

# Didáctica de Nuevas Tecnologías en la E.S.O.

Programación y Matemáticas  
Vol. II



# **Didáctica de Nuevas Tecnologías en la E.S.O. - Programación y Matemáticas**

**(Vol II)**

**Francisco Luis Flores Gil**

© 2008. Francisco Luis Flores Gil  
Portada diseño y difusión de la obra: Íttakus



Licencia Creative Commons

Edición cortesía de [www.publicatuslibros.com](http://www.publicatuslibros.com). Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciador (pero no de una manera que sugiera que tiene su apoyo o apoyan el uso que hace de su obra). No puede utilizar esta obra para fines comerciales. Si altera o transforma esta obra, o genera una obra derivada, sólo puede distribuir la obra generada bajo una licencia idéntica a ésta. Al reutilizar o distribuir la obra, tiene que dejar bien claro los términos de la licencia de esta obra. Alguna de estas condiciones puede no aplicarse si se obtiene el permiso del titular de los derechos de autor. Nada en esta licencia menoscaba o restringe los derechos morales del autor.

Publicatuslibros.com es una iniciativa de:



Íttakus, sociedad para la información, S.L.  
C/ Sierra Mágina, 10.  
23009 Jaén-España  
[www.ittakus.com](http://www.ittakus.com)



# ÍNDICE

<b>1. INTRODUCCIÓN</b> .....	<b>6</b>
<b>2. ¿QUÉ ES VISUAL BASIC?</b> .....	<b>7</b>
<b>3. PRÁCTICA 1. CALCULADORA REDUCIDA</b> .....	<b>8</b>
3.1.    OBJETOS DE LA APLICACIÓN.....	9
3.2.    PROPIEDADES DE LOS OBJETOS .....	9
3.3.    CÓDIGO DE LA APLICACIÓN .....	10
<b>4. PRÁCTICA 2. CALCULADORA COMPLETA</b> .....	<b>11</b>
4.1.    OBJETOS DE LA APLICACIÓN Y SUS PROPIEDADES.....	11
4.2.    FORMULARIO Y MÓDULO DE LA APLICACIÓN .....	12
4.2.1. Módulo .....	12
4.2.2. Formulario.....	14
<b>5. PRÁCTICA 3. EL ASISTENTE PARA APLICACIONES DE VB</b> .....	<b>17</b>
5.1.    INTRODUCCIÓN. ....	17
5.2.    CREAR UN FORMULARIO .....	18
5.2.1. Asistente para aplicaciones - Introducción .....	18
5.2.2. Asistente para aplicaciones - Tipo de interfaz .....	18
5.2.3. Asistente para aplicaciones - Menús .....	18
5.2.4. Asistente para aplicaciones - Personalizar la barra de herramientas.....	18
5.2.5. Asistente para aplicaciones - Recursos.....	18
5.2.6. Asistente para aplicaciones - Conexión a internet.....	19
5.2.7. Asistente para aplicaciones - Formularios estándar.....	19
5.2.8. Asistente para aplicaciones - Formulario de acceso a datos .....	19
5.2.8.1. Asistente para formulario de datos - Pantalla Introducción .....	20
5.2.8.2. Asistente formulario de datos - Pantalla Tipo de base de datos .....	20
5.2.8.3. Asistente para formulario de datos - Base de datos.....	20
5.2.8.4. Asistente para formulario de datos - Formulario.....	21
5.2.8.5. Asistente para formulario de datos - Origen de datos .....	21
5.2.8.6. Asistente para formulario de datos - Selección de controles .....	22
5.2.8.7. Asistente para formulario de datos - Finalizado .....	22
5.2.8.8. Asistente para formulario de datos - Finalizado .....	23
5.3.    RESULTADO DE LA PRÁCTICA .....	24
<b>6. PRÁCTICA 4. CREAR UN FORMULARIO MAESTRO - DETALLE</b> .....	<b>25</b>
6.1.    TABLAS NECESARIAS .....	25
6.2.    ASISTENTE PARA APLICACIONES - INTRODUCCIÓN .....	25
6.3.    ASISTENTE PARA APLICACIONES - TIPO DE INTERFAZ .....	26
6.4.    ASISTENTE PARA APLICACIONES - MENÚS .....	26
6.5.    ASISTENTE PARA APLICACIONES - PERSONALIZAR BARRA DE HERRAMIENTAS .....	27
6.6.    ASISTENTE PARA APLICACIONES - RECURSOS .....	27
6.7.    ASISTENTE PARA APLICACIONES - CONEXIÓN A INTERNET .....	27
6.8.    ASISTENTE PARA APLICACIONES - FORMULARIO ESTÁNDAR.....	27
6.9.    ASISTENTE PARA APLICACIONES - FORMULARIO DE ACCESO A DATOS .....	27
6.9.1. Asistente para formulario de datos - Introducción .....	28
6.9.2. Asistente para formulario de datos - Pantalla tipo de base de datos.....	28
6.9.3. Asistente para formulario de datos - Base de datos .....	28
6.9.4. Asistente para formulario de datos - Formulario.....	29
6.9.5. Asistente para formulario de datos - Origen de registros principal.....	30
6.9.6. Asistente formulario de datos - Relación del origen de registros .....	31
6.9.7. Asistente para formulario de datos - Selección de controles.....	31

6.9.8.	Asistente para formulario de datos - Finalizado.....	32
6.10.	ASISTENTE PARA APLICACIONES - FINALIZADO.....	33
<b>7.</b>	<b>MODIFICAR EL MENÚ DEL PROYECTO GESTION_DE_PEDIDOS.....</b>	<b>35</b>
<b>8.</b>	<b>AGREGAR EL FORMULARIO CLIENTES AL PROYECTO GESTION_DE_PEDIDOS....</b>	<b>39</b>
<b>9.</b>	<b>PRÁCTICA 5. CREAR FORMULARIO DE MANTENIMIENTO DE UNA TABLA.....</b>	<b>42</b>
9.1.	OBJETIVOS .....	42
9.2.	EL ADMINISTRADOR VISUAL DE BASE DE DATOS DE VISUAL BASIC.....	43
9.3.	CREAR EL FORMULARIO DE MANTENIMIENTO DE LA TABLA PROVINCIAS .....	46
9.3.1.	Creación del proyecto .....	46
9.3.2.	Objetos del formulario.....	46
9.3.3.	Código del formulario.....	48
<b>10.</b>	<b>PRÁCTICA 6. CREAR FORMULARIO DE CONSULTA.....</b>	<b>50</b>
10.1.	CREACIÓN DEL PROYECTO .....	51
10.2.	AGREGAR COMPONENTE.....	51
10.3.	OBJETOS DEL FORMULARIO.....	52
10.4.	CÓDIGO DEL FORMULARIO .....	53
10.5.	RESULTADO .....	54
<b>11.</b>	<b>PRÁCTICA 7. CREAR INFORME DE DATOS .....</b>	<b>55</b>
11.1.	OBJETIVO .....	56
11.2.	AGREGAR DATA ENVIROMENT Y ESTABLECER CONEXIÓN CON BASE DE DATOS .....	56
11.3.	CREACIÓN DEL INFORME .....	62
11.4.	VISUALIZAR EL INFORME .....	63

## 1. Introducción

El Visual Basic 6.0 constituye un lenguaje de programación muy visual y sencillo de manejar.

Su gran coordinación con otras herramientas de Microsoft como Excel y Access así como lo sencillo de su aprendizaje lo convierte en un lenguaje de programación muy propio para los que se quieran iniciar en el mundo de la programación.

Al mismo tiempo Microsoft proporciona de manera gratuita versiones para ser usadas por el personal docente desde su página web.

Es por esto por lo que es ideal para aquellos profesores y alumnos de la E.S.O. que quieran aprender un lenguaje de programación.

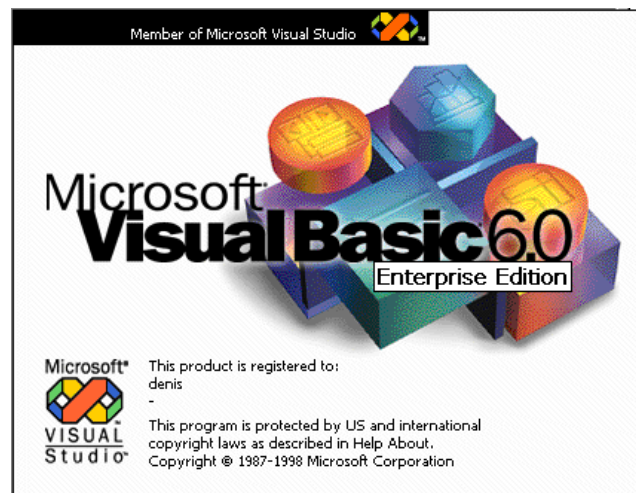
En este libro trataremos de explicar los puntos fundamentales del Visual Basic mediante prácticas que irán siendo explicadas paso a paso.

## 2. ¿Qué es Visual Basic?

Visual Basic es un lenguaje de programación desarrollado por Alan Cooper para Microsoft.

El lenguaje de programación es un dialecto de BASIC, con importantes añadidos.

Su primera versión fue presentada en 1991 con la intención de simplificar la programación utilizando un ambiente de desarrollo completamente gráfico que facilitara la creación de interfaces gráficas y en cierta medida también la programación misma.



Visual Basic constituye un IDE (entorno de desarrollo integrado o en inglés Integrated Development Environment) que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código (programa donde se escribe el código fuente), un depurador (programa que corrige errores en el código fuente para que pueda ser bien compilado), un compilador (programa que traduce el código fuente a lenguaje de máquina), y un constructor de interfaz gráfica o GUI (es una forma de programar en la que no es necesario escribir el código para la parte gráfica del programa, sino que se puede hacerlo de forma visual).

En este libro vamos a usar la versión Visual Basic 6.0.

### 3. Práctica 1. Calculadora reducida

Los alumnos de la E.S.O. trabajan frecuentemente con la calculadora en clase para realizar todo tipo de operaciones matemáticas.

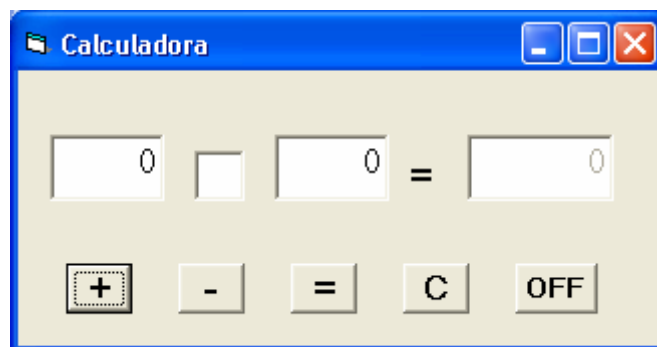
En los casos de los centros TIC el uso de los ordenadores durante las clases es muy frecuente por lo que los alumnos están acostumbrados al manejo la calculadora que traen los diferentes sistemas operativos como WINDOWS. Pero, ¿conocen realmente como funciona internamente una calculadora?



Con esta práctica en Visual Basic, los alumnos entenderán perfectamente el funcionamiento de una calculadora viendo las diferentes operaciones que debe de realizar internamente cada vez que la usamos.

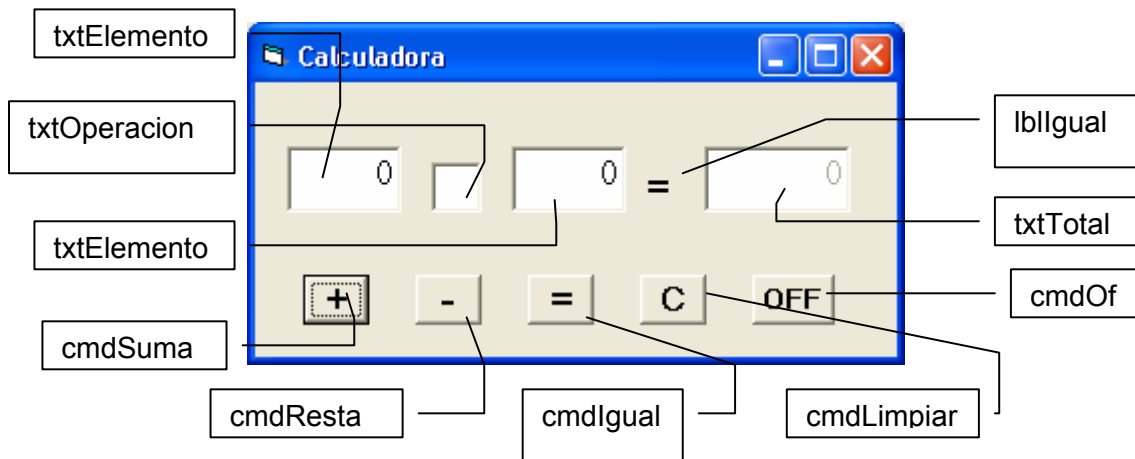
Este tipo de prácticas puede englobar varias asignaturas: Matemáticas, Tecnología e Informática, por lo que los profesores deberán coordinarse para su realización.

A continuación iremos describiendo los diferentes pasos que deberán ir dando los alumnos para obtener finalmente su calculadora simple, que tendrá la siguiente apariencia.



