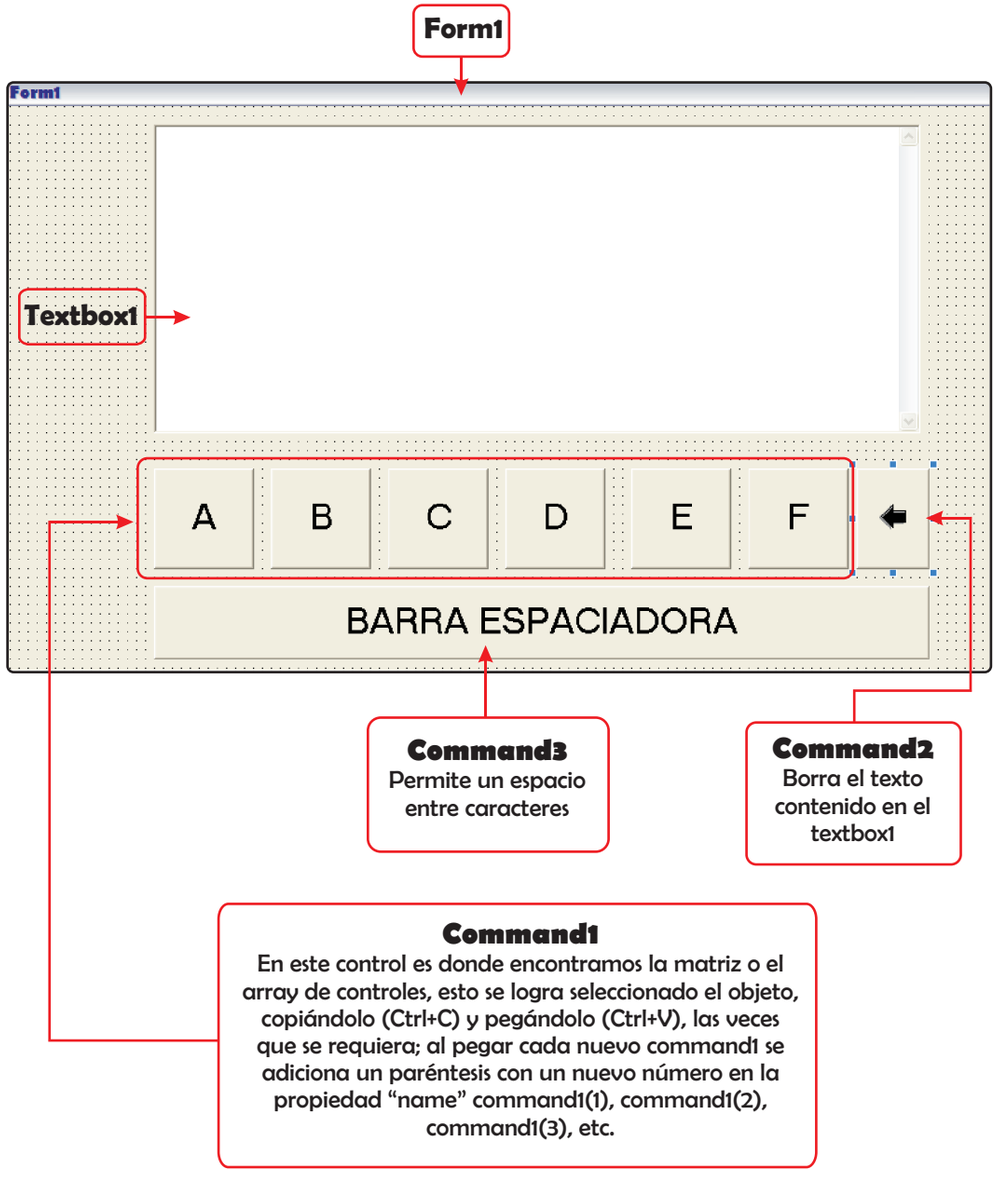
1. Font para seleccionar el tamaño y tipo de fuente en que queremos visualizar.

2.Multiline en la opción True que es la que me permite visualizar varios renglones de texto.

3.Scrollbars en la opción (2) vertical para poder desplazar el texto contenido.



Estas son las opciones que encuentro en la propiedad scrollbars



El código de programación es el siguiente:

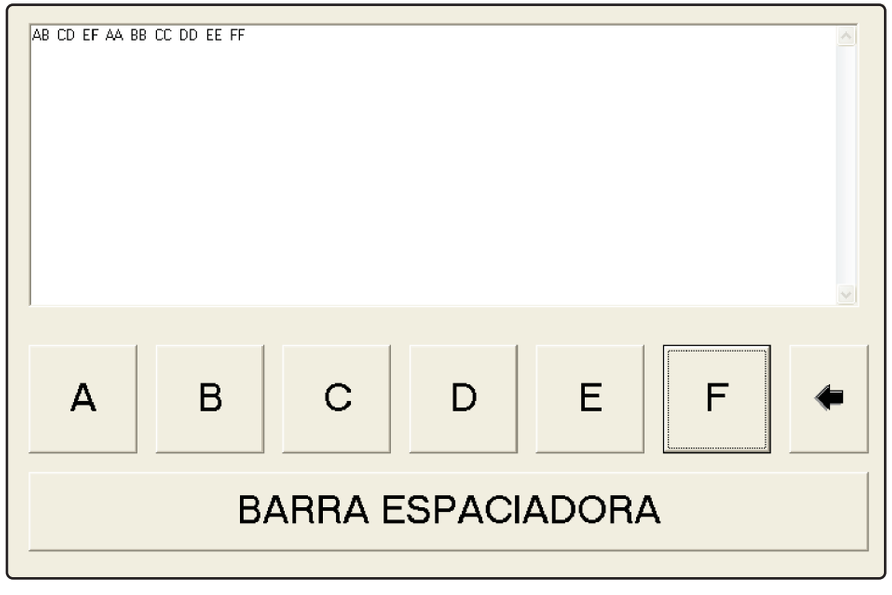
```
Private Sub Command1_Click(Index As Integer)
Text1.Text = Text1.Text + Command1(Index).Caption
End Sub
```

```
Private Sub Command2_Click()
Text1.Text = ""
End Sub
```

```
Private Sub Command3_Click()
Text1.Text = Text1.Text + " "
End Sub
```

