CONTENIDO

REFERENCIAS

- **Mastering Visual Basic 6 Fundamentals**
  Microsoft Official Curriculum (M.O.C)
- **Enciclopedia de Visual Basic**
  Editorial: RAMA; Autor: Ceballos, Javier;
- **MSDN Library Visual Studio 6.0**
- **Página Principal de Visual Studio:** www.microsoft.com/spanish/msdn/vstudio/default.asp
- **Visual Basic Developer Center:** msdn.microsoft.com/vbasic/
- **Mastering Microsoft:** msdn.microsoft.com/mastering/
  msdn.microsoft.com/developer
  www.microsoft.com
- **Portal de Programación:** www.lawebdelprogramador.com
- **Página Web:** www.programacion.com
- **Página Web:** www.elguille.com
- **Página Web:** www.allapis.com

Laboratorio No. 1: Creando una aplicación sencilla

En este ejercicio, aprenderá como crear una aplicación sencilla en Visual Basic.

Objetivos

- Identificar los elementos del entorno de desarrollo de Visual Basic
- Agregar controles al formulario
- Definir las propiedades de los controles
- Crear una aplicación ejecutable

Ejercicio No. 1: Usando Controles

➢ **Iniciando Visual Basic**

1.- Seleccione Standard.exe, y haga clic en **Abrir** para comenzar con un nuevo proyecto.

Un proyecto nuevo con una forma será creado.

Agregue controles al formulario

1.- Seleccione un control en la barra de controles.
2.- Ubique el puntero sobre cualquier área del formulario
3.- Presione el botón izquierdo del mouse mientras arrastra el puntero hacia cualquier posición deseada dentro del formulario.
4.- Repetir el proceso con un número diferente de controles en la barra de controles.

Reciba ayuda con los controles

1.- Seleccione el control agregado previamente dentro del formulario.
2.- Presione F1, la ayuda Visual Basic se abre con la explicación del control seleccionado.
3.- Examine la información sobre el control, y cierre la ventana de ayuda.

Ejecute la aplicación

1.- En el menú **Ejecutar,** haga clic sobre **Iniciar**, o haga clic sobre el botón  en la barra de herramientas. Visual Basic debería abrir el formulario ejecutándose. Recuerde que puede utilizar la tecla de función **F5**
2.- Pruebe el control en el formulario.
3.- Cierre la aplicación y regrese al modo diseño, usando uno de estos métodos :

a.- En la barra de herramienta, haga clic en el botón  .
b.- En el menú **Ejecutar**, haga clic sobre **Terminar.**
c.- En la barra de titulo, haga clic en el botón 

Remover el control de la forma

1.- Mueva el puntero del mouse sobre el control.
2.- Seleccione con el botón izquierdo del mouse. Se muestra varios cuadritos de color, indicando que el control ha sido seleccionado.
3.- En el menú **Edición**, haga clic sobre **eliminar**, o presione la tecla Suprimir. Todos los controles que estén seleccionados serán borrados del formulario.

Ejercicio No. 2: Creando una aplicación con código.

➢ **Agregando un cuadro de Texto (TextBox) y una Etiqueta(Label) al formulario y alinéelos horizontalmente.**

1. Agregue un cuadro de texto al formulario
2. Agregue una etiqueta al formulario, ubicándola al lado del cuadro de texto.
3. Haga clic sobre el formulario y arrastre el puntero del mouse alrededor de los 2 controles, con la finalidad de seleccionar ambos controles recientemente creados.
4. Seleccione un control en el formulario para alinearlo con el otro control.
5. En el menú **Formato**, seleccione **Alinear** y entonces seleccione **Medio** para alinear al centro ambos controles.

- **Defina las propiedades del control en modo de diseño**

1. Haga clic sobre el formulario para deseleccionar los controles, y seleccione el control etiqueta en el formulario.
2. En la ventana de propiedades, cambie la propiedad **Caption** a "MiEtiqueta".
3. Seleccione el Cuadro de Texto.
4. En la ventana de propiedades, cambie la propiedad **Text** a "MiCuadroTexto".

- **Cambie las propiedades del control en tiempo de ejecución.**

1. Haga doble clic en el cuadro de texto.

   La venta del editor de código se abre mostrando el procedimiento del evento **text1_Change()**.

2. Agregue la siguiente línea de código al procedimiento :

```
Label1.caption = Text1.text
```

   Este código cambia el **caption** del control etiqueta al texto que el usuario tipea en el cuadro de texto.

3. Ejecute la aplicación y tipee algún texto en el cuadro de texto. ¿Que sucede? Comparta con la clase sus impresiones.
4. Cierre la aplicación y regrese al modo de diseño.

- **Guarde el Proyecto**

1. En el menú **Archivo**, seleccione **Guardar Proyecto**.
2. Cuando Visual Basic le pregunte si desea guardar el formulario, cambie la carpeta de destino a C:\VisualBasic\Laboratorio1, mantenga el nombre **form1** y presione guardar.
3. Cuando Visual Basic le pregunte si desea guardar el proyecto, cambie la carpeta de destino a C:\VisualBasic\Laboratorio1, mantenga el nombre **Proyecto1** y presione guardar.

- **Crear un archivo ejecutable (.EXE)**

1. En el menú **Archivo**, seleccione **Generar Proyecto1.exe** para hacer el archivo ejecutable.
2. En el cuadro de texto asigne un nombre al archivo, escriba **MiPrimeraAplicación** y entonces presione **Aceptar**. Visual Basic compila el proyecto actual y crea un archivo ejecutable que puede ser directamente ejecutado desde Windows.
3. Salga de Visual Basic y guarde cualquier cambio.

- **Crear un archivo ejecutable (.EXE)**

1. Presione el botón **Inicio**, Seleccione **Programas**, ejecute **El Explorador de Windows**. Y busque el programa recientemente compilado en la carpeta C:\VisualBasic\Laboratorio1, Si el programa corre sin problemas, ** ***Felicitaciones, siga adelante, lo esta haciendo muy bien*** **
2. Ejecute la aplicación recientemente creada haciendo doble clic en el archivo ejecutable **MiPrimeraAplicacion**.
3. Pruebe la aplicación escribiendo algún texto en el cuadro de texto.
4. Cierre la aplicación.

Ejercicio No. 3: Usando el código del archivo ayuda de MSDN.

- **Abra el proyecto Proyecto1.**

1. Inicie Visual Basic, si no esta iniciado.
2. Seleccione la pestaña reciente, seleccione Proyecto1 y presione **Abrir**.
3. Si el formulario no está visible, selecciónelo en el Explorador de Proyectos y presione sobre ver objetos.

- **Use el contenido de la ayuda MSDN.**

1. Haga doble clic en cualquier área del formulario, fuera del Cuadro de texto y de la etiqueta.
   La siguiente línea de código aparecerá en la ventana del Editor de Código:

   ```
   PRIVATE SUB FORM_LOAD()

   END SUB
   ```

2. En la ventana del Editor de Código, seleccione **MouseMove** en la caja de lista de Procedimientos. *"El combo de lista ubicado a la derecha dentro del editor de código"*

   El contenido de la ventana del editor de código cambia a:

   Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)

   End Sub

3. Escriba **FillColor** entre las líneas **Sub** y **End Sub**.
4. Seleccione con el puntero del mouse la palabra **FillColor**, y presione F1.

   La ayuda MSDN se desplegará con la información sobre las propiedades FillColor.

- **Copie el código ejemplo de la ayuda MSDN y péguelo dentro de la aplicación**

1. Haga clic sobre el saltador de tópicos de ejemplos en la parte superior de la ventana de ayuda.

   Una ventana de ayuda se abrirá con un ejemplo de cómo usar la propiedad **FillColor**.

2. Seleccione las líneas de código entre las palabras Sub y End Sub.
3. Presione el botón derecho del mouse en cualquier lugar sobre el texto seleccionado y proceda a copiar para almacenar en el portapapeles.
4. Cierre la ventana de ayuda del MSDN.
   Usted debería estar ahora en la ventana del editor de código, desplegando el procedimiento del evento **Form_MouseMove**.Si no, en la ventana del Editor de Código, seleccione **Form** en la lista de Objetos y **MouseMove** en la lista de Procedimientos para moverse hasta ese procedimiento.

5. Sombree el comando **FillColor**.
6. En el menú **Edición**, seleccione **pegar**, esto instantáneamente hará aparecer el código que tenemos en el portapapeles dentro del procedimiento **Form_MouseMove**. El Resultado se debería ver exactamente como se ilustra a continuación :

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)

 FillColor = QBColor(Int(Rnd * 15)) ' Elige un valor aleatorio de ' FillColor. FillStyle = Int(Rnd * 8) ' Elige un valor aleatorio de FillStyle. Circle (X, Y), 250 ' Dibuja un círculo.


End Sub
```

- **Guarde y prueba su aplicación**

1. En el menú **Archivo**, seleccione **Guardar Proyecto** para guardar los cambios del proyecto y el formulario.
2. Ejecute la aplicación y mueva el puntero del mouse alrededor del formulario. ¿Qué sucede? Comparta sus impresiones con el resto de la clase.
3. Cierre la aplicación.

LABORATORIO No. 2: Creando una aplicación Visual Basic

Objetivos

Después de completar este laboratorio, estará en capacidad de:

- Crear una sencilla aplicación en Visual Basic
- Crear un procedimiento de evento

Ejercicio No. 1: Crear una pantalla Logon

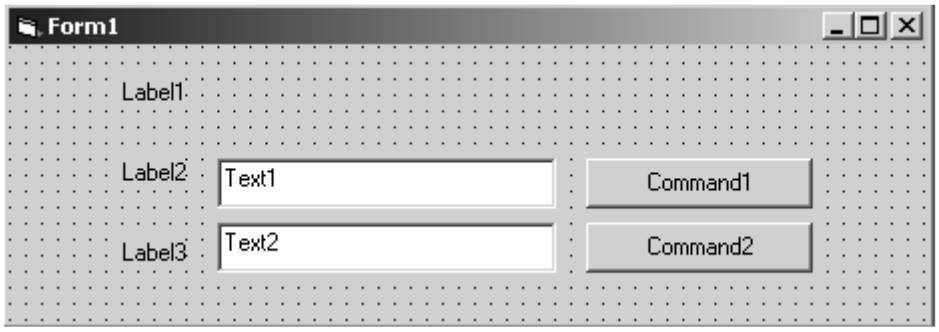
En este ejercicio, creará una pantalla de Logon para una aplicación. Comenzará un nuevo proyecto, cambiará el nombre al formulario, colocará controles en el formulario y definirá las propiedades de los controles.

➢ **Comience un nuevo proyecto.**

1. Ejecute Visual Basic (si no se está ejecutando). En el menú **Archivo**, seleccione **Nuevo Proyecto**.
2. Seleccione **Standard EXE**, and presione **Aceptar**.

➢ **Agregar controles al formulario por defecto**

1. Agregue 3 etiquetas al formulario.
2. Agregue 2 cuadro de textos al formulario.
3. Agregue 2 botones de comando (Command Button) al formulario.
4. Mueva y redimensione los controles con la finalidad de que se vean como en la siguiente ilustración :



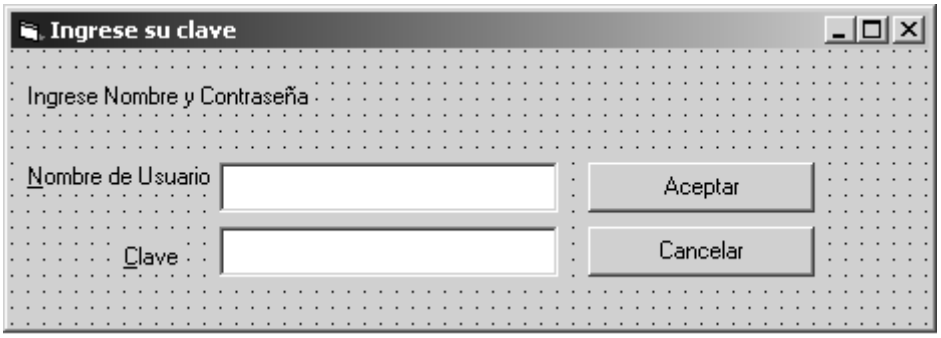
➢ **Definir las propiedades para cada uno de los controles**

1. Seleccione cualquiera de los controles en el formulario.
2. En la ventana de propiedades, seleccione la propiedad del control que este listada en la tabla siguiente.
3. Ingrese el nuevo valor para la propiedad que se especifica en la tabla.

Objeto Actual	Propiedad	Nuevo Valor
Label1	Name	lblInstrucciones
	Caption	Ingrese Nombre y Contraseña
Label2	Name	lblNombreUsuario
	Caption	&Nombre de Usuario
Label3	Name	lblpassword
	Caption	&Clave
Text1	Name	txtNombreUsuario
	Text	En blanco
Text2	Name	txtClave
	Text	En blanco

	PasswordChar	*
Command1	Name	cmdAceptar
	Caption	Aceptar
	Default	True
Command2	Name	cmdCancelar
	Caption	Cancelar
	Cancel	True
Form1	Name	frmLogon
	Caption	Ingrese su clave
	BorderStyle	1 – Fixed Single

El resultado de esto deber parecerse a la siguiente ilustración:



- **Agregar código al evento clic del botón cmdAceptar**

1. Haga doble clic sobre el botón de comando (command button) cmdAceptar. Para abrir la ventana del editor de código con el siguiente código insertado :