### 3.1. Objetos de la aplicación

Los objetos que usaremos en la práctica serán:



### 3.2. Propiedades de los objetos

Las propiedades de los objetos que usaremos en la práctica serán:

Nombre del objeto	Propiedades
txtElemento1	Nombre: txtElemento1 Alignment: Right Justify Text:
txtOperacion	Nombre: txtOperacion Alignment: Center Text: Enabled: False
txtElemento2	Nombre: txtElemento Alignment: Right Justify Text:
TxtTotal	Nombre: txtTotal Alignment: Right Justify Text: Enabled: False
LblIguar	Nombre: lblIguar Alignment: Center Caption: =
CmdSuma	Nombre: cmdSuma Caption: +
CmdResta	Nombre: cmdResta Caption: -
cmdLimpiar	Nombre: cmdLimpiar Caption: +
CmdOff	Nombre: cmdOff Caption: +

### 3.3. Código de la aplicación

El único formulario existente tendrá por código:

```
Private Sub cmdIguar_Click()
    On Error GoTo Error
    If txtOperacion.Text = "+" Then
        txtTotal.Text = Val(txtElemento1.Text) + Val(txtElemento2.Text)
    Else
        If txtOperacion.Text = "-" Then
            txtTotal.Text = Val(txtElemento1.Text) - Val(txtElemento2.Text)
        Else
            MsgBox "Introduzca la operación que desea realizar",
                vbInformation, "AVISO"
        End If
    End If
Error:
    If Err.Number <> 0 Then
        MsgBox "Descripción del Error: " & Err.Description, vbCritical,
            "ERROR"
    End If
Exit Sub
End Sub

Private Sub cmdLimpiar_Click()
    txtOperacion.Text = ""
    txtElemento1.Text = 0
    txtElemento2.Text = 0
    txtTotal.Text = 0
End Sub

Private Sub cmdOff_Click()
    End
End Sub

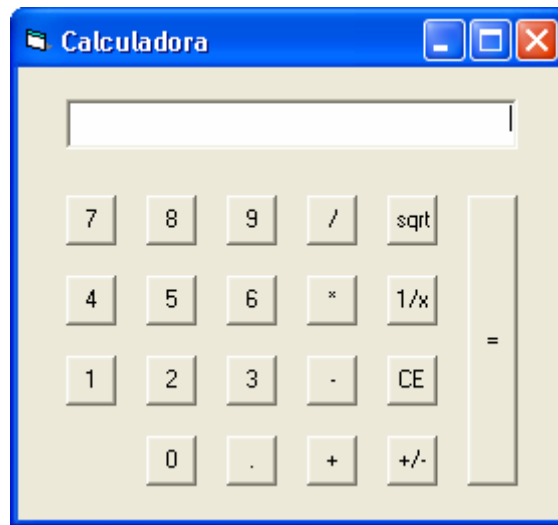
Private Sub cmdResta_Click()
    txtOperacion.Text = "-"
End Sub

Private Sub cmdSuma_Click()
    txtOperacion.Text = "+"
End Sub

Private Sub Form_Load()
    txtElemento1.Text = 0
    txtElemento2.Text = 0
    txtTotal.Text = 0
End Sub
```

## 4. Práctica 2. Calculadora completa

Objetivo: Crear una calculadora con el siguiente aspecto:



### 4.1. Objetos de la aplicación y sus propiedades

Las propiedades de los objetos que usaremos en la práctica serán:

Nombre del objeto	Propiedades
cmd0	Nombre: cmd0 Caption: 0
cmd1	Nombre: cmd1 Caption: 1
...	...
cmd9	Nombre: cmd9 Caption: 9
CmdPunto	Nombre: cmdPunto Caption: .
cmdDivision	Nombre: cmdDivision Caption: /
cmdMultiplicacion	Nombre: cmdMultiplicacion Caption: *
CmdResta	Nombre: cmdResta Caption: -
CmdSuma	Nombre: cmdSuma Caption: +
CmdSqrt	Nombre: cmdsqrt Caption: sqrt
cmdInverso	Nombre: cmdInverso Caption: 1/x
CmdSuma	Nombre: cmdCE Caption: CE
CmdSigno	Nombre: cmdSigno Caption: +/-
CmdIguar	Nombre: cmdIguar Caption: =
Text1	Nombre: Text1 Alignment: Right Justify Text:

## 4.2. Formulario y módulo de la aplicación

Esta práctica en Visual Basic no sólo está compuesta de un Formulario, como la anterior, sino también de un módulo.

Un módulo es un fichero Visual Basic donde escribimos parte del código de nuestro programa, y sólo parte, porque suele haber código en el formulario también.

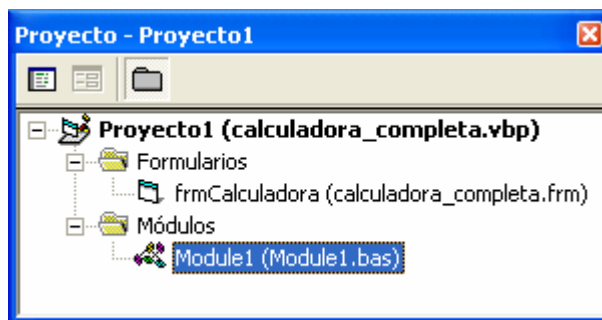
### 4.2.1. Módulo

En nuestro módulo vamos a definir una serie de variables que nos servirán para controlar el funcionamiento de la aplicación.

Las variables declaradas serán públicas para ello usaremos la palabra Public y por ello estarán accesibles desde todos los formularios de la aplicación. Para conseguirlo tendremos que declararlas en el módulo de código, no en la sección de declaraciones del formulario de los que conste la aplicación.

Para crear un módulo de código en el menú principal de Visual Basic marcamos en el menú principal las siguientes opciones Proyecto/Agregar módulo y aparecerá junto a los demás formularios de la ventana de proyecto aunque con un icono distinto indicando que se trata de un módulo de código.

Su apariencia es la siguiente:



El código que pondremos en nuestro módulo será:

```
Public A As Double
Public C As Double
Public Flag As String
Public blnLimpiar As Boolean
Public blnsign As Boolean

Sub Limpiar()
    If blnLimpiar = True Then
        frmCalculadora.Text1.Text = ""
        blnLimpiar = False
    End If
End Sub

Sub Operacion()
    Select Case Flag
        Case "suma"
            A = A + Val(frmCalculadora.Text1.Text)
        Case "resta"
```

```
    A = A - Val(frmCalculadora.Text1.Text)
Case "multiplicacion"
    A = A * Val(frmCalculadora.Text1.Text)
Case "division"
    If Val(frmCalculadora.Text1.Text) <> 0 Then A = A /
Val(frmCalculadora.Text1.Text)
    Case Else
        A = Val(frmCalculadora.Text1.Text)
End Select
frmCalculadora.Text1.Text = A
blnLimpiar = True
End Sub
```

#### 4.2.2. Formulario

El formulario tendrá el siguiente código:

```
Private Sub Form_Load()  
    Text1.Text = ""  
    A = 0  
    C = 0  
    Flag = ""  
    blnsign = False  
    blnLimpiar = False  
End Sub  
  
Private Sub cmd0_Click()  
    Limpiar  
    Text1.Text = Text1.Text & "0"  
End Sub  
  
Private Sub cmd1_Click()  
    Text1.Text = Text1.Text & "1"  
End Sub  
  
Private Sub cmd2_Click()  
    Limpiar  
    Text1.Text = Text1.Text & "2"  
End Sub  
  
Private Sub cmd3_Click()  
    Limpiar  
    Text1.Text = Text1.Text & "3"  
End Sub  
  
Private Sub cmd4_Click()  
    Limpiar  
    Text1.Text = Text1.Text & "4"  
End Sub  
  
Private Sub cmd5_Click()  
    Limpiar  
    Text1.Text = Text1.Text & "5"  
End Sub  
  
Private Sub cmd6_Click()  
    Limpiar  
    Text1.Text = Text1.Text & "6"  
End Sub  
  
Private Sub cmd7_Click()  
    Limpiar  
    Text1.Text = Text1.Text & "7"  
End Sub
```

```

Private Sub cmd8_Click()
    Limpiar
    Text1.Text = Text1.Text & "8"
End Sub

Private Sub cmd9_Click()
    Limpiar
    Text1.Text = Text1.Text & "9"
End Sub

Private Sub cmdCE_Click()
    Text1.Text = ""
End Sub

Private Sub cmdInverso_Click()
    If Text1 <> "0" And Text1 <> "" Then
        Text1.Text = 1 / (Val(Text1.Text))
    Else
        MsgBox "División por cero", vbCritical, "ERROR"
    End If
End Sub

Private Sub cmdPunto_Click()
    Text1.Text = Text1.Text & "."
End Sub

Private Sub cmdSigno_Click()
    If blnsign = False Then
        If Val(Text1.Text) > 0 Then
            Text1.Text = "-" & Text1.Text
        End If
        blnsign = True
    Else
        Text1.Text = -Val(Text1.Text)
        blnsign = False
    End If
End Sub

Private Sub cmdSuma_Click()
    Operacion
    Flag = "suma"
End Sub

Private Sub cmdResta_Click()
    Operacion
    Flag = "resta"
End Sub

Private Sub cmdMultiplicacion_Click()
    Operacion
    Flag = "multiplicacion"
End Sub

```

```

Private Sub cmdDivision_Click()
    Operacion
    Flag = "division"
End Sub

```

```

Private Sub cmdsqrt_Click()
    On Error GoTo Error
    Text1.Text = Sqr(Text1.Text)
Error:
    If Err.Number <> 0 Then
        MsgBox "Sólo puede calcularle la raíz cuadrada a un número
positivo", vbCritical, "ERROR"
    End If
    Exit Sub
End Sub

```

```

Private Sub cmdIguar_Click()
    Select Case Flag
        Case "suma"
            C = A + Val(Text1.Text)
            Text1.Text = C
        Case "division"
            C = A / Val(Text1.Text)
            Text1.Text = C
        Case "multiplicacion"
            C = A * Val(Text1.Text)
            Text1.Text = C
        Case "resta"
            C = A - Val(Text1.Text)
            Text1.Text = C
    End Select
    Flag = ""
    A = 0
    C = 0
End Sub

```

Observe como desde el formulario se llama a los procedimientos Limpiar y Operación definidos en el módulo que anteriormente ha creado.

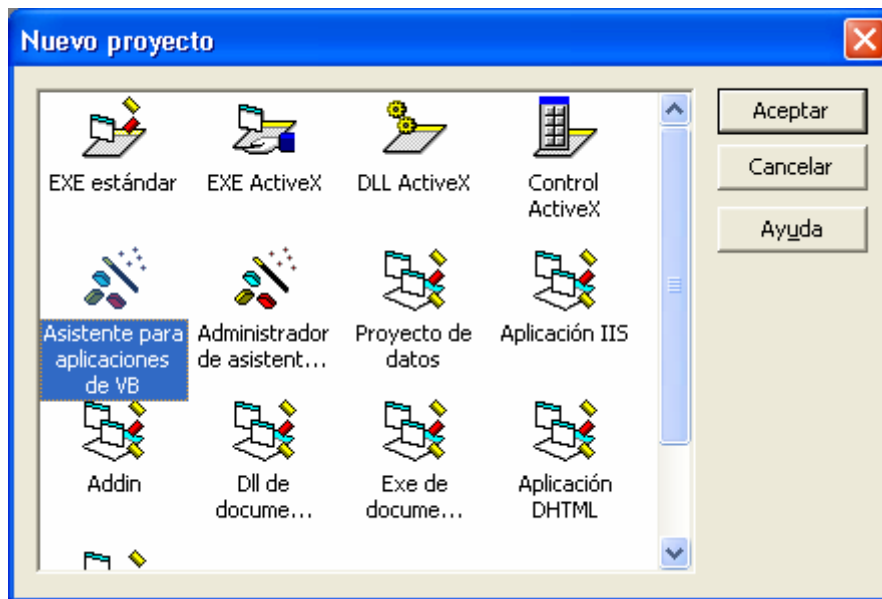
Al mismo tiempo dentro del formulario utilizamos las variables que había definido en el módulo, pudiendo usarlas gracias a que en el módulo fueron definidas como tipo Public.

## 5. Práctica 3. El Asistente para aplicaciones de VB

### 5.1. Introducción.

El Asistente para aplicaciones de Visual Basic permite generar una aplicación como si de una plantilla general se tratara. Además posteriormente y partiendo de lo que el asistente ha creado podremos personalizar la aplicación.

Podremos acceder al asistente al intentar crear un nuevo proyecto, desde el menú principal hacemos: Archivo - Nuevo proyecto, nos aparece el siguiente cuadro de diálogo:



Seleccione la opción Asistente para aplicaciones de VB. Nos aparecerá el asistente el cual esta compuesto por múltiples cuadros de diálogo en los que debe ir eligiendo y definiendo las características de la aplicación que va a realizar.

## 5.2. Crear un formulario

Como ejemplo para el uso del asistente vamos a crear un formulario que nos permita realizar el mantenimiento completo de los datos existentes en la tabla Artículos de la base de datos: Gestion Pedidos.

Nombre de la tabla	Tipo de datos	Descripción
CodArticulo	Texto	Código del artículo
Descripción	Texto	Descripción del artículo
Comentario	Texto	Comentario del artículo

### 5.2.1. Asistente para aplicaciones - Introducción

En la pantalla que nos aparece pulsar el botón Siguiente ya que en nuestro caso no tenemos ningún perfil desde el que cargar la configuración

### 5.2.2. Asistente para aplicaciones - Tipo de interfaz

El asistente presenta tres opciones:

- Interfaz de múltiples documentos (MDI): Consiste en una ventana principal que contiene visualmente ventanas secundarias.
- Interfaz de un único documento (SDI): Todas las ventanas de la aplicación existen a un mismo nivel
- De tipo explorador: Similar a SDI

Vaya seleccionando una a una cada una de ellas y observe que al eligiéndolas a la derecha del cuadro de diálogo nos va apareciendo la descripción de cada una de las opciones.

En nuestro caso elija la opción: Interfaz de un único documento (SDI) y ponga el nombre: "Gestion\_de\_pedidos" a la aplicación (no deje espacios en blanco en el nombre) y pulse Siguiente.