La programación de este command1(Index) es la que se aplica al array de controles; La cual consiste en que se debe escribir en el textbox1 el valor contenido en la propiedad caption (letra) de cada uno de los command1( ) (teclas)

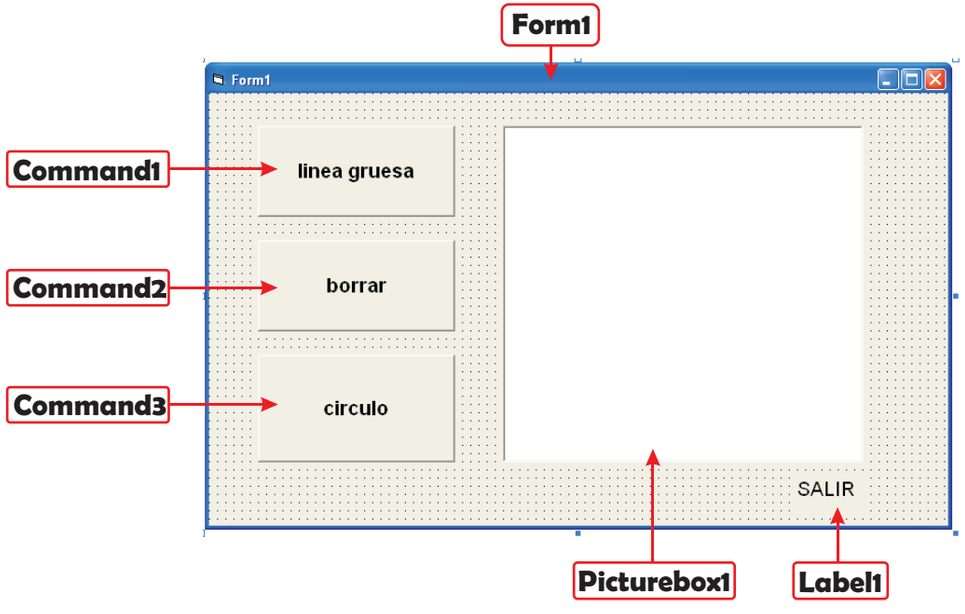
Recordemos que las teclas para escribir las letras funcionan dando clic con el mouse sobre cada una de ellas y esta sería su apariencia final.



Ejercicio 4

En este ejercicio trabajaremos algo de programación gráfica, creando una pizarra muy elemental de dibujo.

Utilizaremos los siguientes componentes o herramientas de Visual Basic: Form1, Picturebox1, Command1, Command2, Command3, Label1.



Para colocar los nombres de cada uno de los botones de comando (Command1, Command2, Command3) utilizamos la propiedad "caption"

Para crear las variables a utilizar, damos doble click sobre el formulario y seleccionamos de la parte superior la opción GENERAL / DECLARACIONES y escribimos:

```
Dim cx As Single
Dim cy As Single
Dim linea
Dim punto
```


Ahora damos doble clic sobre el picturebox1y abrimos la ventana de código, debemos escoger el evento Mouse down y escribimos el siguiente código:

```
Private Sub Picture1_MouseDown(Button As Integer, Shift As Integer, X As
Single, Y As Single)
If punto = 1 Then
Picture1.PSet (X, Y)
Else
Picture1.Line -(X, Y)
End If
If Button = 1 Then
cx = X: CurrentX = X
cy = Y: CurrentY = Y
End If
End Sub
```

A continuación escribimos el siguiente código para los command1, Command2, Command3 y Label1 que va a ser la opción salir del programa:

```
Private Sub Command1_Click()
Picture1.DrawWidth = 2
End Sub
```

Con el comando DrawWidth se define el grosor de la linea.

```
Private Sub Command2_Click()
Picture1.Cls
End Sub
```

Con el comando CLS (ClearScreen) se borra el contenido del picturebox1.

```
Private Sub Command3_Click()
Picture1.Circle (cx, cy), 300
End Sub
```

Con el comando Circle dibujamos el círculo en las coordenadas cx y cy del picturebox1, el valor 300 es el tamaño del círculo.

```
Private Sub Form_Load()
Picture1.MousePointer = 4
End Sub
```

Con el comando MousePointer seleccionamos el puntero del Mouse a utilizar.

```
Private Sub Label1_Click()
End
End Sub
```

Con el comando End se cierra la aplicación o programa.