```
Private Sub cmdAceptar_Click()

End Sub
```

2. Agregue código al procedimiento del evento click para desplegar un mensaje con los valores actuales de la caja de texto (Text Box) Nombre y Clave :

```
MsgBox "Nombre de Usuario: " & TxtNombreUsuario.text & ", Clave: "
& txtClave.text
```

3. Ejecute la aplicación. Escriba algo sobre el cuadro de texto del nombre de usuario y de la clave, y presione **ACEPTAR**. ¿Qué sucede?.
4. Escriba un nombre diferente de usuario y una clave diferente, presione **ENTER**. ¿Qué sucede?, Intercambie sus impresiones con la clase.
5. Cierre la aplicación y regrese al modo diseño.

- **Agregar código al evento clic del botón cmdCancelar**

1. Haga doble clic sobre el botón de comando (command button) cmdCancelar. Para abrir la ventana del editor de código con el siguiente código insertado :

```
Private Sub cmdCancelar_Click()

End Sub
```

2. Agregue código al procedimiento del evento click para desplegar un mensaje :

```
MsgBox "Este botón cancela la acción"
```

3. Ejecute la aplicación. Escriba algo sobre el cuadro de texto del nombre de usuario y de la clave, y presione **CANCELAR**. ¿Qué sucede?
4. Escriba un nombre diferente de usuario y una clave diferente, presione la tecla **ESCAPE**. ¿Qué sucede?, Intercambie sus impresiones con la clase.
5. Cierre la aplicación y regrese al modo diseño.

- **Asignar nombre a nuestro proyecto**

1. En el menú **Proyecto**, seleccione **Propiedades** del proyecto1.
2. En el nombre del proyecto, escriba **MiPrimerProyecto**, y presione **Aceptar**.

- **Guardar el proyecto**

1. En el menú **Archivo**, seleccione **Guardar proyecto**.
2. Cuando Visual Basic le pregunte si desea guardar el formulario, cambie la carpeta de destino a C:\VisualBasic\Laboratorio2, asigne el nombre **frmLogon** y presione guardar.
3. Cuando Visual Basic le pregunte si desea guardar el proyecto, cambie la carpeta de destino a C:\VisualBasic\Laboratorio2, mantenga el nombre **MiPrimerProyecto** y presione guardar.

Ejercicio No. 2: Agregar código enabled (Habilitado) y Disabled (Deshabilitado) a un botón

En este ejercicio, agregará código que habilitará el botón aceptar, solo cuando el nombre del usuario y la clave hayan sido ingresados en los cuadros de texto.

Cuando se diseña una aplicación, se debe minimizar la posibilidad de que los usuarios comentan errores. Una forma de hacer esto es, deshabilitando los controles que al ser seleccionados pueden causar algún error.

- **Habilitar el botón Aceptar.**

1. En la ventana de propiedades, defina la propiedad **Enabled** del botón **cmdAceptar** a **Falso/False**.
2. Haga clic en el cuadro de texto **TxtNombreUsuario**.
   Se abre la ventana del editor de código para el procedimiento del evento **txtNombreUsuario_Change**.
3. Agregue las siguientes líneas de código al evento **Change** del txtNombreUsuario :

```
If txtnombreUsuario.text <> "" And txtClave.text <>"" then
      cmdAceptar.enabled = true
Else
      cmdAceptar.enabled = false
end if
End Sub
```

4. Copie el código del evento txtNombreUsuario_Change() hacia el evento txtclave_change() del cuadro de texto txtClave.
5. Guarde y pruebe la aplicación. ¿Esta el botón Aceptar habilitado?
6. Escriba el nombre del usuario. ¿Esta el botón Aceptar habilitado?,¿Porque?, Comparta sus impresiones o dudas con la clase.
7. Escriba la clave del usuario. ¿Esta el botón Aceptar habilitado?,¿Porque?, Comparta sus impresiones o dudas con la clase.

LABORATORIO No. 3
Implementando formularios y manejo de Datos con Archivos Planos

Objetivos
Copie todos los archivos del Laboratorio2 en la carpeta de Laboratorio3 y comience a trabajar Después de completar este laboratorio, estará en capacidad de:

- Agregar un nuevo formulario a un proyecto existente.
- Cambiar el formulario de inicio de un proyecto.
- Cargar y desplegar un formulario desde otro funcionario.
- Validar diferentes tipos de datos en un formulario
- Manipular Archivos planos con Visual Basic
- Optimizar el código de programación / tip's / mejores practicas de desarrollo
- Implementar Funciones y Procedimientos dentro de un Proyecto
- Uso de Variables y Constantes

Ejercicio No. 1: Agregar el formulario principal

En este ejercicio, agregará un formulario, frmPrincipal, al proyecto. Este formulario será el principal en la aplicación. Por ahora frmPrincipal será usado para cargar el frmLogon.

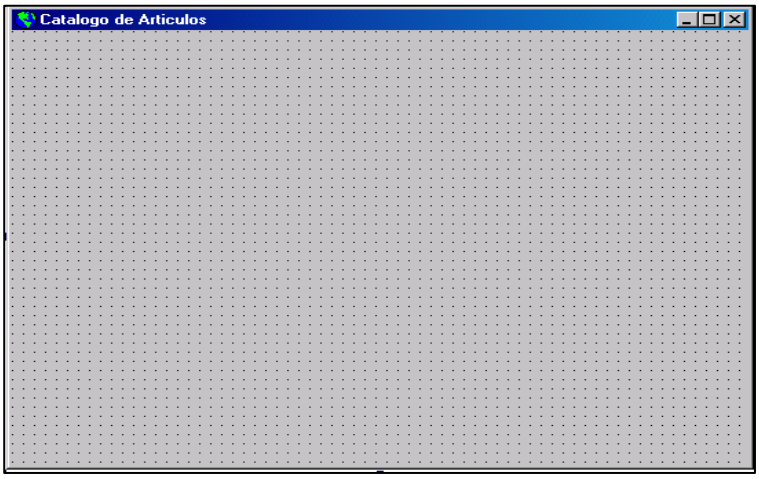
- **Abrir el proyecto MiPrimerProyecto.**
- **Agregar un nuevo formulario al proyecto.**

1. En el menú **Proyecto**, seleccionar agregar formulario.
2. En el cuadro de dialogo, **Agregar Formulario**, seleccione el formulario y entonces haga clic sobre **Abrir**.

Un nuevo formulario debería ser agregado al proyecto y desplegado en el entorno Visual Basic.

3. Defina las propiedades del formulario que se presentan a continuación :

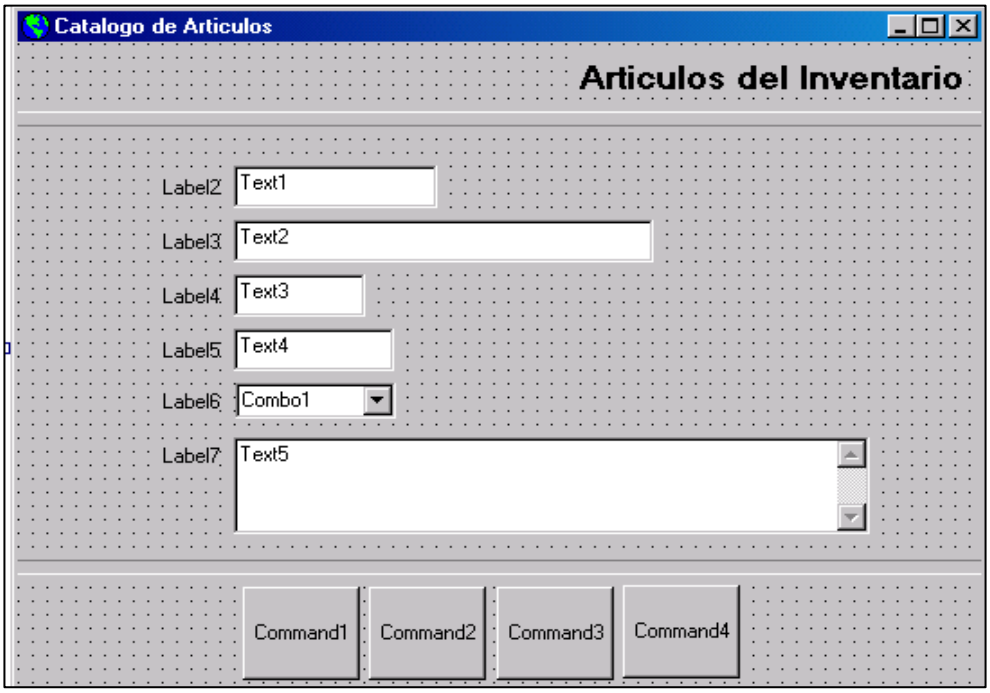
Propiedades	Definiciones
(Name)	Frmarticulos
Caption	Catalogo de Articulos
Border Style	1 – Fixed Single
StarUpPosition	CenterScreen
Icon	**...** \Visualbasic\Iconos



Ejercicio No. 2: Creando la interfaz del usuario.

- **Agregando los controles al formulario frmarticulos**

1. Agregue las etiquetas (label), cuadros de texto (textbox), combo de selección (combobox) y botones de comando (CommandButton) como se muestran en la siguiente ilustración:

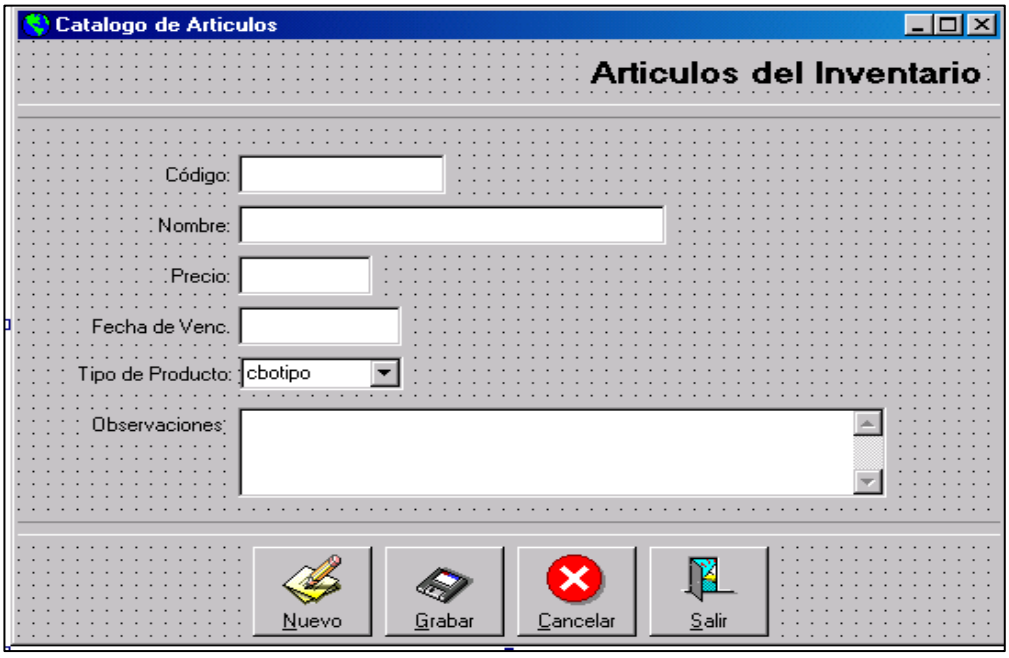


2. Definir las propiedades de los objetos como se presentan a continuación:

Objeto	Propiedad	Nuevo Valor
Label1	Name	Lbltitulo
	Caption	Articulos del Inventario
	Autosize	True
	Font	**...** MS Sans Serif / Negrita / 14
Label2	Name	Lblcodigo
	Caption	Código_
Label3	Name	Lblnombre
	Caption	Nombre:
Label4	Name	Lblprecio
	Caption	Precio:
Label5	Name	Lblfechaven
	Caption	Fecha de Venc.:
Label6	Name	Lbltipo
	Caption	Tipo de Producto:
Label7	Name	Lblobservaciones
	Caption	Observaciones
Text1	Name	Txtcodigo
	Text	En blanco
Text2	Name	Txtnombre
	Text	En Blanco
Text3	Name	Txtprecio
	Text	En blanco

Text4	Name	Txtfecha_ven
	Text	En blanco
Combo1	Name	Cbotipo
	List	Nacional (*usar Ctrl+Enter)*
		Importado
	Style	Dropdown Combo
	Text	En Blanco
Text5	Name	Txtobservaciones
	Text	En blanco
	Multiline	True
	ScrollBars	2.- Vertical
Command1	Name	Cmdnuevo
	Caption	&Nuevo
	Style	1- Graphical
	Picture	(Icono) **...**
Command2	Name	Cmdgrabar
	Caption	&Grabar
	Style	1 – Graphical
	Picture	(Icono) ...
Command3	Name	Cmdcancelar
	Caption	&Cancelar
	Style	1 – Graphical
	Picture	(Icono) ...
Command4	Name	Cmdsalir
	Caption	&Salir
	Style	1 – Graphical
	Picture	(Icono) ...
	Cancel	Trae

El resultado de esto deber parecerse a la siguiente ilustración:



- **Cambiar el formulario de inicio**

1. Guarde el formulario con el nombre frmarticulos en la carpeta Laboratorio3
2. En el menú **Proyecto**, seleccione propiedades de **MiPrimerProyecto**.
3. En la lista de los objetos de inicio, en la pestaña general, seleccione **frmlogon**, presione Aceptar.

- **Guardar y probar la aplicación**

1. Guardar el Proyecto.
2. Ejecute la aplicación. Escriba un nombre de usuario y clave, presione aceptar. ¿Qué sucede? ¿Por qué no desaparece el formulario Logon?.Comparta sus impresiones con el resto de la clase. ¿Lo lógico es que desaparezca el formulario de logon y se muestre el de artículos ?
3. Cierre la aplicación y regrese al modo diseño.

Ejercicio No. 3: Modificando el código del Formulario frmlogon de la aplicación.

- **Finalice la aplicación cuando el usuario presione Cancelar en frmLogon o Salir en frmPrincipal.**

1. Agregue código al procedimiento del evento clic del botón cmdcancelar del formulario frmLogon para finalizar la aplicación :