### 5.2.3. Asistente para aplicaciones - Menús

Aquí puede elegir los menús y submenús que va a tener su aplicación. En la práctica deje los que vienen por defecto. Pulse Siguiente.

### 5.2.4. Asistente para aplicaciones - Personalizar la barra de herramientas

Personalice la barra de herramientas principal que va a tener su aplicación. En la práctica deje los que vienen por defecto. Pulse Siguiente.

### 5.2.5. Asistente para aplicaciones - Recursos

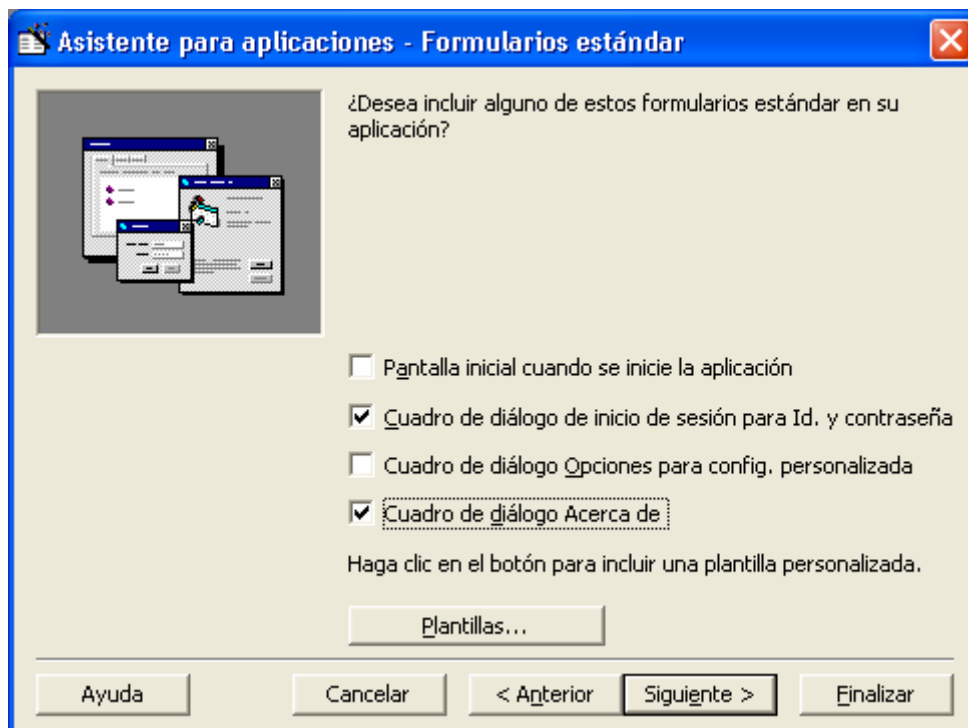
En la práctica seleccione No y pulse Siguiente

### 5.2.6. Asistente para aplicaciones - Conexión a internet

Le da la opción de tener un explorador Web personalizado en la aplicación que este realizando. En la práctica seleccione Sí y como URL predeterminada ponga: <http://www.tasi.us.es> y pulse Siguiente.

### 5.2.7. Asistente para aplicaciones - Formularios estándar

Puede elegir alguno de los formularios estándar que Visual Basic le ofrece. Para la práctica seleccione las opciones marcadas en la siguiente pantalla:



Pulse Siguiente.

### 5.2.8. Asistente para aplicaciones - Formulario de acceso a datos

En esta pantalla y en las secundarias asociadas a ella le indicaremos al Asistente que queremos realizar en nuestra aplicación una conexión con una base de datos.

Para la práctica pulse el botón Crear nuevo formulario...

Aparece asistente específico para formulario de datos con las siguientes pantallas que a continuación explicamos.

### 5.2.8.1. Asistente para formulario de datos - Pantalla Introducción

En primer lugar nos vuelve a preguntar por el perfil, vuelva a no seleccionar ninguno pulsando sobre Siguiente.

### 5.2.8.2. Asistente formulario de datos - Pantalla Tipo de base de datos



Seleccione la opción Access y pulse sobre siguiente.

### 5.2.8.3. Asistente para formulario de datos - Base de datos

Pulse sobre el botón Examinar... y seleccione la base de datos en donde están la tabla ó tablas a partir de las cuales queremos crear el formulario.

En la práctica elija la base de datos GestionPedidos y pulse el botón Siguiente.

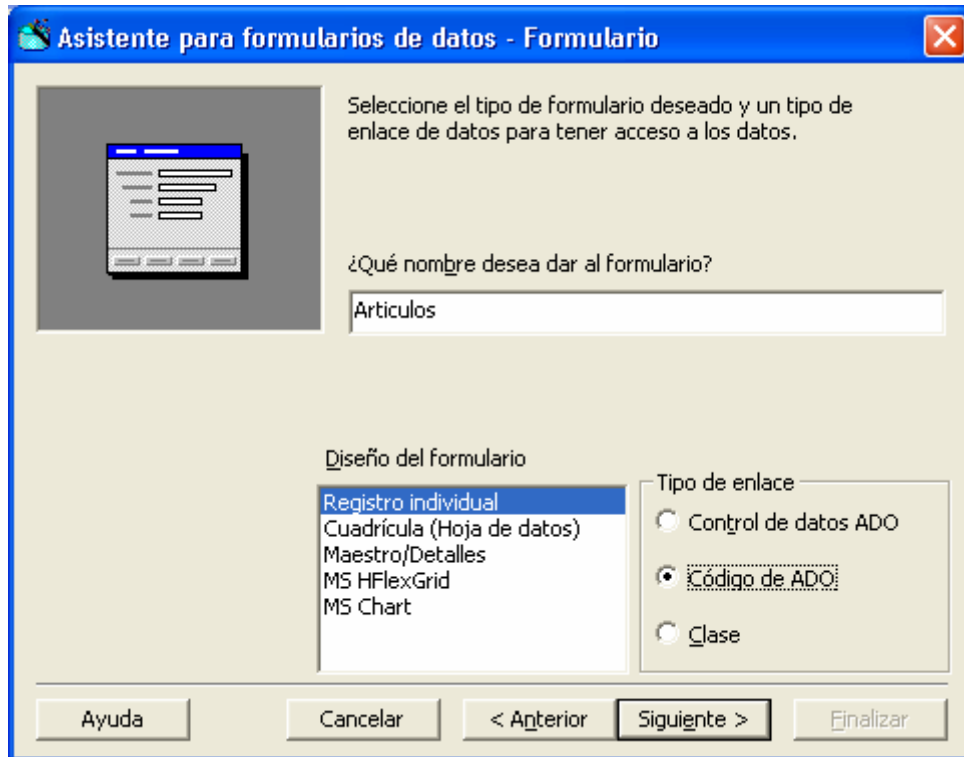
**Atención:** La base de datos de Access se ha de encontrar en una de las versiones soportadas por Visual Basic 6.0. En caso de dar error convertir la base de datos desde Access a una versión anterior a la actual, por ejemplo Access97.

Para acceder en Access a la opción convertir la base de datos hacer:

Menú principal - Herramientas – Utilidades de la base de datos –  
Convertir base de datos

#### 5.2.8.4. Asistente para formulario de datos - Formulario

Póngale el nombre “Articulos” al formulario y teniendo en cuenta que el formulario que quiere crear esta basado en una única tabla elija dentro de Diseño del formulario la opción Registro individual y como Tipo de enlace elija Código de ADO.



Pulse Siguiente.

#### 5.2.8.5. Asistente para formulario de datos - Origen de datos

En este cuadro de diálogo pasaremos la información al asistente del nombre de la tabla y los campos de la misma que queremos que aparezcan en el formulario.

En la práctica seleccione como origen de registro la tabla Articulos, como campos seleccionados pase todos los campos de la tabla y por último elija que los registros en el formulario se nos presenten ordenados por el campo CodArticulo.

Finalmente el cuadro de diálogo debe quedar:



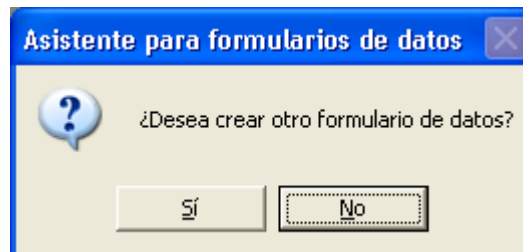
#### 5.2.8.6. Asistente para formulario de datos - Selección de controles

Aquí puede elegir los botones junto con sus funcionalidades que quiere que tenga el formulario.

Para la práctica y teniendo en cuenta que queremos realizar el mantenimiento completo de la tabla dejaremos marcadas todas las opciones disponibles y pulsaremos el botón Siguiete.

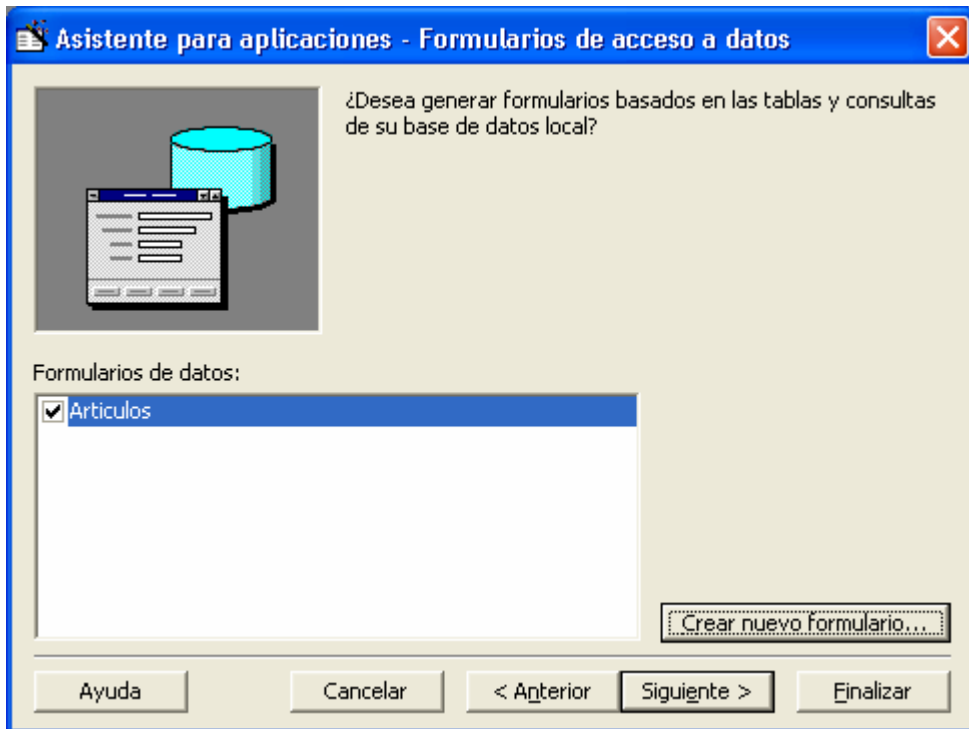
#### 5.2.8.7. Asistente para formulario de datos - Finalizado

Por último pulse el botón Finalizar. Aparecerá la pantalla:



Elija No.

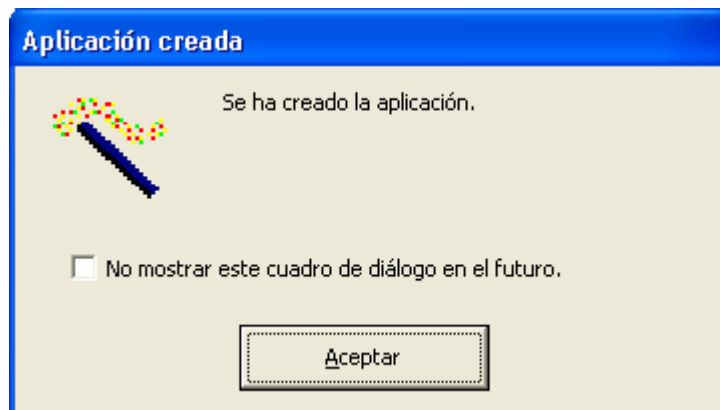
Le volverá a aparecer el cuadro de diálogo Formularios de acceso a datos, pero informando de que ya ha creado el Formulario de datos Artículos:



Pulse el botón Siguiete.

#### 5.2.8.8. Asistente para formulario de datos - Finalizado

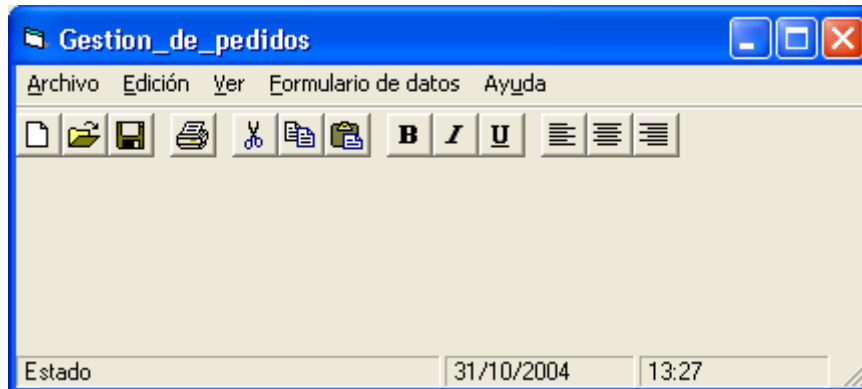
Pulse el botón Finalizar. Se le presenta el siguiente cuadro de diálogo.



Guarde la aplicación. Desde el menú principal pulse la opción: Archivo - Guardar proyecto. Tendrá que ir poniéndole nombre a cada uno de los formularios generados.

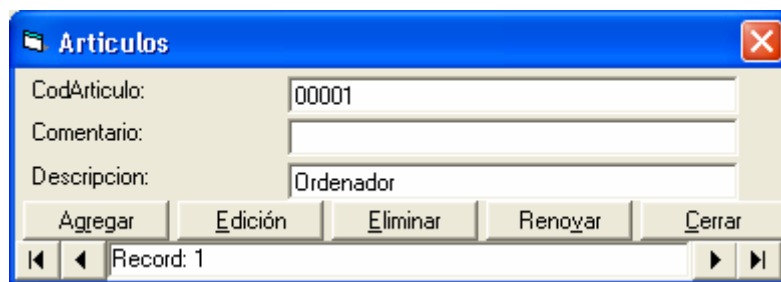
### 5.3. Resultado de la práctica

La aplicación resultante presenta el siguiente aspecto:



Observe que presenta un menú con las opciones que había seleccionado.

Eligiendo la opción Formulario de datos - Artículos, se nos presenta la pantalla con la que podremos realizar el mantenimiento de la tabla Artículos existente en la base de datos Gestion de Pedidos



La nueva aplicación que se ha creado puede ser modificada, personalizando cada una de sus opciones.

## 6. Práctica 4. Crear un formulario Maestro - Detalle

En la anterior práctica explicamos como crear una aplicación en Visual Basic formada por un menú general y un formulario asociado a una tabla de la base de datos usando el asistente para aplicaciones de Visual Basic.

Vamos a complicarlo un poco más, para ello en esta práctica vamos a realizar tres operaciones:

1. Creamos un nuevo formulario llamado Clientes, asociado a las tablas Clientes y Pedidos, que será del tipo Maestro-Detalle, esto es, el formulario obtendrá los datos de dos de las tablas de nuestra base de datos Gestion Pedidos Access97, mostrando los registros de la tabla detalle (Pedidos) asociados al registro activo de la maestro (Clientes).
2. Una vez creado este formulario, modificaremos el menú del proyecto anterior de forma que recoja una opción para poder acceder al nuevo formulario creado.
3. Por último agregaremos el formulario Clientes que acabamos de crear al proyecto realizado en la práctica anterior.

### 6.1. Tablas necesarias

Como ejemplo para el uso del asistente vamos a crear un formulario que nos permita realizar el mantenimiento completo de los datos existentes en la tabla Clientes de la base de datos: Gestion Pedidos. Mostrándonos a su vez en el mismo formulario los pedidos efectuados por el cliente activo en el formulario.

Nombre del Campo	Tipos de datos	Descripción
CodCliente	Texto	Código del Cliente
Nombre	Texto	Nombre del Cliente
DNI	Texto	DNI del cliente
Telefono	Número	Teléfono del cliente
Comentario	Memo	Comentarios sobre el cliente

Nombre del Campo	Tipos de datos	Descripción
CodPedido	Texto	Código de pedido
CodArticulo	Texto	Código de artículo
CodCliente	Texto	Código de cliente
NumArticulo	Número	Número de artículos del pedido
PrecioTotal	Número	Precio total del pedido
FechaPedido	Fecha/Hora	Fecha de realización del pedido
FechaPago	Fecha/Hora	Fecha del pago del pedido

Abra el Asistente para aplicaciones de Visual Basic.

### 6.2. Asistente para aplicaciones - Introducción

Pulsar el botón siguiente ya que en nuestro caso no tenemos ningún perfil desde el que cargar la configuración

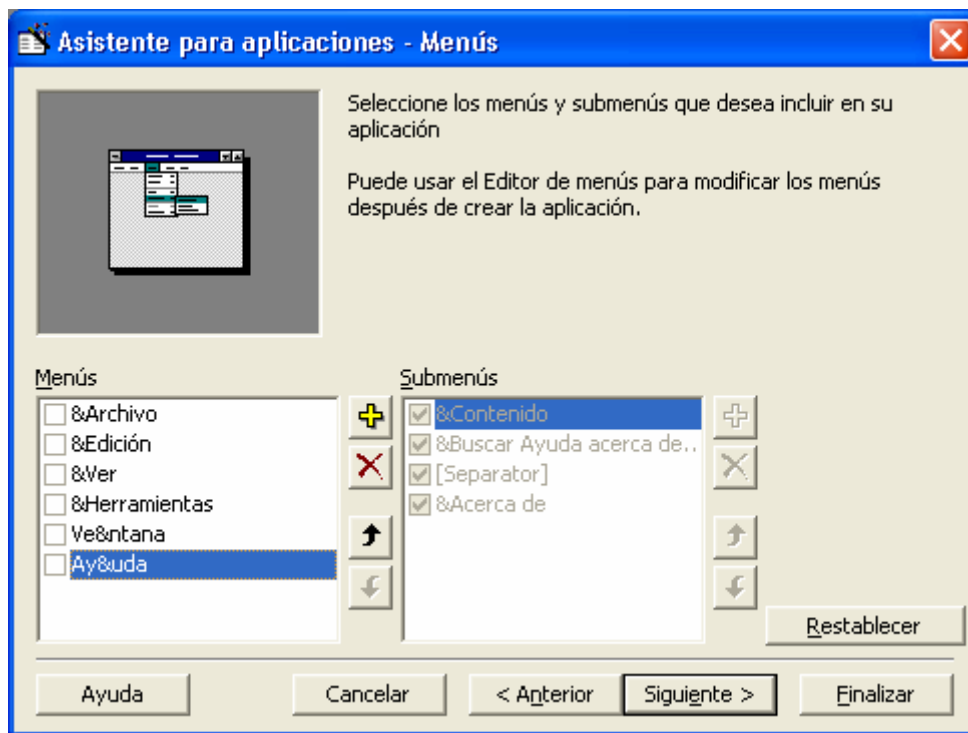
### **6.3. Asistente para aplicaciones - Tipo de Interfaz**

Elija la opción: Interfaz de un único documento (SDI) y ponga el nombre: "Gestion\_de\_clientes" a la aplicación (no deje espacios en blanco en el nombre) y pulse Siguiente.

### **6.4. Asistente para aplicaciones - Menús**

Como a nosotros no nos interesa crear una aplicación completa sino sólo crear un formulario desmarque todos.

Le debe quedar la pantalla como la siguiente:



### **6.5. Asistente para aplicaciones - Personalizar barra de herramientas**

Quite todas las opciones de la barra de herramientas y pulse Siguiete.

### **6.6. Asistente para aplicaciones - Recursos**

En la práctica seleccione No y pulse Siguiete.

### **6.7. Asistente para aplicaciones - Conexión a internet**

Seleccione No y pulse Siguiete.

### **6.8. Asistente para aplicaciones - Formulario estándar**

Para la práctica no seleccione las ninguna de las opciones. Pulse Siguiete.

### **6.9. Asistente para aplicaciones - Formulario de acceso a datos**

En esta pantalla y en las secundarias asociadas a ella le indicaremos al Asistente que queremos realizar en nuestra aplicación una conexión con una base de datos.

Para la práctica pulse el botón Crear nuevo formulario.

Aparece el asistente específico para formulario de datos con las siguientes pantallas que a continuación explicamos:

### **6.9.1. Asistente para formulario de datos - Introducción**

En primer lugar nos vuelve a preguntar por el perfil, vuelva a no seleccionar ninguno pulsando sobre Siguiente.

### **6.9.2. Asistente para formulario de datos - Pantalla tipo de base de datos**

Seleccione la opción Access y pulse sobre siguiente.

### **6.9.3. Asistente para formulario de datos - Base de datos**

Pulse sobre el botón Examinar... y seleccione la base de datos en donde están la tabla ó tablas a partir de las cuales queremos crear el formulario.

En la práctica elija la base de datos Gestión Pedidos y pulse el botón Siguiente.

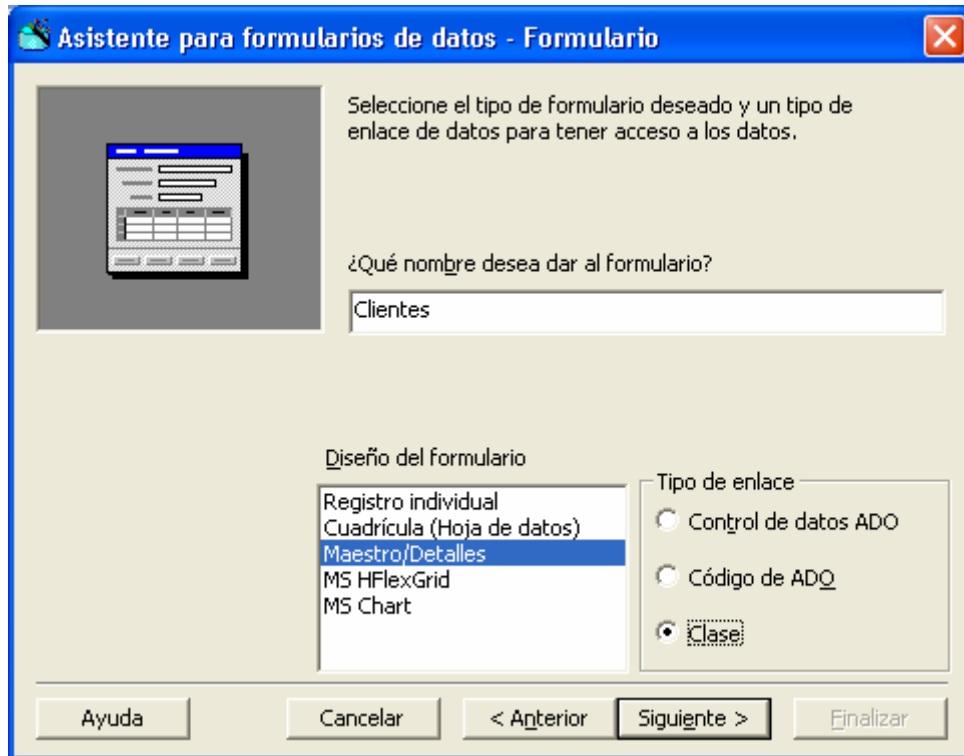
**Atención:** La base de datos de Access se ha de encontrar en una de las versiones soportadas por Visual Basic 6.0. En caso de dar error convertir la base de datos desde Access a una versión anterior a la actual, por ejemplo Access97.

Para acceder en Access a la opción convertir la base de datos hacer:

Menú principal - Herramientas – Utilidades de la base de datos –  
Convertir base de datos

#### 6.9.4. Asistente para formulario de datos - Formulario

Póngale el nombre “Clientes” al formulario y teniendo en cuenta que el formulario que quiere crear esta basado en una única tabla elija dentro de Diseño del formulario la opción Maestro/Detalles y como Tipo de enlace elija Clase.



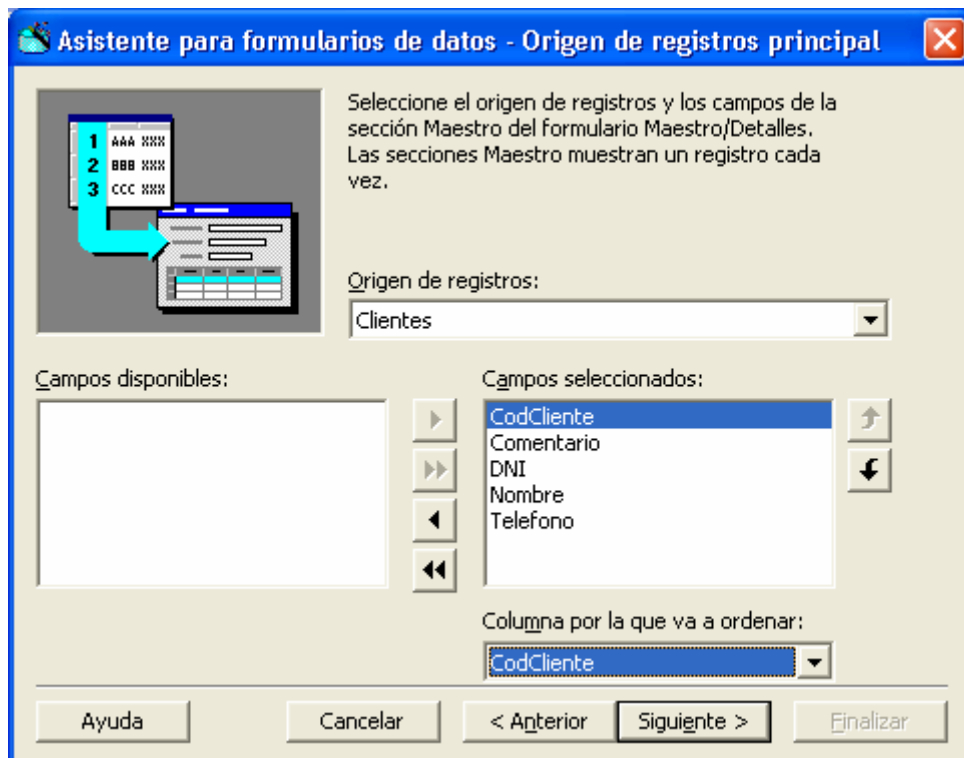
Pulse Siguiente.

### 6.9.5. Asistente para formulario de datos - Origen de registros principal

En este cuadro de diálogo pasaremos la información al asistente del nombre de la tabla principal de la relación Maestro/Detalle, es decir, la tabla Maestro Clientes en nuestro caso y los campos de la misma que queremos que aparezcan en el formulario.

En la práctica seleccione como origen de registro la tabla Clientes, como campos seleccionados pase todos los campos de la tabla y por último elija que los registros en el formulario se nos presenten ordenados por el campo CodCliente.