```
Unload me
End
```

2. Agregue código al procedimiento del evento clic del botón cmdSalir del formulario **frmarticulos** como se muestra a continuación:

```
Dim vresp As Integer
vresp = MsgBox("Desea Salir del Sistema", vbQuestion + vbYesNo, "Sistema de Inventario")
If vresp = vbYes Then
  cmdSalir.Tag = "PRESIONADO"
  Unload Me
End If
```

3. Agregue código al procedimiento del evento Unload del formulario frmarticulos para finalizar la aplicación :

```
If cmdSalir.Tag = "PRESIONADO" Then
   Cancel = False
Else
  Dim vresp As Integer
  vresp = MsgBox("Desea Salir del Sistema", vbQuestion + vbYesNo, "Sistema de Inventario")
  If vresp = vbYes Then
     Cancel = False
  Else
     Cancel = True
  End If
End If
```

- **Guarde y pruebe la aplicación :**

1. Guarde el proyecto.

2. Ejecute la aplicación. Escriba el nombre del usuario y la clave. Presione **Aceptar**. ¿Qué sucede?
3. Ahora presione **Salir**. ¿Qué sucede?. Comparta sus impresiones y/o dudas con el resto de los participantes.
4. Cierre la aplicación y regrese al modo diseño.

Ejercicio No. 4: Escribiendo Código en el Formulario

En este ejercicio, agregará código para aperturar el archivo plano en los diferentes modos (For Append, for Input) agregar la funcionalidad a los botones de comando del formulario. Escribir funciones y Procedimientos para reducir el tamaño del código y optimar los proyectos de desarrollo.

- **Declarando Constantes a nivel General del Formulario.**

1. Dentro del editor de código de **frmarticulos**, en la sección de (General) / (Declaraciones) agregar el código que a continuación se presenta:

```
'declarando una constante publica a nivel del formulario
Const varchivo = "c:\productos.txt"
```

Nota: Pruebe agregar una constante para el manejo de los títulos mostrados con la instrucción msgbox para personalizar su aplicación. (Opcional)

- **Programando el evento load del formulario.**

```
Private Sub Form_Load()
  'abrir el archivo para lectura/escritura/inserción
  Open varchivo For Append As #1
End Sub
```

2. Modifique los procedimientos cmdsalir_click() y form_unload() y agregue la instrucción para cerrar el archivo de texto:

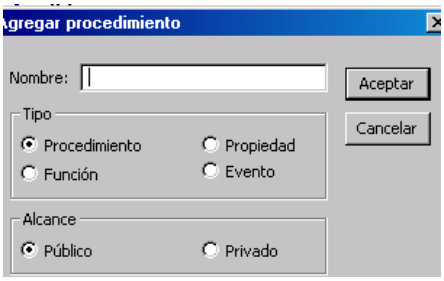
```
Close #1 'cerrando el archivo plano
```

3. Ejecute la Aplicación, pruebe si todo esta funcionando bien.

➢ Programando Procedimientos y Funciones

1. Desde el editor de código, en el menú **Herramientas**, seleccione **Agregar Procedimientos**.
2. En la caja de dialogo, coloque el nombre y el tipo (Función / Procedimiento / Propiedad / Evento) y presione **Aceptar**

Con Los procedimientos y funciones definidas por el programador se pueden reducir muchas líneas de código y mejorar el rendimiento de las aplicaciones. Una de las ventajas de utilizarlas radica en el hecho de que pueden ser llamadas de diferentes parte de un programa.



- **Programando procedimientos dentro del formulario.**

```
Sub plimpiar_campos()
 'procedimiento que limpia los campos del formulario
  txtcodigo.Text = ""
  txtnombre.Text = ""
  txtprecio.Text = ""
  txtfecha_ven.Text = ""
  cbotipo.Text = ""
  txtobservaciones.Text = ""
End Sub
```

- **Programando funciones dentro del formulario.**

```
Function fbotones(vargumento As String)
 'apagar/encender los botones de la barra de herramientas
  If vargumento = "PRENDER" Then
     cmdnuevo.Enabled = True
     cmdSalir.Enabled = True
     cmdgrabar.Enabled = False
     cmdcancelar.Enabled = False
  Else
     cmdnuevo.Enabled = False
     cmdSalir.Enabled = False
     cmdgrabar.Enabled = True
     cmdcancelar.Enabled = True
  End If
End Function

Function fhabilitar_campos(vargumento As Boolean)
'función que habilitar/deshabilita los campos segun el argumento
'recibido por la función
   txtcodigo.Enabled = vargumento
   txtnombre.Enabled = vargumento
   txtprecio.Enabled = vargumento
   txtfecha_ven.Enabled = vargumento
   cbotipo.Enabled = vargumento
   txtobservaciones.Enabled = vargumento
End Function
```

- **Programando los botones de comando y Ajustar el código del formulario**

```
Private Sub Form_Load()
  'abrir el archivo para lectura/escritura/insercion
  Open varchivo For Append As #1
  fbotones ("PRENDER")
  fhabilitar_campos (False)
End Sub

Private Sub cmdnuevo_Click()
 Call plimpiar_campos
 fhabilitar_campos (True)
 fbotones ("APAGAR") 'apagar los botones y prender Grabar y Cancelar
 txtfecha_ven.Text = Date
 txtcodigo.SetFocus
End Sub
```

```
Private Sub Form_Unload(Cancel As Integer)
If cmdSalir.Tag = "PRESIONADO" Then
   Cancel = False
Else
  Dim vresp As Integer
  vresp = MsgBox("Desea Salir del Sistema", vbQuestion + vbYesNo, "Sistema de Inventario")
  If vresp = vbYes Then
     Close #1 'cerrar el archivo plano
     Cancel = False
  Else
     Cancel = True
  End If
End If
End Sub

Private Sub cmdgrabar_Click()
     Write #1, txtcodigo.Text, txtnombre.Text, txtprecio.Text, txtfecha_ven.Text, cbotipo.Text, txtobservaciones.Text
       MsgBox "Registro Agregado Satisfactoriamente..", vbExclamation + vbDefaultButton1, "Sistema de Inventario"
       plimpiar_campos
       fbotones ("PRENDER") 'apagar todos los botones y encender nuevo y salir
       fhabilitar_campos (False)
End Sub

Private Sub cmdcancelar_Click()
  Call plimpiar_campos
  fbotones ("PRENDER")
  fhabilitar_campos (False)
End Sub

Private Sub cmdSalir_Click()
     Dim vresp As Integer
     vresp = MsgBox("Desea Salir del Sistema", vbQuestion + vbYesNo, "Sistema de Inventario")
     If vresp = vbYes Then
        Close #1 'cerrar el archivo plano
        cmdSalir.Tag = "PRESIONADO"
        Unload Me
     End If
End Sub
```

4. Pruebe el sistema, presione F5 para correr la aplicación, compruebe si todo esta funcionando bien.

Ejercicio No. 5: Validando la Entrada de Datos

En este ejercicio, se dará cuenta que es muy importante verificar los datos contenidos en las cajas de texto (textbox) antes de ser almacenadas en un archivo, base de datos o para su procesamiento posterior. A este punto, consulte con el facilitador cuales son las diferentes maneras de validación de la información, los eventos y funciones utilizadas.

- **Modificando el botón de Grabar**

1. Haz doble clic en el botón de Grabar del formulario **frmarticulos**

La ventana del editor de código se abre, con el procedimiento del evento **Click**.

2. Complete el cuerpo del procedimiento con el código que se muestra a continuación:

```
'**************************************
 'validando los objetos antes de grabar el registro
 '**************************************
 If IsNull(txtcodigo.Text) Or txtcodigo.Text = "" Then
    MsgBox "Debe ingresar un código válido.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
    txtcodigo.SetFocus
    Exit Sub
 End If
 If Not IsNumeric(txtprecio.Text) Then
    MsgBox "Debe ingresar un monto válido.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
    txtprecio.SetFocus
    Exit Sub
 End If
 If Not IsDate(txtfecha_ven.Text) Then
     MsgBox "Debe ingresar una fecha válida.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
    txtfecha_ven.SetFocus
    Exit Sub
 End If
' ********* fin de las validaciones *************
```

3. Pruebe el sistema, presione **F5**, intente grabar un registro dejando campos en blanco, Guarde y pruebe su trabajo.

Ejercicio No. 6: Agregando una Función de Búsqueda

En este ejercicio, agregará una función definida por el programador, cuyo objetivo será de verificar si un código ya existe en el archivo. Es decir, si estamos dentro de un formulario para la inserción de registros, no podemos permitir que un usuario agregue 2 veces el mismo código.

Antes de comenzar, seria interesante visualizar el contenido del archivo plano donde se esta guardando la información de los registros. Recuerde que dicho archivo se encuentra en la ruta c:\productos.txt, utilice el explorador de Windows, y con la ayuda del block de notas revise el contenido del archivo. Antes de continuar recuerde cerrar el notepad.

La lógica de la función a implementar debe abrir el archivo plano para lectura y revisar el contenido línea a línea de string (registro) para luego descomponerlo y extraer el código del producto para luego compararlo con el valor presente en la caja de texto txtcodigo.

- **Programando la función de búsqueda**

1. En la Ventana de Código del formulario frmarticulos, agregue la siguiente función:

```
Function fexiste(pcodigo As String) As Boolean
 Dim vregistro As String, vcodigo As String, i As Integer
 Close #1 'cerrar temporalmente el archivo
 Open varchivo For Input As #1 'abrirlo nuevamente pero como solo lectura
 Do While Not EOF(1)
   Line Input #1, vregistro
   For i = 2 To Len(vregistro) 'recorrer todo el registro
     'determinar la posicion del campo codigo, detectando la ubic. de las comillas dobles "
     If Mid(vregistro, i, 1) = Chr(34) Then Exit For
   Next i
```

```
    vcodigo = UCase(Mid(vregistro, 2, i - 2))
    If UCase(pcodigo) = vcodigo Then
      fexiste = True
      Exit Do
    Else
      fexiste = False
    End If
  Loop
  Close #1 'cerrar como de solo lectura
  Open varchivo For Append As #1 'reabrirlo con modalidad  insertar (append)
End Function
```

2. Modifique el evento **click** del botón Grabar del formulario **frmarticulos** como se muestra a continuación, agregue los cambios que sean necesarios:

```
Private Sub cmdgrabar_Click()
  Dim vexiste As Boolean
  '****************************************
  'validando los objetos antes de grabar el registro
  '****************************************
  If IsNull(txtcodigo.Text) Or txtcodigo.Text = "" Then
     MsgBox "Debe ingresar un código válido.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
     txtcodigo.SetFocus
     Exit Sub
  End If
  If Not IsNumeric(txtprecio.Text) Then
     MsgBox "Debe ingresar un monto válido.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
     txtprecio.SetFocus
     Exit Sub
  End If
 If Not IsDate(txtfecha_ven.Text) Then
      MsgBox "Debe ingresar una fecha válida.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
     txtfecha_ven.SetFocus
     Exit Sub
 End If
' ********* fin de las validaciones *************
  'llamada a la función de validacion
  vexiste = fexiste(txtcodigo.Text)

  If vexiste Then
     MsgBox "Disculpe,, el código ya existe,,<<Verifique>>", vbCritical +
vbDefaultButton1
     txtcodigo.SetFocus
     Exit Sub
  End If
  Write #1, txtcodigo.Text, txtnombre.Text, txtprecio.Text, txtfecha_ven.Text,
cbotipo.Text, txtobservaciones.Text
  MsgBox "Registro Agregado Satisfactoriamente..", vbExclamation + vbDefaultButton1,
"Sistema de Inventario"
  plimpiar_campos
  fbotones ("PRENDER") 'apagar todos los botones y encender nuevo y salir
  fhabilitar_campos (False)
End Sub
```

3. Guarde y pruebe su trabajo.

LABORATORIO No. 4: Acceso a Base de Datos

Objetivos

Recuerde que debe crear una carpeta en su computador c:\laboratorio4, para esta practica no se requieren copiar los archivos contenidos en otros proyectos. La carpeta debe estar vacía.

Después de completar este laboratorio, estará en capacidad de:

- Usar el asistente Data Form para crear un formulario.
- Trabajar con ADO Data Control

Antes de continuar

Ejercicio No. 1: Crear un formulario de información del cliente con el asistente Data Form

- **Instalar el asistente Data Form**

1. Abra el Visual Basic, si no está abierto. Comience un nuevo proyecto. Basado en un **Exe estándar**
2. En el menú de complementos, seleccione el asistente para formulario de datos...
3. Si no se encuentra disponible esta opción. Ejecute los pasos 4 y 5
4. Seleccione el **Administrador de complementos**, y cuando aparezca la ventana Administrador de complementos, seleccione **VB6 Data Form Wizard**.
5. En la Sección comportamiento de carga, haga clic Cargado/Descargado y Cargar al iniciar y presione Aceptar.

- **Construir el formulario de información del cliente**

1. En el Menú **Complementos**, seleccione **Asistente para formulario de datos...**
2. Presione **siguientes** en la ventana Asistente para formulario de datos.
3. Seleccione **Access** en la lista de Base de datos, y presione **siguiente**.
4. Presione **Examinar** y seleccione **NWIND.MDB**, y presione **siguiente**. La cual se encuentra ubicada en C:\Archivos de programa\Microsoft Visual Studio\VB98
5. Asígnele el nombre de **frmClientes**, seleccione **Registro Individual** como la diseño del formulario, y **Control de Datos Ado** para el tipo de enlace, presione **siguiente**.
6. En la lista desplegable **Origen de Registro**, seleccione Clientes.
7. Agregue los campos **IdCliente NombreCompañia** y **NombreContacto** de la lista de campos disponibles a la lista de campos seleccionados, presione **siguiente**.
8. Haga Clic en Seleccionar todos (Seleccionar. todo) y presione **siguiente**.
9. Haga Clic en Finalizar, y presione **Aceptar**.

- **Remover el formulario por defecto.**

1. En el explorador de proyectos, seleccione Form1. **Nota:** El explorador de proyectos se activa presionando las teclas **Ctrl+R**
2. En el explorador de Proyectos, presione botón derecho del Mouse, y seleccione Quitar Form1.
3. En el explorador de Proyectos, seleccione propiedades del Proyecto1. **Nota:** También puedes activar las propiedades del proyecto haciendo click en Proyecto -> Propiedades del proyecto del menú principal
4. Nombre el proyecto como **ProyectoAdo**, y defina **frmClientes** como formulario de inicio, presione **Aceptar**.

5. Guarde el proyecto y pruebe la aplicación. Use los botones hacia delante y hacia atrás en el Data Control Ado para moverse a través de la base de datos. Modifique el texto en el campo nombre de la compañía y presione el botón Actualizar (update).
6. Presione Cerrar y finalice la aplicación.

Ejercicio No. 2: Crear un formulario de información de ordenes con el asistente Data Form

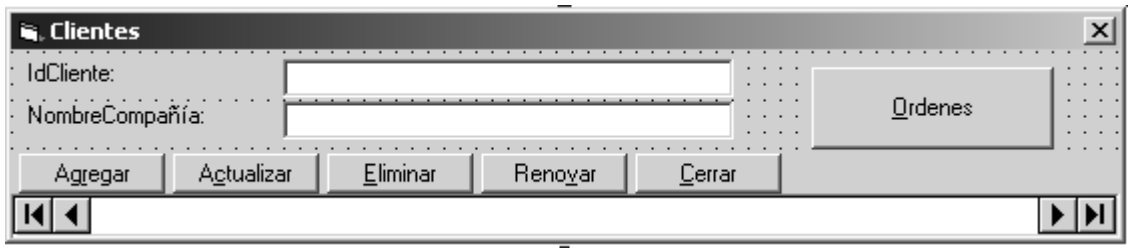
En este ejercicio, usará el Data Form para diseñar un formulario que desplegará información de órdenes. Entonces agregará un botón al formulario frmClientes para mostrar el formulario frmOrdenes.

- **Construir el formulario información de ordenes.**

1. En el Menú **Complementos**, seleccione **Asistente para formulario de datos…**
2. Presione **siguientes** en la ventana Asistente para formulario de datos.
3. Seleccione **Access** en la lista de Base de datos, y presione **siguiente**
4. Presione **Examinar** y seleccione **NWIND.MDB,** y presione **siguiente**.
5. Asignele el nombre de **frmOrdenes**, seleccione cuadrícula (Hoja de datos) en el cuadro de lista como la diseño del formulario, y **Control de Datos ADO** para el tipo de enlace, presione **siguiente**.
6. En la lista de Origen de Registro, seleccione **Pedidos**.
7. Agregue los campos **IdCliente, Idpedido, FechaPedido, FechaEntrega** y **Destinatario** de la lista de campos disponibles a la lista de campos seleccionados, presione **siguiente**.
8. Haga Clic en Seleccionar todos (Seleccionar todo) y presione **siguiente**.
9. Haga Clic en Finalizar, y presione **Aceptar.**

- **Despliegue el formulario frmOrdenes desde el formulario frmClientes.**

1. Ensanche el formulario frmClientes y agregue un botón Ordenes :



2. En el evento click del botón Ordenes, despliega el formulario **frmOrdenes** de manera modal.

```
frmOrdenes.show VbModal
```

Cuando un formulario se muestra de manera modal, el usuario debe proporcionar información o cerrar el formulario antes de usar cualquier otra parte de la aplicación

- **Guarde y pruebe su aplicación**

1. Guarde el proyecto.
2. Ejecute la aplicación. Haga clic en el botón Ordenes, navegue a través de los registros de la tabla Ordenes y trate de redimensionar el formato de los registros de la tabla Ordenes
3. Haga clic en Cerrar en el formulario frmOrdenes para regresar al formulario frmClientes.
4. Clic en cerrar y cierre la aplicación.

Ejercicio No. 3: Enlazando el formulario de Clientes con el formulario Ordenes con programación

A este punto, es bueno recordar que todo el código generado por los asistentes puede ser modificados parcial o totalmente para satisfacer las necesidades particulares de un proyecto de desarrollo.

Como se podrá apreciar, cuando se corre el sistema, siempre se muestran todas las ordenes de todos los clientes, lo cual es un comportamiento no deseado, lo correcto es mostrar únicamente los registros que correspondan a cada cliente.

- **Creando una sentencia SQL para filtrar los registros mostrados en el formulario**

5. Salve el proyecto.
6. Ejecute la aplicación. Haga clic en el botón Ordenes, navegue a través de los registros de la tabla Ordenes y trate de redimensionar el formato de los registros de la tabla Ordenes
7. Haga clic en Cerrar en el formulario frmOrdenes para regresar al formulario frmClientes.
8. Clic en cerrar y cierre la aplicación.
9. En Modo diseño, seleccione el formulario **frmordenes**, active el editor de código y seleccione el evento sub **form_load()**
10. Agregue el siguiente código como se muestra a continuación:

```
Private Sub Form_Load()
Dim vsql as string
vsql = "select IdCliente,IdPedido,FechaPedido,FechaEntrega,Destinatario from Pedidos"
vsql = vsql & " Where IdCliente='" & frmclientes.txtFields(0).Text & "'"
datPrimaryRS.RecordSource = vsql
datPrimaryRS.Refresh
End Sub
```

11. Pruebe el sistema**, presione F5**, verifique el comportamiento de los formularios, que pasa cuando se invoca la pantalla de las ordenes emitidas por un cliente. ¿ Que pasa, se esta realmente filtrando la información. ? Si es así, ***Muchas Felicitaciones***, Ud. Ha realizado un Excelente Trabajo, en caso contrario, lo invito a verificar los pasos llevados a cabo o consulte con el facilitador.
12. Salve su proyecto, comparta sus opiniones con el resto de los participantes.

LABORATORIO No. 5
Acceso a Base de Datos con Programación Usando ControlData

Objetivos
Copie todos los archivos de la carpeta **c:\Laboratorio3** en una nueva carpeta por nombre **c:\Laboratorio5**
Después de completar este laboratorio, estará en capacidad de:

- Usar el ControlData para acceso a Datos contenidos en Base de Datos y otros contenedores
- Programar botones de comando para el mantenimiento de Registros

Ejercicio No. 1: Creando la Base de Datos

En este ejercicio, Ud. Debe crear su propia Base de Datos en Access 97, que luego será manipulada desde su aplicación en Visual Basic

➢ **Creando la Base De Datos en Access 97**

1. Ejecute Microsoft Access 97
2. En la Caja de Dialogo, seleccione **Base de Datos en Blanco** y haga clic en Aceptar
3. En la siguiente Caja de Dialogo *Archivo Nueva Base de Datos*, indique el nombre del archivo de base de datos **inventario** y haga clic en **Crear,** Asegurese que el archivo este guardando en la ruta \laboratorio5, verifique el combo de selección Guardar en
4. En la pestaña de **Tablas**, presione el botón de **Nuevo**, para indicarle a Access que desea crear una nueva tabla.
5. Seleccione la opción **Vista Diseño** en la siguiente caja de dialogo y presione el botón de **Aceptar.**
6. Define la estructura de la tabla como se indica en la siguiente ilustración

Campo	Tipo de Datos	Tamaño del Campo
♦ Codigo (campo clave)	Texto	10
Nombre	Texto	25
Precio	Númerico	Simple
Fecha_ven	Fecha/Hora	
Tipo	Texto	10
Observaciones	Memo	

7. Cierre la venta de Diseño y cuando le pregunte Desea Guardar los Datos del Diseño, haga clic en el botón de **Si,** indique como nombre de la tabla **articulos** y presione el botón de Aceptar
8. Seleccionando la tabla de **articulos**, presione el botón de Abrir, Agregue varios registros de prueba. Mínimo 3 registros completos
9. Cierre Microsoft Access, Verifique que se encuentra guardada en la ruta c: \laboratorio5 la base de datos **inventario.mdb,** recuerde que más adelante será manipulada desde Visual Basic.

Ejercicio No. 2: Programar un Formulario de Mantenimiento de Registros

En este ejercicio, seleccionará el formulario **frmarticulos.frm** creado en el **Laboratorio 3** para su reutilización, agregará nuevos botones de comando y llevará a cabo ciertas modificaciones y ajustes.

- **Creando la Interfaz del Usuario.**

1. En el menú **Proyecto CTRL+R**, seleccione el formulario frmarticulos.
2. Agregue los botones necesarios, la interfaz debe lucir como se muestra en la siguiente ilustración

3. Defina las propiedades de los objetos como se presentan a continuación

Objeto	Propiedad	Nuevo Valor
Command1	Name	CmdEditar
	Caption	&Editar
	Style	1- Graphical
	Picture	(Icono) **...**
	Tooltiptext	Permite Editar un Registro
Command2	Name	Cmdborrar
	Caption	&Borrar
	Style	1 – Graphical
	Picture	(Icono) ...
	Tooltiptext	Permite Borrar Registros
Command3	Name	Cmdprimero
	Caption	&Primero
	Style	1 – Graphical
	Picture	(Icono) ...

	Tooltiptext	Muestra el primero registro del catalogo
Command4	Name	Cmdsiguiente
	Caption	&Siguiente
	Style	1 – Graphical
	Picture	(Icono) ...
	Tooltiptext	Visualizar el siguiente registro
Command5	Name	Cmdultimo
	Caption	&Ultimo
	Style	1 – Graphical
	Picture	(Icono) ...
	Tooltiptext	Ver ultimo registro de la tabla
Command6	Name	Cmdanterior
	Caption	&Anterior
	Style	1 – Graphical
	Picture	(Icono) ...
	Tooltiptext	Mostrar el anterior registro
Command7	Name	Cmdbuscar
	Caption	&Buscar
	Style	1 – Graphical
	Picture	(Icono) ...
	Tooltiptext	Búsqueda de Registros
Command8	Name	Cmdayuda
	Caption	&Ayuda
	Style	1 – Graphical
	Picture	(Icono) ...
	Tooltiptext	Visualizar ayuda del sistema

4. Seguidamente agregue el  control data que va a permitir la comunicación con la Base de Datos.
5. Insértelo en cualquier sitio del formulario, indique las siguientes propiedades:

Data1	(Nombre)	Datarticulos
	Connect	Access
	DatabaseName	... \seleccione la base de datos **inventario.mdb**
	RecordSource	articulos

6. Ahora bien, Ud. Debe enlazar cada campo de la tabla de la base de datos con las cajas de texto (textbox) del formulario activo, con la fin de manipular y visualizar la información contenida en la misma.
7. Seleccione cada una de las cajas de texto e indique las siguientes propiedades.

Txtcodigo	DataSource	Datarticulos
	Datafield	Codigo

8. Pruebe el funcionamiento del sistema, presione F5, y navegue por cada uno de los registros. Utilice el control data para desplazarse
9. Cierre el formulario, vamos a ocultar el control data, seleccione en modo diseño el control **datarticulos** y coloque la propiedad **Visible** en **False**
10. Ahora, Vamos a sustituir la funcionalidad del control data por los botones de programación de nuestra barra de herramientas, comience a programar y a probar de manera simultanea cada botón de comando con el siguiente set de instrucciones:
11. Programe cada botón de comando de la barra de herramientas y al mismo tiempo presione **F5** para correr y probar la aplicación, si tiene problemas con alguna rutina, consulte al facilitador del curso.

```
Private Sub cmdprimero_Click()
    Datarticulos.Recordset.MoveFirst 'mover el apuntador al principio
End Sub

Private Sub cmdultimo_Click()
   Datarticulos.Recordset.MoveLast 'movernos al ultimo
End Sub

Private Sub cmdsiguiente_Click()
  Datarticulos.Recordset.MoveNext
  If Datarticulos.Recordset.EOF Then
     Datarticulos.Recordset.MovePrevious
     MsgBox "Estamos al final del archivo.."
  End If
End Sub

Private Sub cmdanterior_Click()
Datarticulos.Recordset.MovePrevious
If Datarticulos.Recordset.BOF Then
  Datarticulos.Recordset.MoveNext
  MsgBox "Estamos ubicados al principio del Archivo"
End If
End Sub

Private Sub cmdbuscar_Click()
Dim vcodigo As String
vcodigo = InputBox("Ingrese el código a buscar:")
If vcodigo = "" Or IsNull(vcodigo) Then Exit Sub
Datarticulos.Recordset.FindFirst "codigo like '" & vcodigo & "*'"
If Datarticulos.Recordset.NoMatch Then
  MsgBox "Registro No existe"
  Datarticulos.Recordset.MoveFirst
End If
End Sub

Private Sub cmdnuevo_Click()
' Call plimpiar_campos
 fhabilitar_campos (True)
 fbotones ("APAGAR") 'apagar los botones y prender Grabar y Cancelar
 txtcodigo.SetFocus
 Datarticulos.Recordset.AddNew 'agregar un nuevo registro
End Sub

Private Sub cmdeditar_Click()
 fhabilitar_campos (True)
 fbotones ("APAGAR") 'apagar los botones y prender Grabar y Cancelar
 txtcodigo.Enabled = False 'deshabilitar el campo clave
```

```
 Datarticulos.Recordset.Edit 'editar el registro
End Sub

Private Sub cmdborrar_Click()
 If MsgBox("Desea Eliminar El Registro", vbYesNo + vbQuestion, "Sistema Inventario") =
vbYes Then
   Datarticulos.Recordset.Delete
   MsgBox "Registro Eliminado Satisfactoriamente..."
   Datarticulos.Recordset.MovePrevious
 End If
End Sub

Private Sub cmdgrabar_Click()
  '****************************************
  'validando los objetos antes de grabar el registro
  '****************************************
  If IsNull(txtcodigo.Text) Or txtcodigo.Text = "" Then
     MsgBox "Debe ingresar un código válido.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
     txtcodigo.SetFocus
     Exit Sub
  End If
  If Not IsNumeric(txtprecio.Text) Then
     MsgBox "Debe ingresar un monto válido.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
     txtprecio.SetFocus
     Exit Sub
  End If
 If Not IsDate(txtfecha_ven.Text) Then
      MsgBox "Debe ingresar una fecha válida.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
     txtfecha_ven.SetFocus
     Exit Sub
 End If
  Datarticulos.Recordset.Update  'actualizando el registro
  MsgBox "Registro Actualizado Satisfactoriamente..", vbExclamation +
vbDefaultButton1, "Sistema de Inventario"
  fbotones ("PRENDER") 'apagar todos los botones y encender nuevo y salir
  fhabilitar_campos (False)
End Sub

Private Sub cmdcancelar_Click()
  Call plimpiar_campos
  fbotones ("PRENDER")
  fhabilitar_campos (False)
  MsgBox "Operación Abortada Satisfactoriamente.."
  Datarticulos.Recordset.CancelUpdate
End Sub
```

12. Pruebe todos los botones, si todo funciona correctamente, ** ***Felicidades*** **, de lo contrario, verifique el contenido y lógica de las rutinas. Como pudo observar, la programación del botón de ayuda no se encuentra ilustrada, En este punto. Consulte con su facilitador los diferentes recursos disponibles para la elaboración de sistemas de ayuda.

13. Realice un toque final al formulario terminado. Programe cada botón de comando de la barra de herramientas y al mismo tiempo presione **F5** para correr y probar la aplicación, si tiene problemas con alguna rutina, consulte al facilitador del curso.

Ejercicio No. 3: Capturando los Errores en Tiempo de Ejecución

- **Toques Finales, Mejorando el Código, Interceptando los Códigos de Error.**

1. Ejecute el programa, presionando la tecla **F5**, intente agregar un nuevo registro utilizando un código ya existente en la Base de Datos. ¿Que Pasa? ¿Se generó un error?. Como pudo apreciar, la base de datos generó un error en tiempo de ejecución, ya que se intento agregar un registro utilizando un código ya existente y por supuesto, existe una violación de la integridad al ser el código la clave principal.
2. Vuelva a ejecutar el programa y anote el código de error generado por el RDBMS. **3022**
3. En Modo Diseño, seleccione el botón de Grabar y presione doble clic para activar la ventana de código
4. Realice ciertos ajustes al código como se ilustran a continuación. Observe que se utilizan las instrucciones **On Error Resume Next** y el objeto **Err** proporcionado por la herramienta Visual Basic

```
Private Sub cmdgrabar_Click()
  On Error Resume Next
  '*****************************************
  'validando los objetos antes de grabar el registro
  '*****************************************
  If IsNull(txtcodigo.Text) Or txtcodigo.Text = "" Then
     MsgBox "Debe ingresar un código válido.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
     txtcodigo.SetFocus
     Exit Sub
  End If
  If Not IsNumeric(txtprecio.Text) Then
     MsgBox "Debe ingresar un monto válido.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
     txtprecio.SetFocus
     Exit Sub
  End If
 If Not IsDate(txtfecha_ven.Text) Then
      MsgBox "Debe ingresar una fecha válida.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
     txtfecha_ven.SetFocus
     Exit Sub
 End If
  Datarticulos.Recordset.Update  'actualizando el registro
  If Err.Number = 3022 Then
    MsgBox "El Registo Ya Existe,,,Intente de Nuevo con otro código"
    txtcodigo.SetFocus
    Exit Sub
  End If
  MsgBox "Registro Actualizado Satisfactoriamente..", vbExclamation + vbDefaultButton1,
"Sistema de Inventario"
  fbotones ("PRENDER") 'apagar todos los botones y encender nuevo y salir
  fhabilitar_campos (False)
End Sub
```

LABORATORIO No. 6
Acceso a Base de Datos con Programación Usando RDO

Ejercicio No. 1: Agregando el nuevo Control RDO

Objetivos
Copie todos los archivos de la carpeta **c:\Laboratorio5** en una nueva carpeta por nombre **c:\Laboratorio6**
Después de completar este laboratorio, estará en capacidad de:

- Usar el RDO (Remote Data Object) para acceso a Datos contenidos en Base de Datos y otros contenedores
- Reutilizar el Código existente en los botones de comando para el mantenimiento de Registros

➢ **Insertando el Nuevo Control en la Barra de Herramientas**

1. Seleccione el formulario frmarticulos en modo diseño, y elimine el control data **datarticulos**
2. Sobre la barra de herramientas de controles, presione botón derecho del ratón y haga clic en la opción **Componentes...**
3. Sobre la caja de dialogo, seleccione el control **Microsoft RemoteData Control 6.0** y haga clic en aceptar. A este punto, puede apreciar que en la caja de herramientas debe aparecer el nuevo control de acceso a datos.
4. Seguidamente agregue el control al formulario. 
5. En Windows, Configuración -> Panel de Control -> Seleccione la opción **Fuentes de Datos ODBC (32 bits)**, en la pestaña de DSN de Sistema, haga clic en el botón de Agregar.
6. Sobre la caja de dialogo de proveedores de repositorios de datos, seleccione **Microsoft Access Driver (*.mdb)** y presione el botón de finalizar.
7. En la siguiente ventana, indique el nombre del origen de datos. Tipee, **conexion_inventario** y haga clic sobre el botón **Seleccionar...** para la ubicación de la base datos inventario.mdb, presione luego el botón de **Aceptar** y luego presione **Aceptar**
8. Maximice Visual Basic, indique las propiedades del nuevo objeto como se indican a continuación:

MSRDC1	(Nombre)	Datarticulos
	DataSourceName	... \seleccione sobre la lista desplegable el nuevo string de conexión recientemente creado **conexión_inventario**
	SQL	Select * from articulos
	Visible	False

1. Ahora bien, como ya todos los objetos se encuentran enlazados. Simplemente hay que realizar ciertos ajustes al código de los eventos de la barra de herramientas.
2. Con RDO en vez de utilizar **RecordSet**, se utilizan unos señores llamados **ResulSet** para ello, Se debe reemplazar todas las referencias a Recordset y sustituirlas por ResulSet.
3. Haga los cambios respectivos como se ilustran a continuación:

Si un procedimiento se muestra como:

```
Private Sub cmdsiguiente_Click()
  datarticulos.Recordset.MoveNext
  If datarticulos.Recordset.EOF Then
     datarticulos.Recordset.MovePrevious
     MsgBox "Estamos al final del archivo.."
  End If
End Sub
```

Cambielo por:

```
Private Sub cmdsiguiente_Click()
  datarticulos.Resultset.MoveNext
  If datarticulos.Resultset.EOF Then
     datarticulos.Resultset.MovePrevious
     MsgBox "Estamos al final del archivo.."
  End If
End Sub
```

4. Repita el procedimiento con todos los eventos. Es decir, en todos los eventos donde se hace referencia al método **Recordset** cambiar a **Resulset**
5. Pruebe la aplicación, presione **F5**

Ejercicio No. 2: Programando una Función de Búsqueda

Si ya ejecutó el programa y probó el funcionamiento de todos los botones, pudo observar que el botón de búsqueda genera un error. Esto sucede porque el método **FindFirst** pertenece al ControlData y el RDO no lo reconoce. Para resolver el problema hay que programar una función de búsqueda, la cual se ilustra a continuación:

1. En Modo Diseño, seleccione el botón de Buscar, presione el botón derecho del mouse y seleccione la opción **ver código**
2. Haga los cambios necesarios como se indican a continuación

```
Private Sub cmdbuscar_Click()
        Dim vcodigo As String, vexiste As Boolean
        vcodigo = InputBox("Ingrese el código a buscar:")
        If vcodigo = "" Or IsNull(vcodigo) Then Exit Sub
        datarticulos.Resultset.MoveFirst
        vexiste = False
        Do While Not datarticulos.Resultset.EOF
          If datarticulos.Resultset!codigo = vcodigo Then
            vexiste = True
            Exit Do
          Else
            datarticulos.Resultset.MoveNext
          End If
        Loop
        If Not vexiste Then
          MsgBox "Registro No existe"
          datarticulos.Resultset.MoveFirst
        End If
End Sub
```

3. Pruebe el funcionamiento del botón de búsqueda, presione F5 y prueba la aplicación
4. Cierre la ventana y guarde el proyecto.

LABORATORIO No. 7
Acceso a Base de Datos con Programación Usando ADO

Ejercicio No. 1: Agregando el nuevo Control ADO Visual (Activex Data Object)

Objetivos

Copie todos los archivos de la carpeta **c:\Laboratorio5** en una nueva carpeta por nombre **c:\Laboratorio7**

Después de completar este laboratorio, estará en capacidad de:

- Usar ADO Visual para acceso a Datos contenidos en Base de Datos y otros contenedores
- Reutilizar el Código existente en los botones de comando para el mantenimiento de Registros

➢ **Insertando el Nuevo Control en la Barra de Herramientas**

1. Seleccione el formulario frmarticulos en modo diseño, y elimine el control Data **datarticulos**
2. Sobre la barra de herramientas de controles, presione botón derecho del ratón y haga clic en la opción **Componentes...**
3. Sobre la caja de dialogo, seleccione el control **Microsoft ADO Data Control 6.0 (OLEDB)** y haga clic en aceptar. A este punto, puede apreciar que en la caja de herramientas debe aparecer el nuevo control de acceso a datos. 
4. Seguidamente agregue el control al formulario.
5. Maximice Visual Basic, indique las propiedades del nuevo objeto como se indican a continuación:

Adodc1	(Nombre)	Datarticulos
	ConnectionString	... \seleccione **Usar nombre de origenes de datos ODBC** y use el string creado **conexión_inventario**
	RecordSource	Select * from articulos
	Visible	False

1. Ahora bien, como ya todos los objetos se encuentran enlazados. Simplemente hay que realizar ciertos ajustes al código de los eventos de la barra de herramientas, los cuales se indican a continuación:

```
Note que en el procedimiento editar se eliminó el método edit, en ADO esto es implícito

Private Sub cmdeditar_Click()
 fhabilitar_campos (True)
 fbotones ("APAGAR") 'apagar los botones y prender Grabar y Cancelar
 txtcodigo.Enabled = False 'deshabilitar el campo clave
End Sub
```

2. Pruebe la aplicación, presione **F5**

Ejercicio No. 2: Programando una Función de Búsqueda

Si ya ejecutó el programa y probó el funcionamiento de todos los botones, pudo observar que el botón de búsqueda genera un error. Esto sucede porque el método **FindFirst** pertenece al ControlData y ADO no lo reconoce. Para resolver el problema hay que programar una función de búsqueda, la cual se ilustra a continuación:

3. En Modo Diseño, seleccione el botón de Buscar, presione el botón derecho del mouse y seleccione la opción **ver código**
4. Haga los cambios necesarios como se indican a continuación

```
Private Sub cmdbuscar_Click()
        Dim vcodigo As String
        vcodigo = InputBox("Ingrese el código a buscar:")
        If vcodigo = "" Or IsNull(vcodigo) Then Exit Sub
        datarticulos.Recordset.MoveFirst
        datarticulos.Recordset.Find "codigo ='" & vcodigo & "'"
        If datarticulos.Recordset.EOF Then
          MsgBox "Registro No existe"
          datarticulos.Recordset.MoveFirst
        End If
End Sub
```

5. Pruebe el funcionamiento del botón de búsqueda, presione F5 y pruebe la aplicación
6. Cierre la ventana y guarde su proyecto

LABORATORIO No. 8
Usando ADO Referencial

Ejercicio No. 1: Agregando una referencia a Activex Data Object

Objetivos
Copie todos los archivos de la carpeta **c:\Laboratorio7** en una nueva carpeta por nombre **c:\Laboratorio8**

Después de completar este laboratorio, estará en capacidad de:

- Usar ADO Referencial para acceso a Base de Datos
- Reutilizar el Código existente en los botones de comando para el mantenimiento de Registros

➢ **Insertando una Referencia a Nuestro Proyecto**

1. Abra el proyecto **MiprimerProyecto** que se encuentra en la carpeta recientemente creada c:\laboratorio8, quite el control visual ADO **datarticulos**
2. Sobre el menú Proyectos, haga clic en la opción **Componentes...** o presione las teclas **CTRL+T**
3. Quite el componente que apunta a **Microsoft ADO Data Control 6.0 (OLEDB)** elimine el tilde y presione el botón de aceptar.

Puede notar que en la barra de herramientas (toolbox) desaparece el control visual ADO ya que a partir de este momento será manipulado a nivel de programación.

4. En el menú de Proyectos, seleccione la opción **Referencias...**
5. En la caja de dialogo, ubique a **Microsoft ActiveX Data Object Library 2.1** y presione aceptar.
6. Si aparece una ventana de conflicto de versiones entre las referencias a las clases, entonces debe remover la referencia que apunta a una versión anterior, es decir, repita el procedimiento y elimine la referencia a **Microsoft ActiveX Data Object Library 2.0**

Ejercicio No. 2: Modificando el código, cambiando la filosofía de trabajo con ADO Referencial

7. Seleccione el formulario **frmarticulos**, en todas las cajas de texto (textbox) quite el enlace del control ADO visual, ubique las propiedades **DataSource** y **DataField** y elimine los valores de dichas propiedades. Repita el procedimiento con todas las cajas de texto.
8. Pruebe la aplicación, de seguro nada funciona. Se generan errores por todos lados, **no se preocupe, eso indica que todo marcha bien.**
9. Seleccione el formulario frmarticulos, presione botón derecho del ratón y active la ventana de código
10. En la Sección (General) (Declaraciones) Cree 2 variables de ambito público para el formulario como se detalla a continuación:

```
'declarando variables a nivel del formulario
Dim vconexion As New ADODB.Connection
Dim vrs As New ADODB.Recordset
```

Como podrá notar, Ud. A declarado una variable para establecer una conexión a la Base de Datos y otra variable para manipular el RecordSet

11. Modifique el código de todos los eventos del formulario **frmarticulos** como se le indica a continuación.

```
Private Sub Form_Load()
  fbotones ("PRENDER")
  fhabilitar_campos (False)
  vconexion.Open ("conexion_inventario")
  vrs.Open "select * from articulos", vconexion, adOpenDynamic, adLockOptimistic
  Call cmdprimero_Click
End Sub
```

Observe que existen 2 métodos que son iguales, pero hacen referencia a dos variables diferentes pero inseparables. Es decir, para manipular un conjunto de registros de un **RecordSet** se tiene que abrir un conjunto de registros **(vrs.open)** pero antes, es necesario abrir una conexión utilizando una variable **(vconexion.open)** la cual apunta a la base de datos en cuestión vía OLEDB, que en nuestro caso se llama **vconexion**

12. Agregue un nuevo procedimiento, desde la ventana de código, seleccione del menú **Herramientas** la opción **Agregar Procedimiento...** coloque como nombre **pmuestradatos** y presione **Aceptar.** Tipee las líneas de código como se detallan a continuación:

```
Private Sub pmuestradatos()
  txtcodigo.Text = IIf(vrs!codigo <> "", vrs!codigo, "")
  txtnombre.Text = IIf(vrs!nombre <> "", vrs!nombre, "")
  txtprecio.Text = IIf(vrs!precio <> "", vrs!precio, "0")
  txtfecha_ven.Text = IIf(vrs!fecha_ven <> "", vrs!fecha_ven, "")
  cbotipo.Text = IIf(vrs!tipo <> "", vrs!tipo, "")
  txtobservaciones.Text = IIf(vrs!observaciones <> "", vrs!observaciones, "")
End Sub
```

13. Revise todos los eventos de la barra de botones, realice los ajustes necesarios en función a las líneas de código que se muestran a continuación:

```
Private Sub cmdnuevo_Click()
 Call plimpiar_campos
 fhabilitar_campos (True)
 fbotones ("APAGAR") 'apagar los botones y prender Grabar y Cancelar
 txtcodigo.SetFocus
 cmdnuevo.Tag = "PRESIONADO"
End Sub

Private Sub cmdeditar_Click()
  fhabilitar_campos (True)
  fbotones ("APAGAR") 'apagar los botones y prender Grabar y Cancelar
  txtcodigo.Enabled = False 'deshabilitar el campo clave
  cmdeditar.Tag = "PRESIONADO"
End Sub