Finalmente el cuadro de diálogo debe quedar:



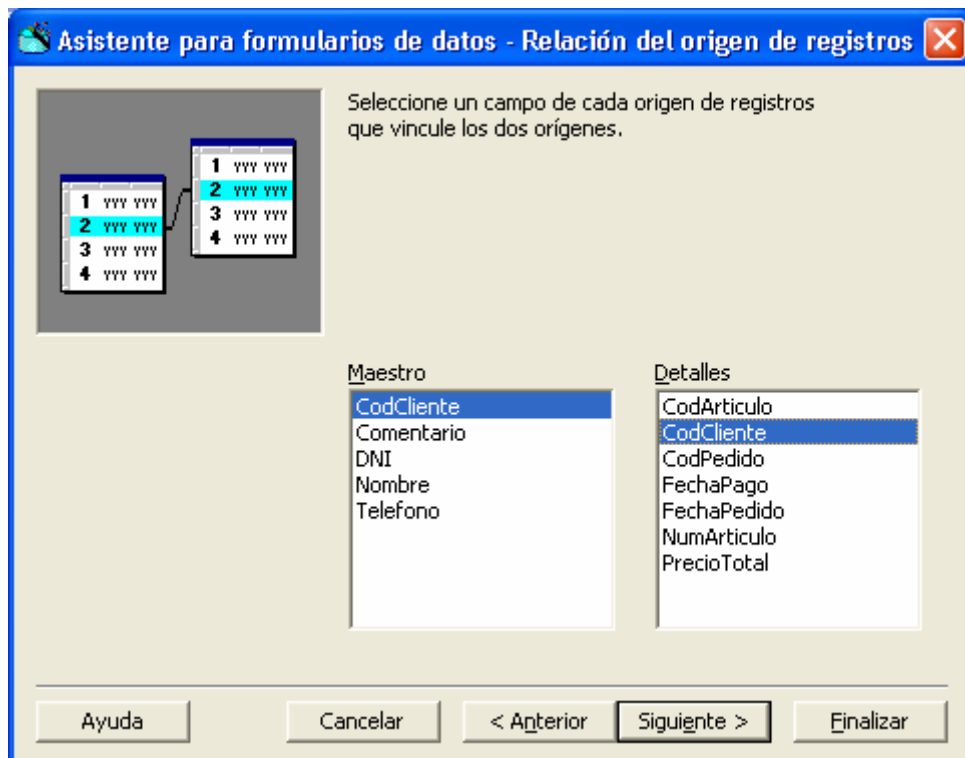
Pulse Siguiete.

### 6.9.6. Asistente formulario de datos - Relación del origen de registros

La relación Maestro/Detalle entre dos tablas viene definida por los dos campos que hay relacionados entre ambas tablas.

En esta pantalla indicaremos dicha relación para ello seleccionaremos los campos existentes en ambas tablas que las relacionan.

En la práctica elija en ambas tablas CodCliente



Pulse Siguiete

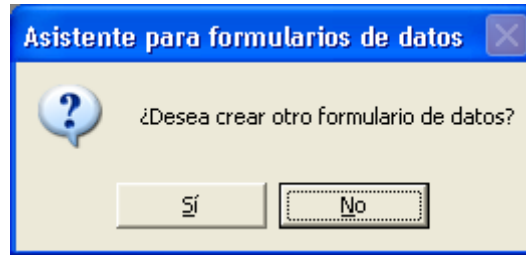
### 6.9.7. Asistente para formulario de datos - Selección de controles

Aquí puede elegir los botones junto con sus funcionalidades que quiere que tenga el formulario.

Para la práctica y teniendo en cuenta que queremos realizar el mantenimiento completo de la tabla dejaremos marcadas todas las opciones disponibles y pulsaremos el botón Siguiete.

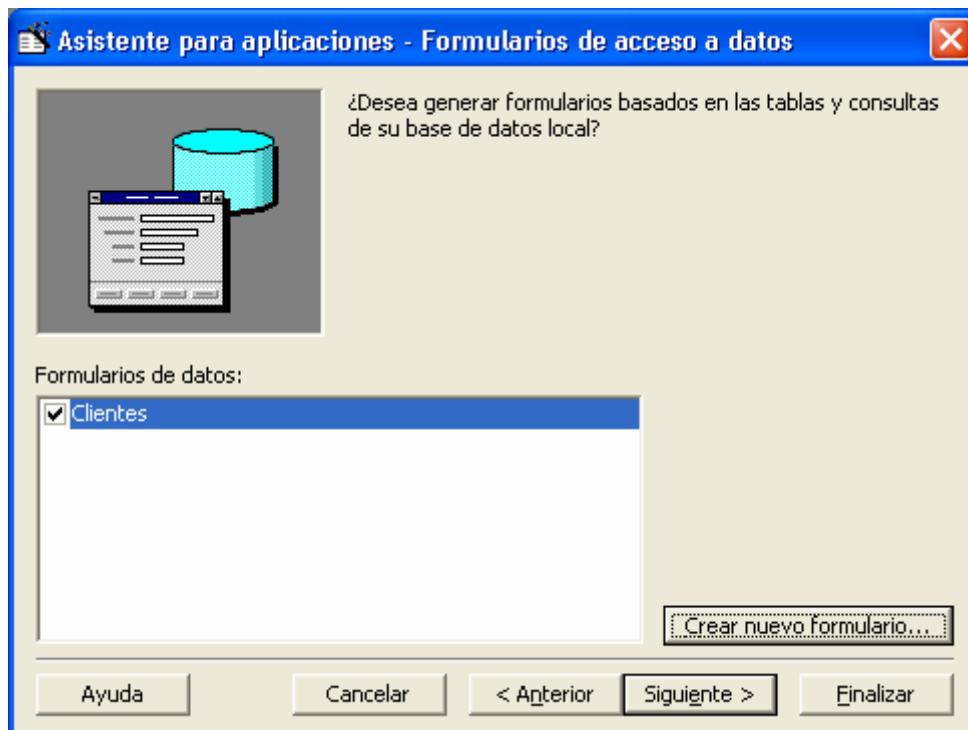
### 6.9.8. Asistente para formulario de datos - Finalizado

Por último pulse el botón Finalizar. Aparecerá la pantalla:



Elija No.

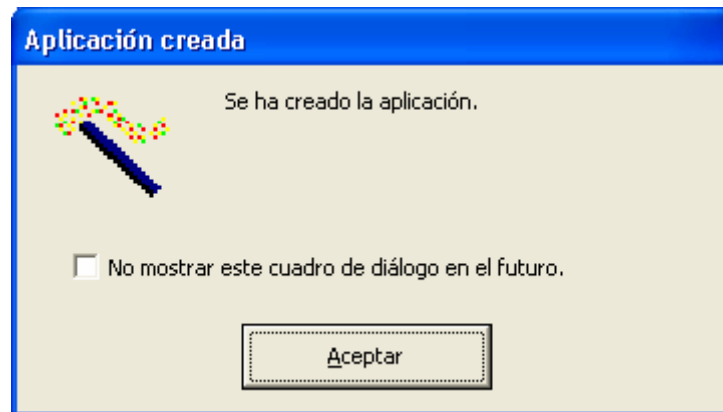
Le volverá a aparecer el cuadro de diálogo Formularios de acceso a datos, pero informando de que ya ha creado el Formulario de datos Artículos



Pulse el botón Siguiete.

## 6.10. Asistente para aplicaciones - Finalizado

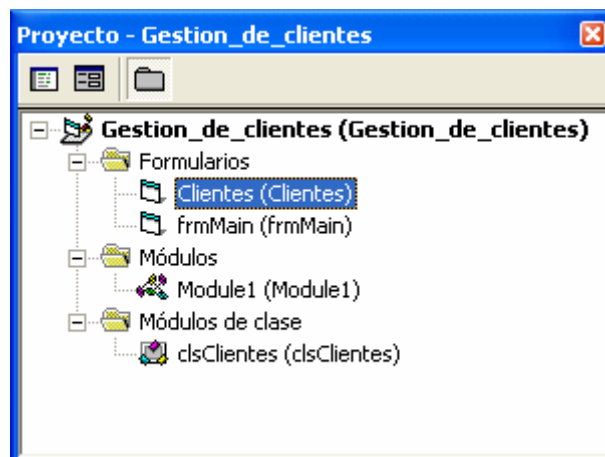
Pulse el botón Finalizar. Se le presenta el siguiente cuadro de diálogo:



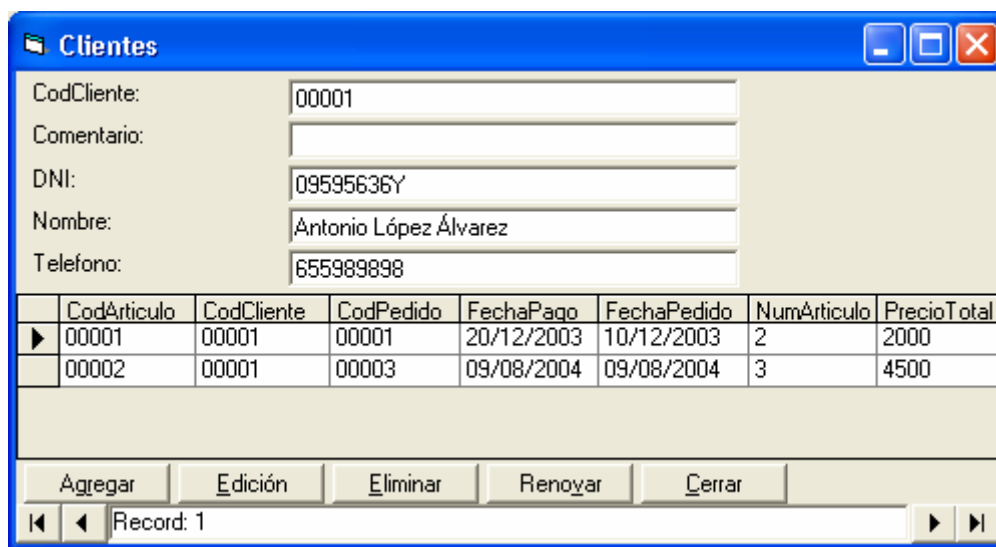
Guarde la aplicación. Desde el menú principal pulse la opción: Archivo - Guardar proyecto.

Se han creado:

- Dos formularios llamados:
  - o Clientes
  - o frmMain
- Un módulo llamado Module1
- Un módulo de Clase llamado clsClientes



El formulario Clientes creado presentará el siguiente aspecto:



The screenshot shows a window titled 'Clientes' with a blue title bar. It contains a form with the following fields:

- CodCliente: 00001
- Comentario: (empty)
- DNI: 09595636Y
- Nombre: Antonio López Álvarez
- Telefono: 655989898

Below the form is a table with the following data:

	CodArticulo	CodCliente	CodPedido	FechaPago	FechaPedido	NumArticulo	PrecioTotal
▶	00001	00001	00001	20/12/2003	10/12/2003	2	2000
	00002	00001	00003	09/08/2004	09/08/2004	3	4500

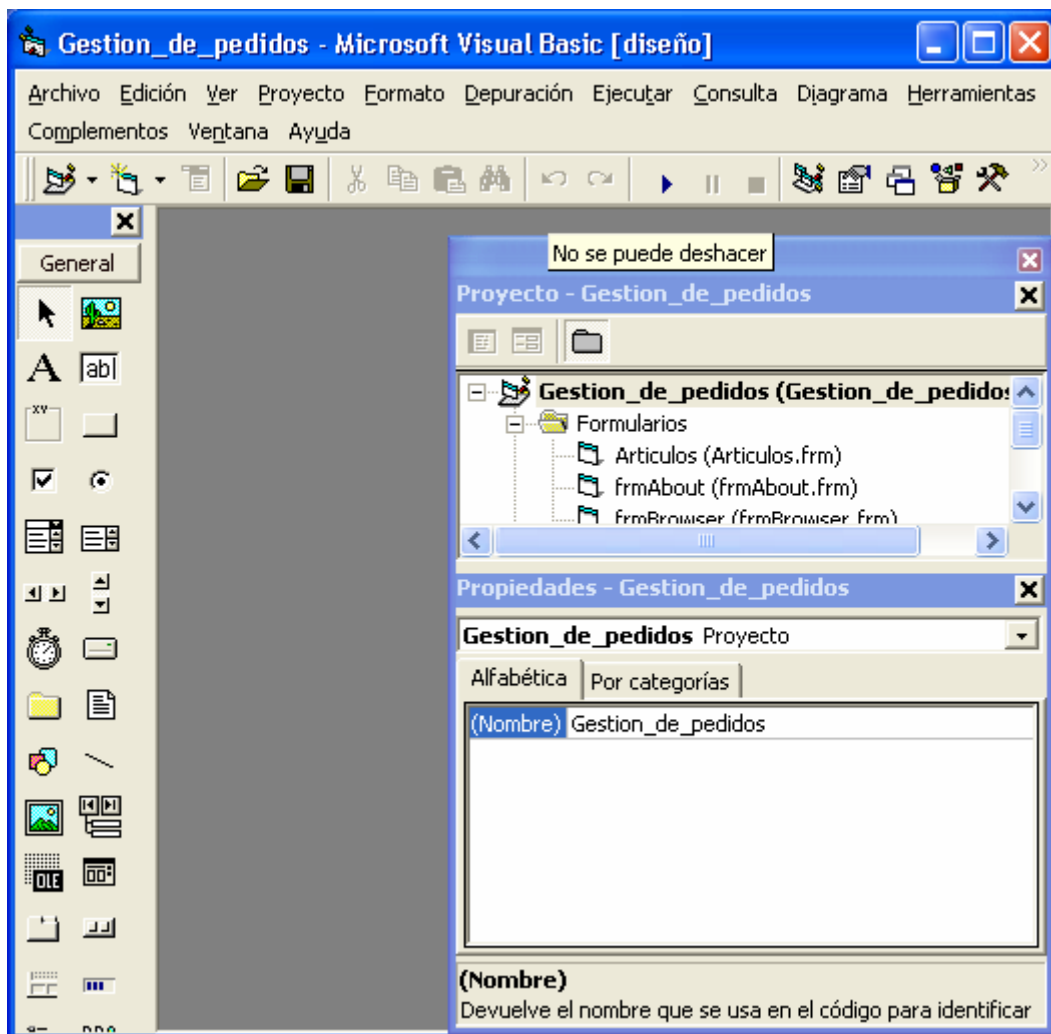
At the bottom of the window, there are five buttons: 'Agregar', 'Edición', 'Eliminar', 'Renovar', and 'Cerrar'. Below the buttons is a record navigation bar showing 'Record: 1' and navigation arrows.

Según la pantalla arriba mostrada el cliente con el código 00001 habrá efectuado 2 pedidos diferentes.

## 7. Modificar el menú del proyecto Gestion\_de\_pedidos

En la práctica anterior creamos una aplicación con un menú desde el que se podía tener acceso a un formulario que permitía el mantenimiento de la tabla Artículos existente en la base de datos Gestion Pedidos. Vamos a modificar el menú de dicha aplicación para que presente también la opción de acceso al formulario Clientes que acabamos de crear.

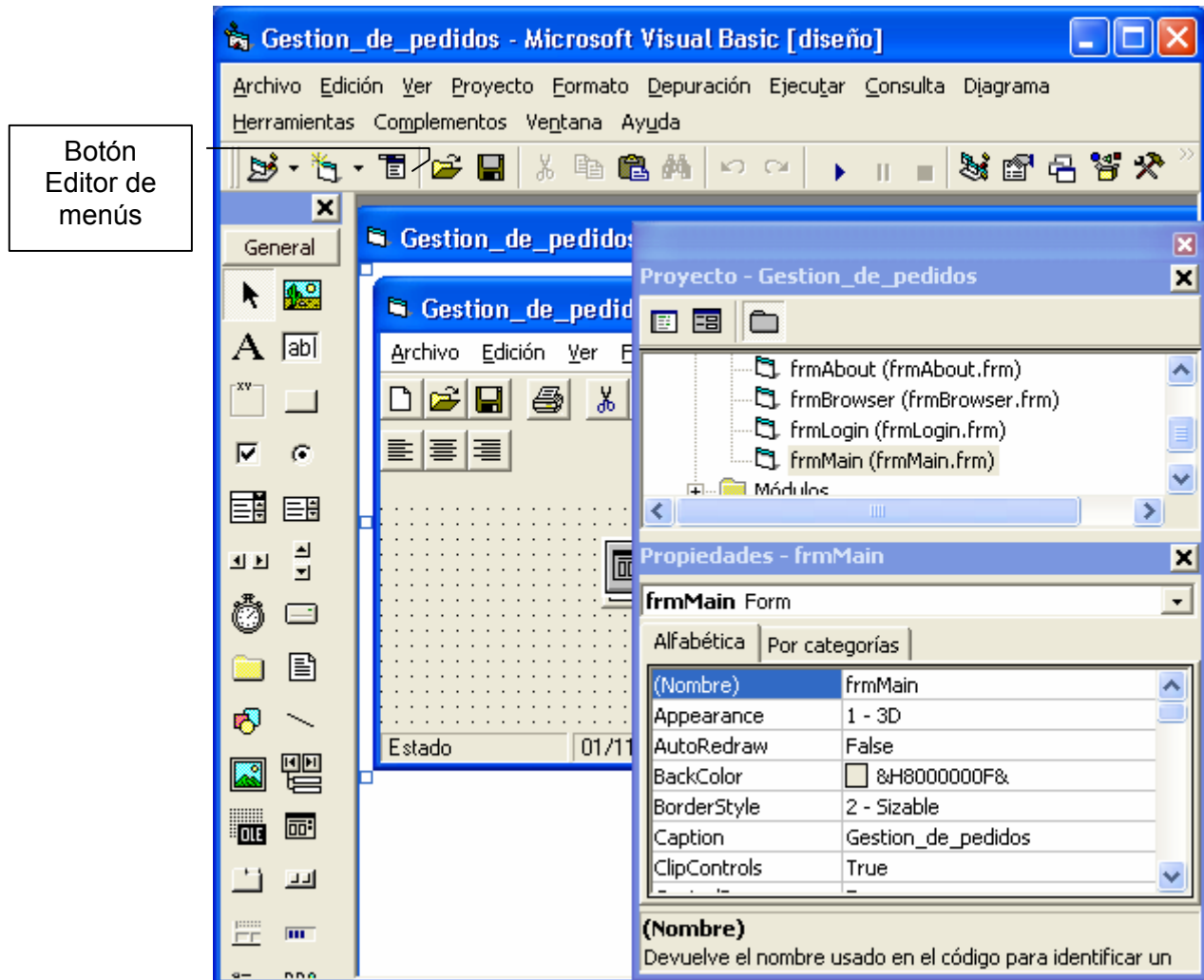
Para ello en primer lugar abra en Visual Basic el proyecto Gestion\_de\_pedidos desde el menú principal de VB pulse Archivo - Abrir proyecto y selecciónelo, le debe aparecer:



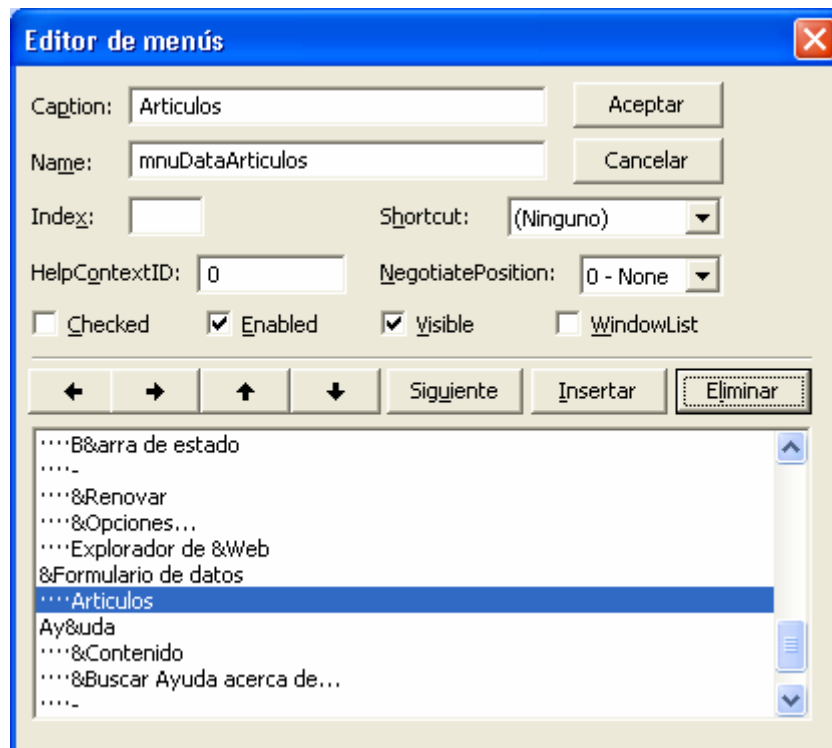
En la ventana Proyecto - Gestion\_de\_pedidos seleccione el formulario frmMain.frm o en su defecto aquel formulario que contenga el menú de la aplicación.

Tras seleccionarlo pulse el botón Ver objeto situado igualmente en la ventana Proyecto - Gestion\_de\_pedidos.

A continuación se activará el botón Editor de menús situado el tercero por la izquierda en la barra de herramientas del entorno de desarrollo.



Púlselo. Nos aparece el cuadro de diálogo Editor de Menús, con el podremos crear o modificar los menús que necesitemos para nuestras aplicaciones.



**Nota:** Un menú puede visualizar órdenes y submenús. Cada opción de menú puede disponer hasta un total de cuatro niveles de submenús.

Para la práctica nos interesa ampliar el submenú Formulario de datos en el que queremos agregar una nueva opción. Para ello seleccione el submenú Articulos y pulse el botón Insertar. Nos aparece una nueva opción rellene las propiedades de dicha opción dando los siguientes valores:

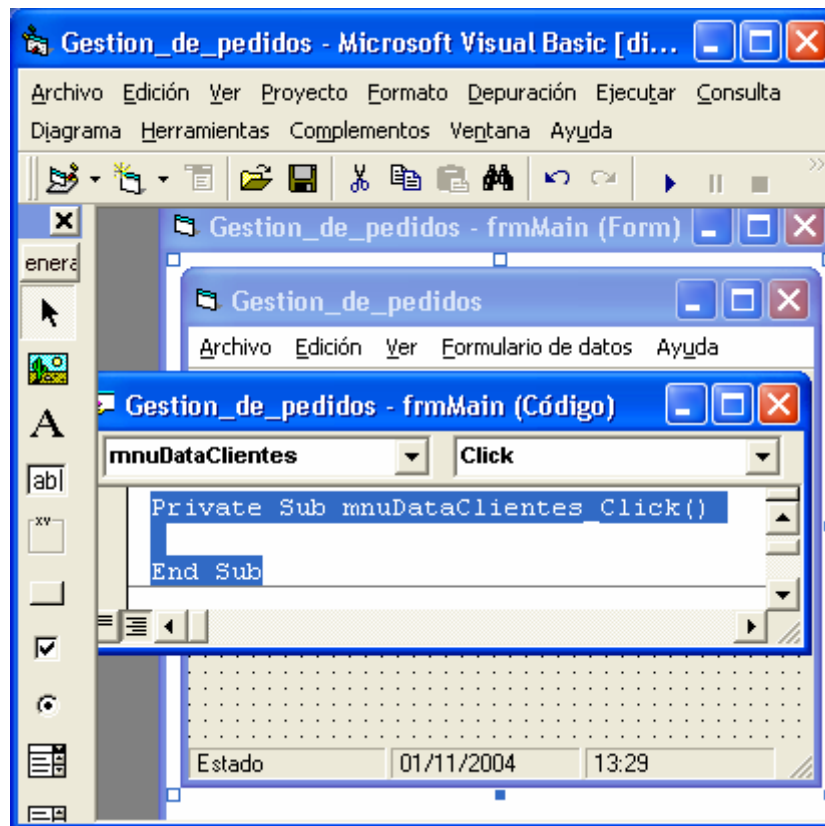
Caption: Clientes  
Name: mnuDataClientes

Pulse el botón Aceptar.

Caso de querer insertar una línea que separe las diferentes opciones del menú se consigue escribiendo en la propiedad Caption de una nueva opción un guión (-) sin olvidarse de darle un nombre (por ejemplo Linea1) a la propiedad Name. Los nombres de las líneas de separación han de ser diferentes en el sistema de menús para cada línea separadora.

Observe que en la ventana de desarrollo si pulsamos sobre la opción Formulario de datos existente en el menú en el formulario frmMain que hemos creado, nos aparece la nueva opción Clientes que acabamos de crear. Pulse sobre ella.

Nos saldrá la pantalla del código del formulario en el que ha creado el Menú de la aplicación, presentando el siguiente aspecto:



Modifiquemos el código poniéndolo:

```
Private Sub mnuDataClientes_Click()  
    Clientes.Show  
End Sub
```

Para que al pulsar sobre la opción de menú Clientes se nos presente el formulario Clientes. Guarde los cambios efectuados en el proyecto.

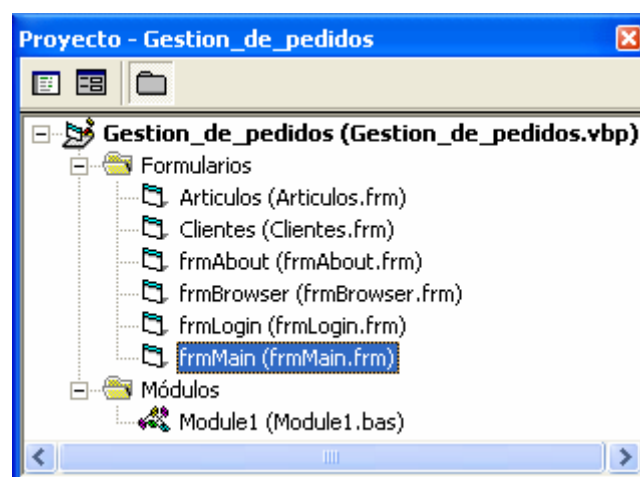
## 8. Agregar el formulario Clientes al Proyecto Gestion\_de\_pedidos

Para agregar el formulario Clientes abra el proyecto Gestion\_de\_pedidos. En la barra de herramientas principal pulse la opción Proyecto - Agregar formulario. Le aparece la ventana:



Pulse sobre la opción Existente, busque el formulario Clientes, selecciónelo y pulse abrir.