Private Sub cmdborrar_Click()
    If MsgBox("Desea Eliminar El Registro", vbYesNo + vbQuestion, "Sistema
    Inventario") = vbYes Then
      vrs.Delete adAffectCurrent
      MsgBox "Registro Eliminado Satisfactoriamente..."
      Call cmdanterior_Click
    End If
End Sub
```

```
Private Sub cmdgrabar_Click()
  On Error Resume Next
  '****************************************
  'validando los objetos antes de grabar el registro
  '****************************************
  If IsNull(txtcodigo.Text) Or txtcodigo.Text = "" Then
     MsgBox "Debe ingresar un código válido.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
     txtcodigo.SetFocus
     Exit Sub
  End If
  If Not IsNumeric(txtprecio.Text) Then
     MsgBox "Debe ingresar un monto válido.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
     txtprecio.SetFocus
     Exit Sub
  End If
 If Not IsDate(txtfecha_ven.Text) Then
      MsgBox "Debe ingresar una fecha válida.. << Verifique >>..!", vbDefaultButton1 +
vbInformation
     txtfecha_ven.SetFocus
     Exit Sub
 End If

 If cmdnuevo.Tag = "PRESIONADO" Then 'se activo el evento Agregar
     vrs.AddNew
     vrs!codigo = txtcodigo.Text
     vrs!nombre = txtnombre.Text
     vrs!precio = Val(txtprecio.Text)
     vrs!tipo = cbotipo.Text
     vrs!fecha_ven = CDate(txtfecha_ven.Text)
     vrs!observaciones = txtobservaciones.Text
 ElseIf cmdeditar.Tag = "PRESIONADO" Then
     vrs!nombre = txtnombre.Text
     vrs!precio = Val(txtprecio.Text)
     vrs!tipo = cbotipo.Text
     vrs!fecha_ven = CDate(txtfecha_ven.Text)
     vrs!observaciones = txtobservaciones.Text
 End If
 vrs.Update
 If Err.Number <> 0 Then
    MsgBox "El Registo Ya Existe,,,Intente de Nuevo con otro código"
    txtcodigo.SetFocus
    Err.Clear
    Exit Sub
 End If
 MsgBox "Registro Actualizado Satisfactoriamente..", vbExclamation + vbDefaultButton1,
"Sistema de Inventario"
 fbotones ("PRENDER") 'apagar todos los botones y encender nuevo y salir
 fhabilitar_campos (False)
 cmdnuevo.Tag = ""
 cmdeditar.Tag = ""
End Sub

Private Sub cmdcancelar_Click()
   Call plimpiar_campos
   fbotones ("PRENDER")
   fhabilitar_campos (False)
   MsgBox "Operación Abortada Satisfactoriamente.."
```

```
  Call pmuestradatos
End Sub

Private Sub cmdprimero_Click()
  vrs.MoveFirst
  Call pmuestradatos
End Sub

Private Sub cmdsiguiente_Click()
  vrs.MoveNext
  If vrs.EOF Then
    vrs.MovePrevious
    MsgBox "Estamos al final del archivo.."
  End If
  Call pmuestradatos
End Sub

Private Sub cmdultimo_Click()
   vrs.MoveLast
   Call pmuestradatos
End Sub

Private Sub cmdanterior_Click()
  vrs.MovePrevious
  If vrs.BOF Then
    vrs.MoveNext
    MsgBox "Estamos ubicados al prinicipio del Archivo"
  End If
  Call pmuestradatos
End Sub

Private Sub cmdbuscar_Click()
  Dim vcodigo As String
  vcodigo = InputBox("Ingrese el código a buscar:")
  If vcodigo = "" Or IsNull(vcodigo) Then Exit Sub
  vrs.MoveFirst
  vrs.Find "codigo ='" & vcodigo & "'"
  If vrs.EOF Then
    MsgBox "Registro No existe"
    vrs.MoveFirst
  End If
  Call pmuestradatos
End Sub
```

14. Ejecute el Sistema, presione **F5** para probar el funcionamiento de la aplicación.
15. Salve el Proyecto

LABORATORIO No. 9: Agregando Menús

Copie todos los archivos de la carpeta **c:\Laboratorio8** en una nueva carpeta por nombre **c:\Laboratorio9**

En este Laboratorio, agregará un menú, una barra de estado y un formulario que sirva como menú principal de nuestra aplicación.

Objetivos

Después de completar este laboratorio, estará en capacidad de:

- Agregar un menú al formulario
- Agregar una barra de estado
- Crear un Nuevo Formulario al Proyecto

Ejercicio No. 1: Agregando un menú y un Nuevo Formulario

- **Abrir el proyecto MiPrimerProyecto**

  Abrir MiprimerProyecto en el que ha estado trabajando. Búsquelo en la nueva carpeta creada

- **Crear un menú en frmPrincipal**

1. Agregue un nuevo formulario que sirva como pantalla principal, en el menú **Proyecto**, presione un clic sobre la opción **Agregar Formulario**
2. Defina las siguientes propiedades sobre el nuevo formulario creado

Form1	(Nombre)	Frmprincipal
	Caption	Menú Principal
	StartUpPosition	CenterScreen
	Icon	... \ selección un icono de su interes en la carpeta de imágenes.
	WindowState	Maximizado
	BackColor	Blanco / White

3. Active el editor de Menú, presionando las Teclas CTRL+E o en su defecto, seleccione del menú **Herramientas**, haga clic en **Editor de Menús...** Agregue los siguientes ítems de menú :

Caption	Name	Shorcut
&Archivo	MnuArchivo	None
…&Catalogo de Articulo	Mnuarticulo	CTRL+A
… -	Mnubarra1	
…E&xit	MnuSalirArchivo	None
&Edición	Mnuedicion	None
…&Copiar	MnuCopiar	CTRL+C
… -	Mnubarra2	
…&Pegar	MnuPegar	CTRL+V
… -	Mnubarra3	
…&Cortar	MnuPegar	CTRL+X

3. Después de agregar un ítem de menú, clic en OK

➢ **Agregando Código en los eventos del Menú**

1. Seleccione en **Modo Diseño** cada opción del menú previamente creado y agregue las siguientes líneas de código pertenecientes a cada item, como se describen a continuación:

```
Private Sub mnuarticulo_Click()
  Load frmarticulos
  frmarticulos.Show
End Sub

Private Sub mnucopiar_Click()
SendKeys "^{C}"
End Sub

Private Sub mnucortar_Click()
SendKeys "^{X}"
End Sub

Private Sub mnupegar_Click()
SendKeys "^{V}"
End Sub

Private Sub mnusalirarchivo_Click()
        If MsgBox("Desea Salir del Sistema", vbYesNo + vbQuestion, "SIST.
       INVENTARIO") = vbYes Then End
End Sub
```

2. Pruebe el Funcionamiento del Sistema, presione la tecla **F5**
3. Salve el Proyecto

Ejercicio No. 2: Agregando una barra de estado

➢ **Agregue Controles comunes de Microsoft Windows al proyecto**

1. En el menú proyecto, seleccione **Componentes**.
2. De la lista de controles disponibles, seleccione **Microsoft Windows Common Controls 6.0**, y presione clic en **Aceptar**.

Note que los controles son agregados a la barra de herramientas de Visual Basic.

➢ **Crear una barra de estado.** 

1. En la barra de herramientas, agregue el control de la barra de estatus en la parte baja del formulario **frmPrincipal**.
2. Defina la propiedad **Name** de la barra de estado como sbrProyecto.
3. Abra el cuadro de diálogo **Property Pages** del control Barra de Status, seleccionando la propiedad la **(Personalizado)** de la barra de estado en la venta de propiedades.
4. Seleccione la pestaña **Paneles**, y presione clic en el botón **Insertar Panel** para insertar un segundo panel.
5. Defina las propiedades para los dos paneles como se muestra a continuación :

Panel Index	Property	Desired Setting
1	Key	Msg
	Style	0 – sbrText

	Autosize	1 – sbrSpring
2	Key	Time
	Alignment	1 – sbrCenter
	Style	5 – sbrTime
	Autosize	2 – sbrContens
3	Key	Cap
	Alignment	1 – sbrCenter
	Style	5 – sbrCaps
	Autosize	2 – sbrContens

6. Haga clic en **Aplicar** para observar los efectos de la definición del ancho mínimo.
7. Haga clic en **Aceptar** cuando se sienta satisfecho del ancho.

- **Mensajes de la barra de estado**

1. Modifique la opción del menú principal, al momento de llamar a la pantalla de artículos, muestre un mensaje sobre la barra recientemente creada, ajuste el código como se indica a continuación:

```
Private Sub mnuarticulo_Click()
  sbrproyecto.Panels("msg").Text = "Catalogo de Artículos..."
  Load frmarticulos
  frmarticulos.Show
End Sub
```

2. Guarde el proyecto y pruebe su trabajo.

Ejercicio No. 3: Agregando una barra de Herramientas (ToolBox)

- **Agregue Controles comunes de Microsoft Windows al proyecto**

1. En el menú proyecto, seleccione **Componentes**.
2. De la lista de controles disponibles, seleccione **Microsoft Windows Common Controls 6.0**, y presione clic en **Aceptar**.

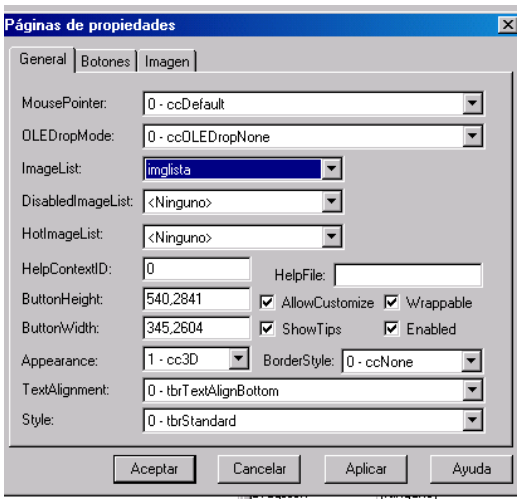
Note que los controles son agregados a la barra de herramientas de Visual Basic.

- **Crear un control de lista de Imágenes (imagelist)** 

6. En la barra de herramientas, agregue el control Imagelist en cualquier parte del formulario, este control es un contenedor de imágenes y solo esta disponible para el programador, es decir, carece de una interfaz de usuario.
7. Defina la propiedad **Name** de la lista de imágenes **imglista**.
8. Abra el cuadro de diálogo **Property Pages** del control Imagelist, seleccionando la propiedad la **(Personalizado)** de la barra de estado en la venta de propiedades.
9. Sobre la pestaña General, seleccione la configuración del tamaño de los iconos de la barra de herramienta, seleccione las dimensiones **16 x 16**
10. Seleccione la pestaña **Imágenes**, y seleccione cada una de las imágenes que van a formar parte del toolbox, presione clic en el botón **Insertar Imagén** para insertar imágenes por lo general con la extensión *.ico, Note que a cada imagen se le asigna automáticamente en la propiedad **index** un valor número consecutivo, para la primera Imagén el valor de **index** es **1**, para la segunda **index** vale **2** y así sucesivamente

- **Crear un control (Toolbox) y enlazarlo con el contenedor de imágenes**

11. En la barra de herramientas, agregue el control Toolbox en cualquier parte del formulario, este control dada su naturaleza, se va a colocar automáticamente en la parte superior del formulario MDI principal
12. Defina la propiedad **Name** del toobox **tboxmenu**.
13. Abra el cuadro de diálogo **Property Pages** del control Toolbox, seleccionando la propiedad la **(Personalizado)** de la barra de estado en la venta de propiedades.
14. Sobre la pestaña General, seleccione la en la lista desplegable el contenedor de imágenes creado en la sección anterior, "imglista"



configuración del tamaño de los iconos de la barra de herramienta, seleccione las dimensiones **16 x 16**

Seleccione la pestaña **Imágenes**, y seleccione cada una de las imágenes que van a formar parte del toolbox, presione clic en el botón **Insertar Imagén** para insertar

15. Inmediatamente, seleccione la pestaña botones. Haga clic en botón de **insertar botón** y defina las opciones del nuevo elemento indicando los valores como se describen a continuación: (Repita el procedimiento para agregar 2 botones en la barra de herramientas)

Index	**Propiedad**	**Valor**
1	Caption Key Image	Clientes CLIENTES 1
2	Caption	Salir
	Key	SALIR
	Image	2

16. Como puede observar, la propiedad **index** se refiere al index del elemento Imagén que se encuentra almacenado el contenedor de imágenes **imglista**, esto quiere decir que deben coincidir los elementos del contenedor de imágenes con las opciones del menu (toolbox) que desea programar.
17. En modo Diseño, sobre el formulario MDI, haga doble clic sobre el objeto tboxmenu
18. Agregue las siguientes lineas de código sobre el evento
19. Ahora, vamos a agregar las siguientes líneas de código sobre el evento **tboxmenu_ButtonClick**, tal como se indica a continuación:

```
Private Sub tboxmenu_ButtonClick(ByVal Button As MSComctlLib.Button)
  Select Case Button.Key
  Case Is = "CLIENTES"
    MsgBox "Invocar al formulario de clientes..."
  Case Is = "SALIR"
    MsgBox "Cerrar la aplicación y retornar a Windows..."
  End Select
End Sub
```

3. Guarde el proyecto y pruebe su trabajo.

LABORATORIO No. 10: Creando reportes con Data Report

En este Laboratorio, agregará un objeto Data Environment, Conección, Command y un Data Report. No se requiere copiar el contenido de la carpeta \Laboratorio8 hacia \Laboratorio9

Objetivos

Después de completar este laboratorio, estará en capacidad de:

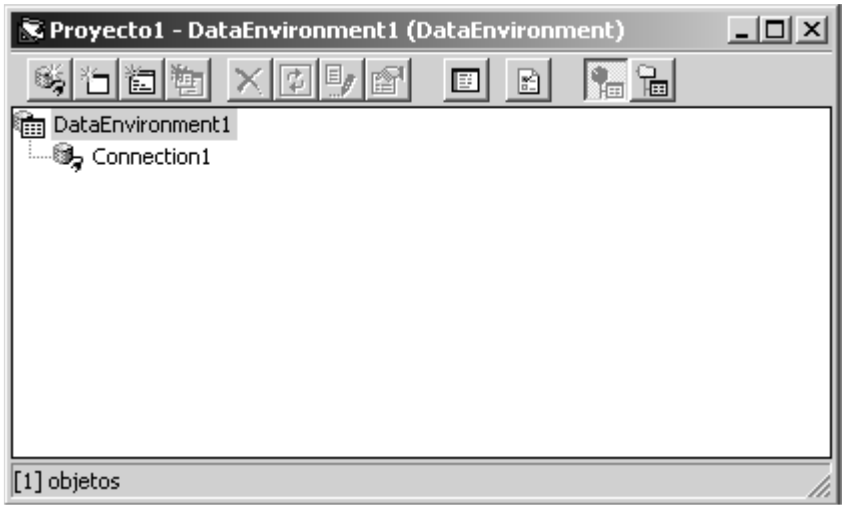
- Agregar un objeto Data Environment
- Agregar un objeto Data command
- Agregar un objeto Data Report
- Crear un reporte con Data Report

Ejercicio No. 1: Agregando el objeto Data Environment

➢ **Agregue el objeto Data Environment**

1. Comience un proyecto en blanco, Agregue un Proyecto basado en .EXE Estandar.
2. En el menú **Proyecto**, seleccione **Mas Diseñadores ActiveX...** y luego haga clic en **Data Environment**.

   Se debe abrir la ventana del DataEnvironment, con dos objetos



3. Seleccione el objeto **DataEnvironment1**, para cambiarle el nombre en el panel de propiedades, asígnele el nombre DEClientes., utilice la ventana de propiedades, presionando **F4**
4. Seleccione el objeto Connection1, para cambiarle el nombre en el panel de propiedades, asignele el nombre cnnBaseDatos.
5. Con el mismo objeto seleccionado, presione el botón derecho del mouse y hacemos clic en propiedades.
6. En la ventana propiedades de vínculo de datos, seleccione el proveedor **Microsoft Jet 4.0 OLE DB Provider**, presione siguiente.

Utilice la ficha **Proveedor** para seleccionar el proveedor de OLE DB adecuado para el tipo de datos a los que desea tener acceso. Microsoft Jet OLE DB es el motor nativo de Base de Datos.

6. En el cuadro de diálogo del nombre de la base de datos presione el botón examinar y seleccione la base de datos que quiera usar, en nuestro caso **NWIND.MDB,** la cual se encuentra ubicada en la ruta C:\Archivos de programa\Microsoft Visual Studio\VB98.
7. Presione el botón Probar conexión. Si la conexión en satisfactoria, presione **Aceptar**.

- **Crear los comandos para acceso de los objetos de Base de datos**

1. Agregue ahora un comando, esto se puede realizar a través del botón  .
2. Seleccione el objeto Command1, a través del panel de propiedades cambie el nombre a cmmClientes.
3. Establezca las propiedades del comando **cmmClientes** (hacer clic en botón derecho – propiedades).
4. En la lista desplegable **Objeto de base de datos** del cuadro de diálogo de propiedades cmmDataReport, seleccione el objeto Tabla, y en la lista desplegable **nombre del objeto** seccione clientes.

5. Presione Aceptar.

Para comprobar si la conexión esta bien realizada, presione el botón ⊞ del objeto cmmDataReport y deben verse todos los campos de la tabla seleccionada.

- **Crear el reporte con Data Report**

1. En el explorador de proyectos, seleccione el Proyecto (Proyecto1), y haga clic con el botón derecho del mouse, en el menú contextual ubique el puntero sobre la opción **Agregar**, entonces presionamos sobre **Data Report**.
2. Aparece la ventana del objeto **Data Report**, donde puede diseñar del Reporte.
3. En la sección Encabezado del reporte, escriba el titulo del reporte, y en la sección Detalles agregue los campos de la tabla que se reflejaran en el reporte.

- **Insertando los campos al reporte**

1. Haga doble clic en el **Data Environment** (DEClientes), haga clic sobre el botón expandir del comando **cmmdataReport**.
2. Cuando se despliegue la lista de campos, haga clic sobre el campo **IdCliente** y sin soltar el botón llevarlo (arrastrar) hasta la sección Detalles del objeto **DataReport**, suelte el botón. Repita el procedimiento para los campos **NombreCompañia**, **NombreContacto**, y **Ciudad.**

En este momento aparece dos veces la palabra IdClientes en la sección Detalles, uno rodeado con línea puntiaguda (etiquetas) y el otro con línea normal (campo de B/D).

3. Borre la etiqueta **IdClientes**, y escriba los títulos al reporte. Repita el procedimiento con los demás campos insertados en el reporte
4. En el panel de propiedades del **Data Report**, en la propiedad **DataSource** seleccione el Data Environment relacionado con el reporte (DEClientes)
5. En la propiedad **DataMember** seleccione **cmmDataReport** que no es más que la tabla relacionada con el reporte.

➢ **Invocando el reporte desde un botón.**

1. Cree un formulario, como ya lo ha hecho en prácticas anteriores.
2. Agréguele un botón de comando. Haga doble clic sobre el botón.
3. En el editor de código escribir la siguiente sentencia :

```
DataReport1.Show
```

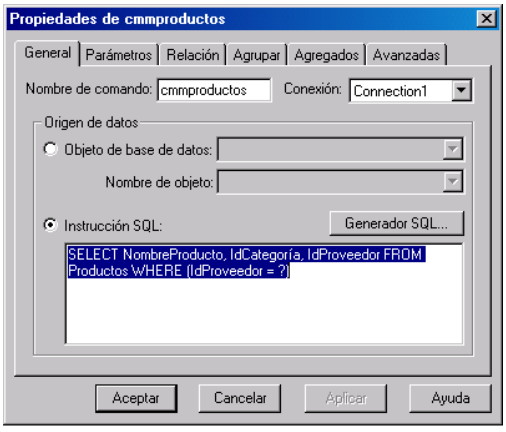
➢ **Guarde y pruebe su aplicación**

1. Guarde el proyecto.
2. Ejecute la aplicación. Haga clic en el botón recién creado.
3. Clic en cerrar y cierre la aplicación.

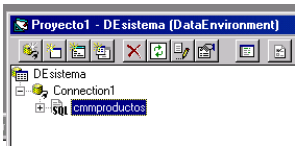
Ejercicio No. 2: Llamar a un reporte con filtros y parámetros.

➢ **Agregue un nuevo comando con filtro**

1. Utilice el mismo proyecto recientemente creado en el ejercicio anterior
2. Agregue ahora un comando, esto se puede realizar a través del botón  .
3. Seleccione el objeto Command1, a través del panel de propiedades cambie el nombre a **cmmproductos**
4. Establezca las propiedades del comando **cmmproductos** (hacer clic en botón derecho de Propiedades).
5. En la lista desplegable **Objeto de base de datos** del cuadro de diálogo de propiedades cmmDataReport, seleccione Instrucción SQL y tipee la sentencia SQL tal cual como se describe a continuación:



6. Tipee la sentencia **SELECT NombreProducto, IdCategoría, IdProveedor FROM Productos WHERE (IdProveedor = ?)** y luego Presione el botón de Aceptar. Note que este nuevo comando en la ventana DataEnvironment tiene un icono diferente a los comandos creados por una tabla, tal como se muestra en la figura:



- **Crear el reporte con Data Report**

4. En el explorador de proyectos, seleccione el Proyecto (Proyecto1), y haga clic con el botón derecho del mouse, en el menú contextual ubique el puntero sobre la opción **Agregar**, entonces presionamos sobre **Data Report**.
5. Aparece la ventana del objeto **Data Report**, donde puede diseñar del Reporte.
6. En la sección Encabezado del reporte, escriba el titulo del reporte, y en la sección Detalles agregue los campos del nuevo comando recientemente creado en el paso anterior.

- **Insertando los campos al reporte**

3. Haga doble clic en el **Data Environment** (DEClientes), haga clic sobre el botón expandir del comando **cmmproductos**
4. Cuando se despliegue la lista de campos, haga clic sobre el campo **NombreProducto** y sin soltar el botón llevarlo (arrastrar) hasta la sección Detalles del objeto **DataReport**, suelte el botón. Repita el procedimiento para los campos **IdCategoria**, **IdProveedor.**

En este momento aparece dos veces la palabra IdCategoria en la sección Detalles, uno rodeado con línea puntiaguda (etiquetas) y el otro con línea normal (campo de B/D).

6. Borre la etiqueta **IdCategoria**, y escriba los títulos al reporte. Repita el procedimiento con los demás campos insertados en el reporte
7. En el panel de propiedades del **Data Report**, en la propiedad **DataSource** seleccione el Data Environment relacionado con el reporte **(DEClientes)**
8. En la propiedad **DataMember** seleccione **cmmproveedor** que no es más que el comando que contiene la sentencia SQL asociada con el reporte.
9. Haga doble clic sobre la ventana del diseñador de reporte para visualizar la ventana de código o seleccionando la ventana reporte invoque la opción **Código** del menú **Ver** en el menú principal.
10. Tipee las siguientes líneas de código, en el evento que se dispara cuando finaliza el reporte, como se indica a continuación:

```
Private Sub DataReport_Terminate()
        DEClientes.rscmmproductos.Close
End Sub
```

- **Invocando el reporte desde un botón.**

4. Cree un formulario, como ya lo ha hecho en prácticas anteriores.
5. Agréguele un botón de comando. Haga doble clic sobre el botón.
6. En el editor de código escribir la siguiente sentencia:

```
DEsistema.cmmproductos ("1")
DataReport2.Show
```

7. En este caso, siempre se van a imprimir los productos del proveedor con Id=1, para mejorar el código, seria interesante solicitar la información de entrada por medio de una lista desplegable (ComboBox) o una caja de texto (textbox) para darle un valor agregado al ejercicio.
8. Pruebe el Reporte
9. Salve el Proyecto