Observe que en la ventana Proyecto - Gestion\_de\_pedidos aparece el formulario que acaba de agregar:



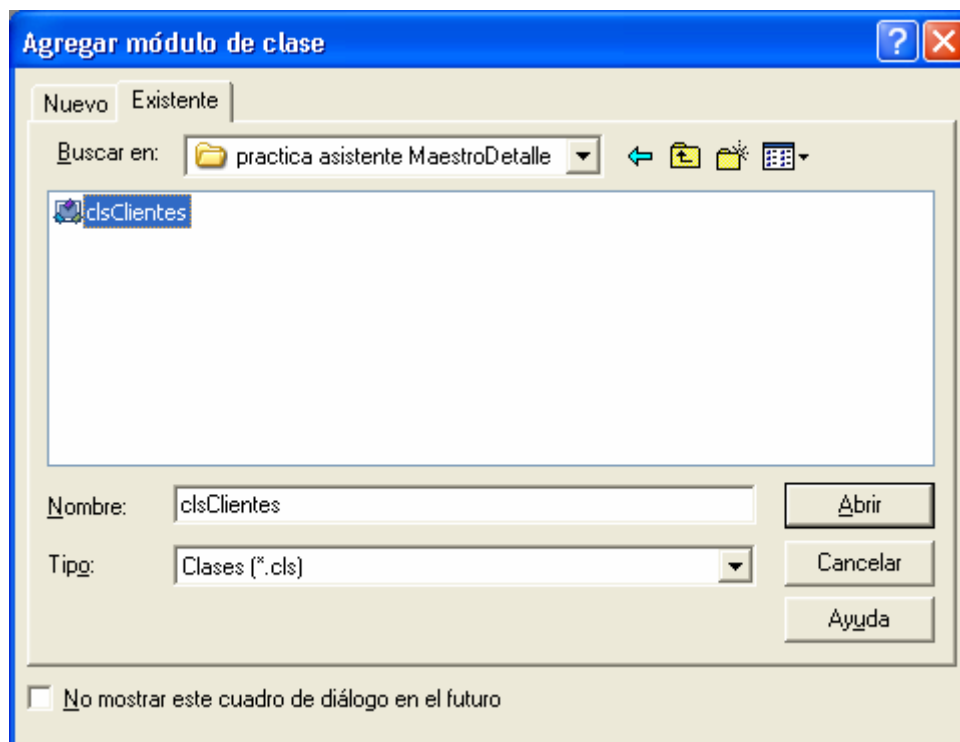
Pero el formulario Clientes necesitaba para su funcionamiento el módulo Module1.bas y el módulo de clase clsClientes.cls.

En el proyecto Gestion\_de\_pedidos ya existe el módulo Module1.bas y además tiene el mismo código que el que necesita el formulario Clientes por lo que no será

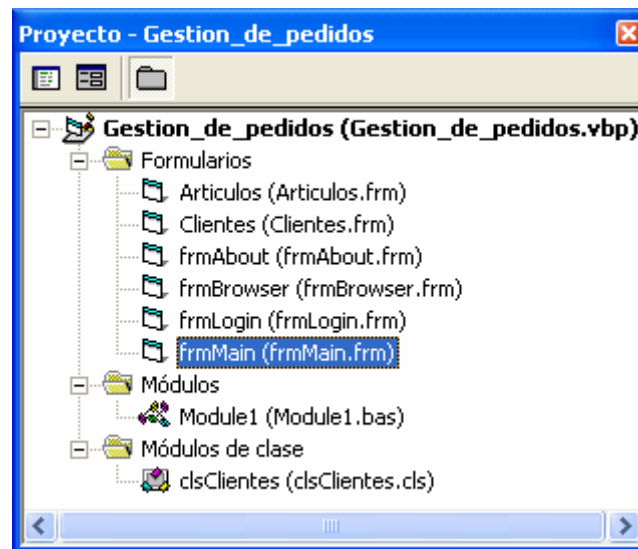
necesario que lo agregemos al proyecto. En cambio el módulo de clase clsClientes.cls si lo necesitamos agregar para que el formulario Clientes funcione en el proyecto Gestion\_de\_pedidos.

Para agregar el módulo de clase desde el menú principal de Visual Basic pulse la opción Proyecto - Agregar módulo de clase.

Pulse sobre la opción Existente, busque el módulo de clase clsClientes.cls, selecciónelo y pulse abrir:



Observe que en la ventana Proyecto - Gestion\_de\_pedidos aparece el módulo de clase que acaba de agregar:



Guarde los cambios efectuados en el proyecto desde la opción del menú principal: Archivo - Guardar Proyecto.

## 9. Práctica 5. Crear Formulario de Mantenimiento de una tabla

Si queremos realizar una aplicación que en un momento determinado de su ejecución necesite aceptar nuevos datos o bien visualizarlos para poder continuar, debemos crear un nuevo formulario con los controles que permitan esas operaciones. En estos casos, los formularios reciben el nombre de cajas de diálogo. Hay tres formas de añadir cajas de diálogo a una aplicación:

- Cajas de diálogo personalizadas: son cajas de diálogo hechas a medida, que se consiguen agregando controles a un formulario.
- Cajas de diálogo predefinidas: son las creadas por medio de las funciones InputBox y MsgBox.
- Cajas de diálogo comunes: son las típicas cajas de diálogo de aplicaciones para Windows, por ejemplo, la caja de diálogo de Abrir o Imprimir.

Visual Basic tiene toda una serie de controles para ser utilizados como mecanismos de entrada/salida (E/S). Por ejemplo: las cajas de texto, los botones de pulsación, las casillas de verificación, botones de opción, listas desplegables, barras de desplazamiento, etc.

Tanto los formularios como los controles que podamos crear en ellos tienen asociados una serie de eventos propios de cada tipo de objeto.

Cada evento tiene relación con una determinada acción que puede ocurrir en o con el formulario (por ejemplo: se carga el formulario, se pulsa un botón, etc). Pero estos eventos sólo desencadenarán una respuesta si hemos escrito código para ellos.

### 9.1. Objetivos

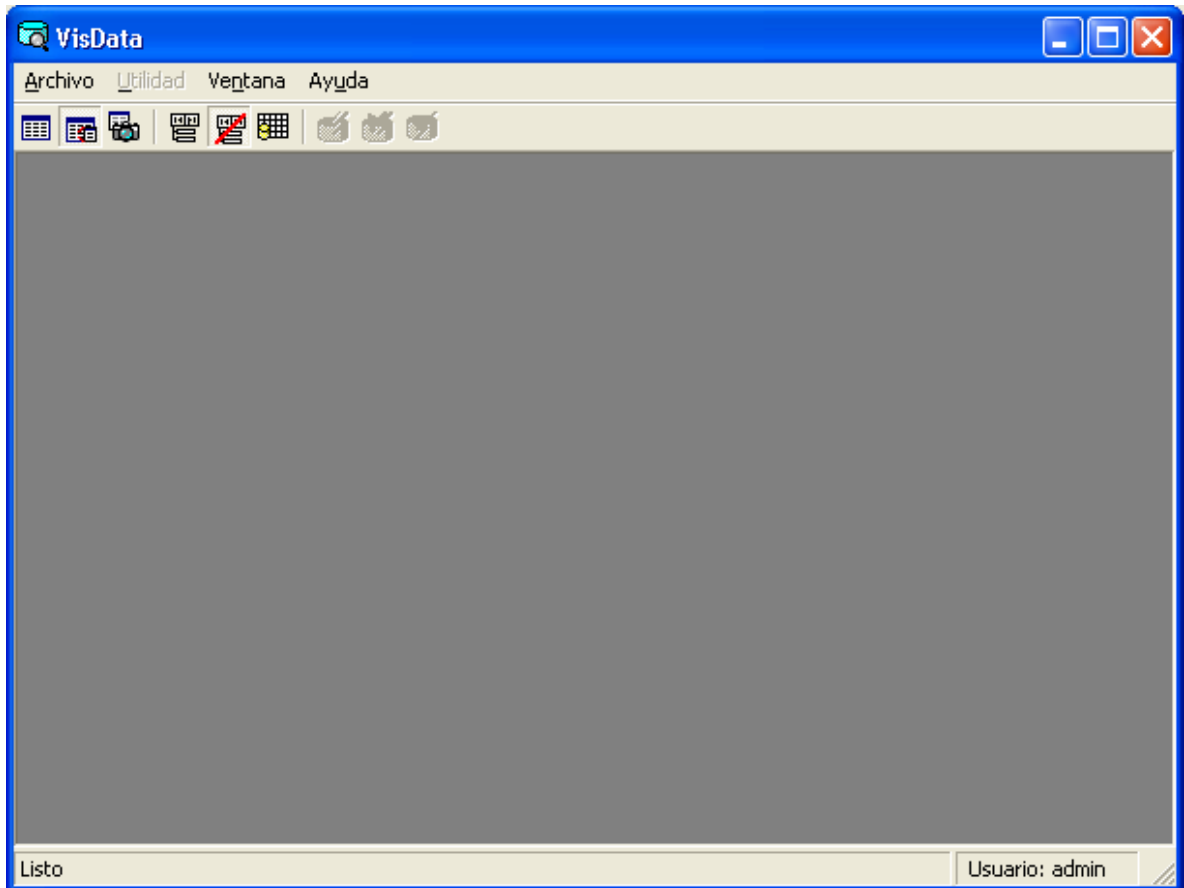
Los objetivos de esta práctica son:

- La modificación de la base de datos Gestion Pedidos, agregándole una tabla llamada Provincias que vamos a crear desde el Administrador visual de base de datos de Visual Basic.
- La creación de formulario de mantenimientos de dicha tabla Provincias, que guardará los nombres referentes a todas las provincias españolas.

## 9.2. El Administrador Visual de Base de Datos de Visual Basic

Visual Basic incorpora el administrador visual de datos con el que podrá crear y establecer las propiedades de las tablas que conformen sus bases de datos.

Para acceder al administrador visual de datos utilice la opción de mismo nombre situada en el menú principal dentro de la opción Complementos. Nos saldrá la siguiente pantalla:

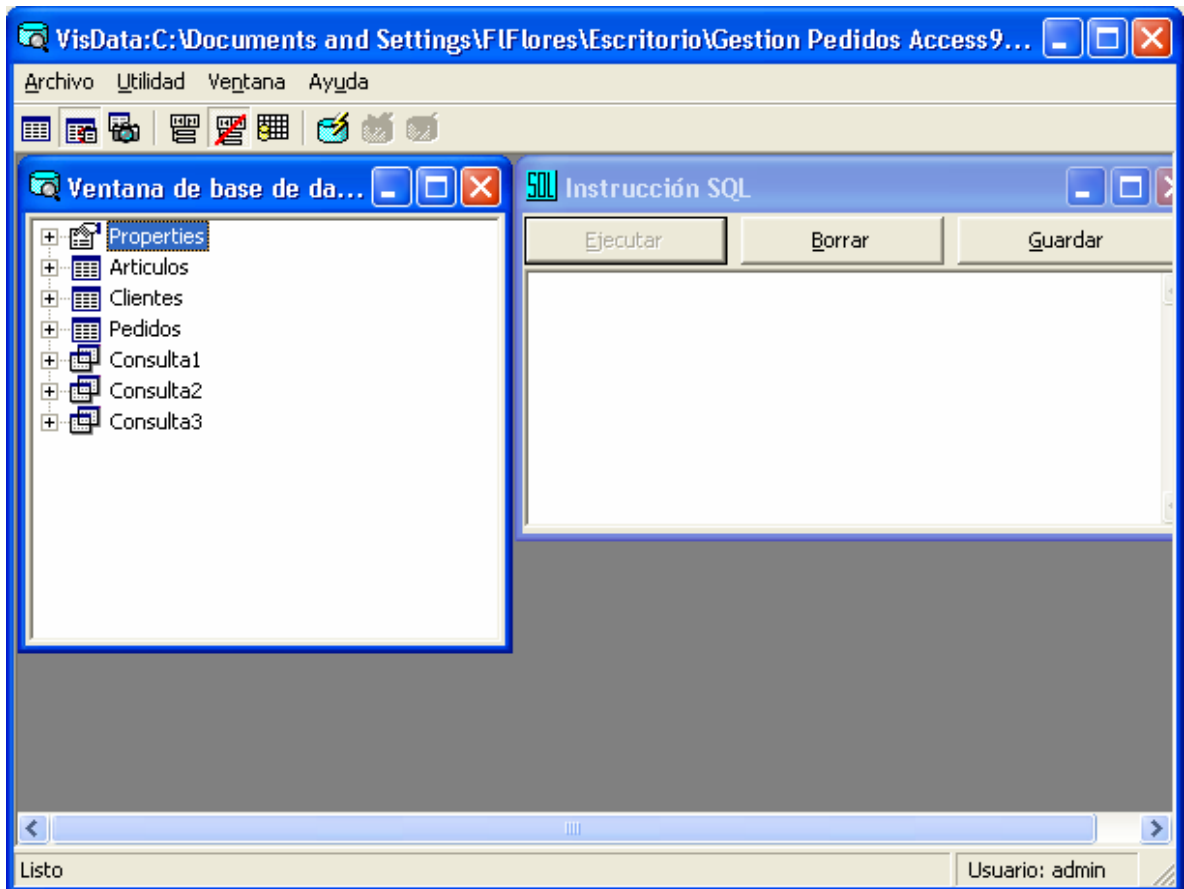


El administrador permite crear una base de datos nueva o modificar una existente.

En nuestro caso dentro del menú del administrador visual pulsemos sobre la opción:

Archivo - Abrir base de datos - Microsoft Access

Tras ello seleccione la base de datos Gestion Pedidos. Nos debe de aparecer la siguiente pantalla:



Para crear la nueva tabla pulsamos el botón derecho del ratón dentro de la ventana de base de datos y elegimos la opción nueva tabla. Nos aparecerá el cuadro de diálogo Estructura de tabla.

Damos el nombre deseado a la tabla y con el botón Agregar campo vamos agregando los campos que queremos que tenga la tabla.

Finalmente el aspecto que nos presentará el cuadro de diálogo es:

**Estructura de tabla**

Name de tabla: Provincias

Lista de campos:

- CodProvincia
- Nombre**
- Descripcion

Nombre: Nombre

Type: Text  FixedLength

Size: 50  VariableLength

CollatingOrder: 1024  AutoIncrement

AllowZeroLength

OrdinalPosition: 0  Required

ValidationText:

ValidationRule:

DefaultValue:

Agregar campo    Quitar campo

Lista de índices:

Primary     Unique     Foreign

Necessary     IgnoreNull

Fields:

Generar la tabla    Cerrar

Para finalizar la creación de la tabla pulse sobre el botón Generar la tabla existente en el cuadro de diálogo en el que nos encontramos.

Observemos como en el Administrador visual de datos nos aparece la nueva tabla Provincias ya creada.

## 9.3. Crear el Formulario de Mantenimiento de la tabla Provincias

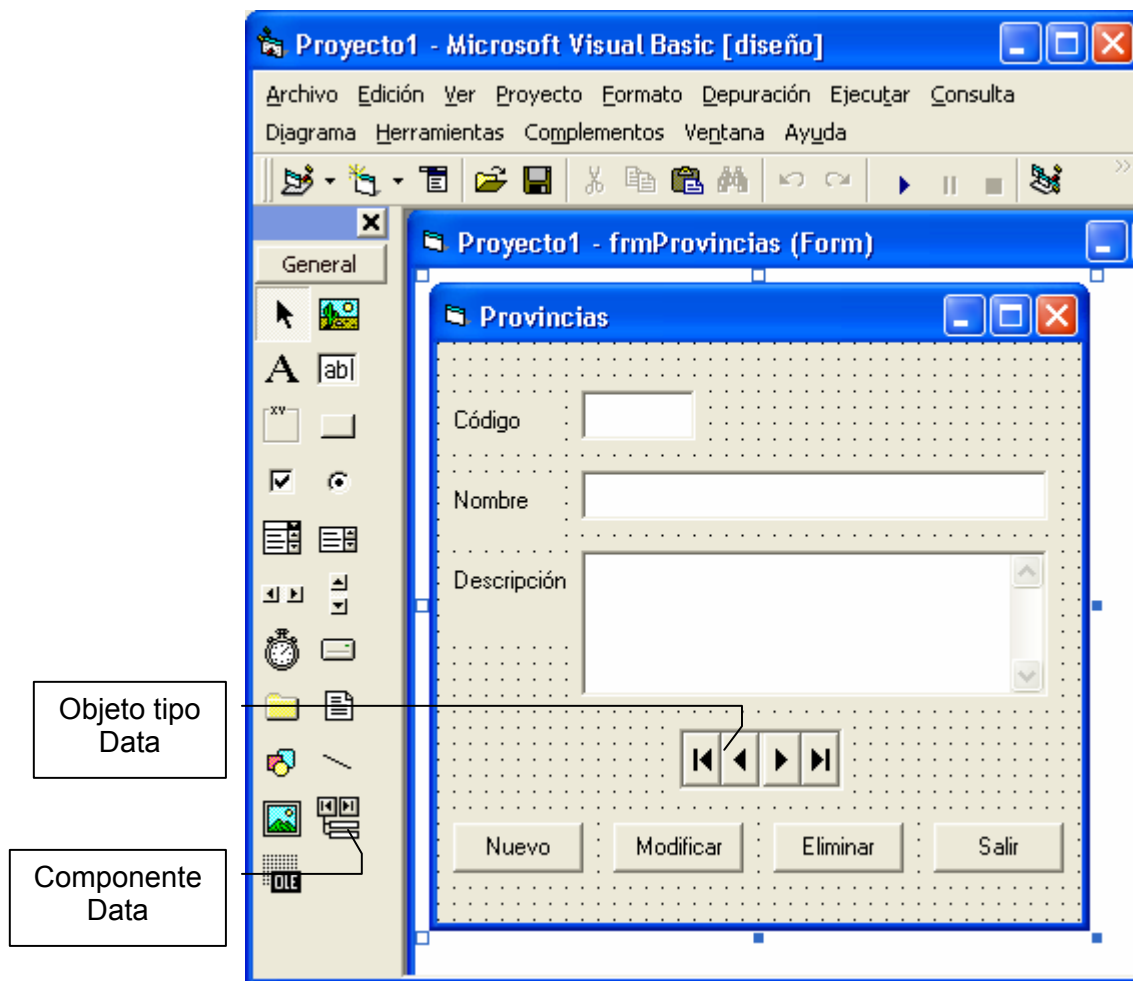
### 9.3.1. Creación del proyecto

Creemos un nuevo proyecto en Visual Basic, para ello en el menú principal seleccionemos:

Archivo - Nuevo proyecto

### 9.3.2. Objetos del formulario

Por defecto el proyecto se nos ha creado con un formulario. Modifiquemos dicho formulario hasta que presente el siguiente aspecto:



Los objetos tipo Data son intermediarios que conectan los controles que gestionan, reciben o envían, los datos y los diferentes elementos de la base de datos que se conectan a través de ella. Para ello:

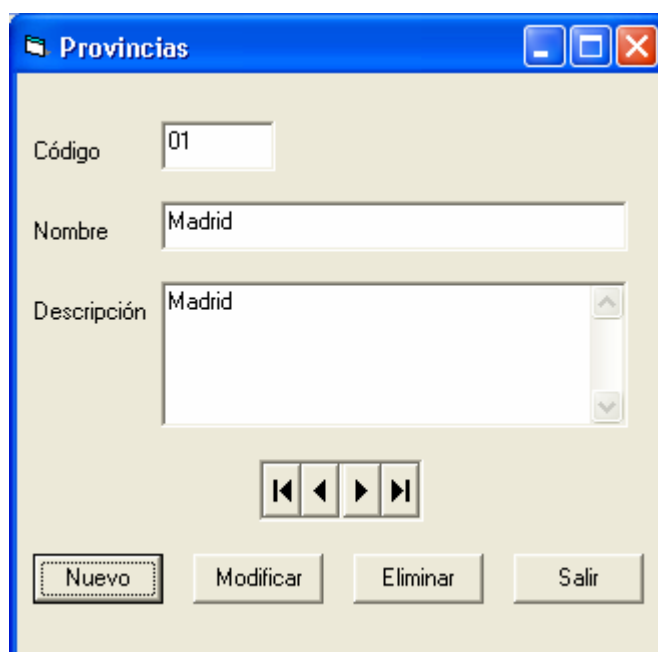
- Si está en una base de datos de Microsoft Access, no es preciso activar la propiedad Connect, si es de otro gestor de bases de datos se deberá especificar este gestor como valor de la propiedad antes de nada.
- El control Data deberá activar su propiedad DataBaseName con la ruta y el nombre de la base de datos a la que se quiera conectar. En nuestro caso seleccione la base datos: Gestion de Pedidos.

- La conexión a una de las tablas de la base de datos se consigue activando la propiedad RecordSource. Antes de hacerlo se debe haber indicado el valor a la propiedad DataBaseName. Para la práctica que estamos realizando, seleccione la tabla Provincias.

Pero el control Data sólo consigue establecer el enlace con la base de datos y la tabla que se ha elegido para el formulario en el que estamos trabajando. Para enlazar con los campos de los registros de la tabla Provincias, deberemos activar dos de las propiedades de los componentes cajas de texto que hemos insertado en nuestro formulario para tal fin. Estas propiedades son:

- DataSource: donde indicará el nombre del control Data. En nuestra práctica será Data1, si es que no lo ha modificado en cada una de las tres cajas de texto que hemos creado en el formulario con el fin de que muestre los datos existentes en la tabla Provincias.
- DataField: debe seleccionar el nombre del campo al que se enlaza la caja de texto. Para la práctica introduzca el campo correspondiente en cada una de las tres cajas de texto existentes.

El resultado que se ha debido conseguir debe ser parecido a:



Observe como se puede desplazar por los registros existentes en la tabla usando el control Data.

### 9.3.3. Código del formulario

Aún nos falta programar el funcionamiento de los botones creados, vamos a realizarlo.

Vamos a necesitar una variable tipo Integer así que en la parte del código fuente dedicada a las declaraciones pondremos:

```
Dim respuesta As Integer
```

Veamos ahora cuales serían los códigos de cada uno de los botones que hemos creado.

El código fuente asociado al botón Nuevo:

```
Private Sub cmdNuevo_Click()  
On Error GoTo Error  
If Text1.Text <> "" And Text2.Text <> "" Then  
Data1.Refresh  
Data1.Recordset.AddNew  
Data1.Recordset.Update  
Else  
MsgBox "Debe introducir el código y el nombre de la provincia", vbCritical,  
"Error en Alta"  
End If  
Error:  
If Err.Number <> 0 Then  
MsgBox "Descripción del Error: " & Err.Description, vbCritical,  
"ERROR"  
End If  
Exit Sub  
End Sub
```

El código fuente asociado al botón Modificar:

```
Private Sub cmdModificar_Click()  
If Text1.Text <> "" And Text2.Text <> "" Then  
Data1.Recordset.Edit  
Else  
MsgBox "Debe introducir el código y el nombre de la provincia", vbCritical,  
"Error en Alta"  
End If  
End Sub
```

El código fuente asociado al botón Eliminar:

```
Private Sub cmdEliminar_Click()
If Data1.Recordset.EOF = False Then
    respuesta = MsgBox("¿Desea borrar esta provincia?", vbYesNo)
    If respuesta = vbYes Then
        Data1.Recordset.Delete
        Data1.Recordset.MoveFirst
    End If
End If
End Sub
```

El código fuente asociado al botón Salir:

```
Private Sub cmdSalir_Click()
End
End Sub
```

## 10. Práctica 6. Crear Formulario de Consulta

En un formulario realizado en Visual Basic podemos realizar consultas en SQL utilizando para ello alguno de los controles que se nos ofrecen como por ejemplo el componente DATA.

Por otro lado en Visual Basic no sólo se pueden pedir datos al usuario de la aplicación que estamos haciendo mediante los controles de un formulario sino que disponemos de un par de cajas de diálogos predefinidas: InputBox y MsgBox

- La función InputBox visualiza una caja de diálogo con un mensaje que indica al usuario el tipo de información que debe introducir. Un detalle importante a tener en cuenta es que InputBox devuelve un dato de tipo Variant (de VarType igual a 8 - String).

El resto de los parámetros existentes en InputBox son opcionales. Si queremos omitir un argumento que precede a otro especificado, hay que poner las correspondientes comas delimitadoras. La sintaxis de InputBox es:

```
InputBox (mensaje, [ título ] [ , por_omisión ] [ , posx ] [ , posy ]
```

- La sentencia MsgBox visualiza un mensaje en una caja de diálogo. Su sintaxis es:

```
MsgBox mensaje [ , botones ][ , título ]
```

El valor de la expresión botones describe el número y el tipo de botones a visualizar, el estilo del icono,...., etc por omisión este valor es cero.

Por otro lado en Visual Basic el Cuadro de Herramientas suele venir por defecto con una serie de componentes, pero nosotros podemos utilizar otros muchos en nuestros proyectos que no están incluidos entre los que aparecen por defecto. Para la práctica vamos a usar uno de ellos el componente Microsoft FlexGrid Control 6.0 (SP3).

Vamos a realizar una práctica que nos permita conjugar todos estos puntos nombrados en la introducción. Para ello vamos a crear un formulario en el que vamos a poder consultar las provincias dadas de alta en la aplicación, mostrándonos el resultado de dicha consulta en un objeto FlexGrid.

## 10.1. Creación del proyecto

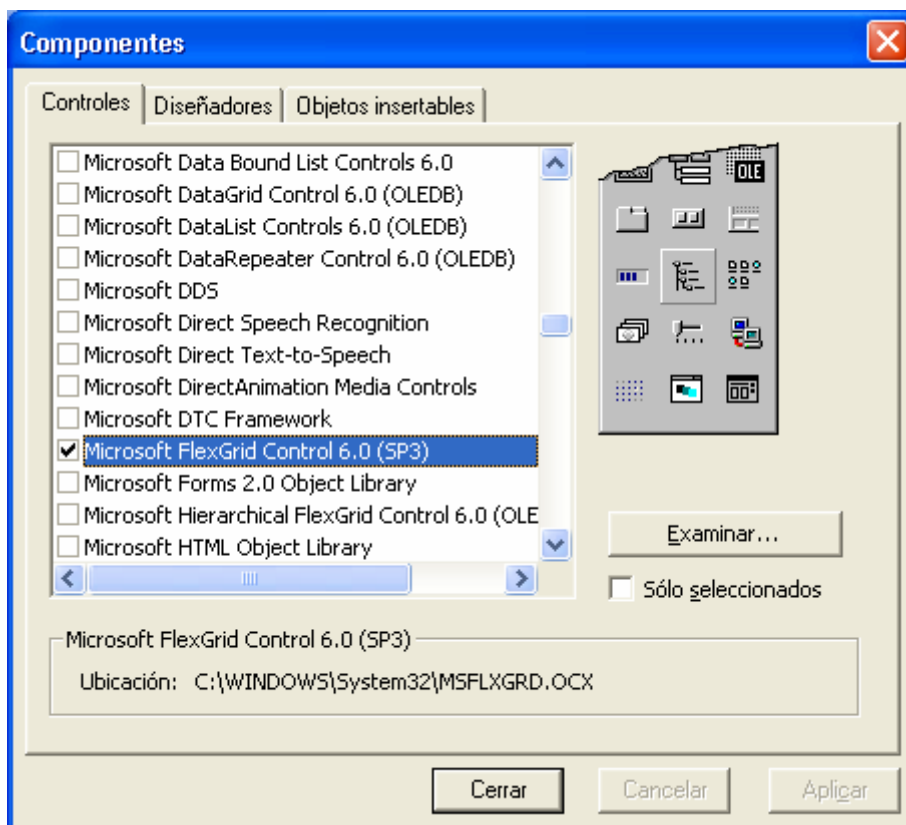
Creemos un nuevo proyecto en Visual Basic, para ello en el menú principal seleccionemos:

Archivo - Nuevo proyecto

## 10.2. Agregar componente

Como ya hemos dicho necesitamos un nuevo componente, el FlexGrid. Para poder usarlo hemos de agregarlo al Cuadro de Herramientas de nuestro proyecto. Para ello vaya al menú principal de VB y seleccione la opción Proyecto y dentro de esta Componentes.

Se nos presentará el cuadro de diálogo Componentes, con el siguiente aspecto:

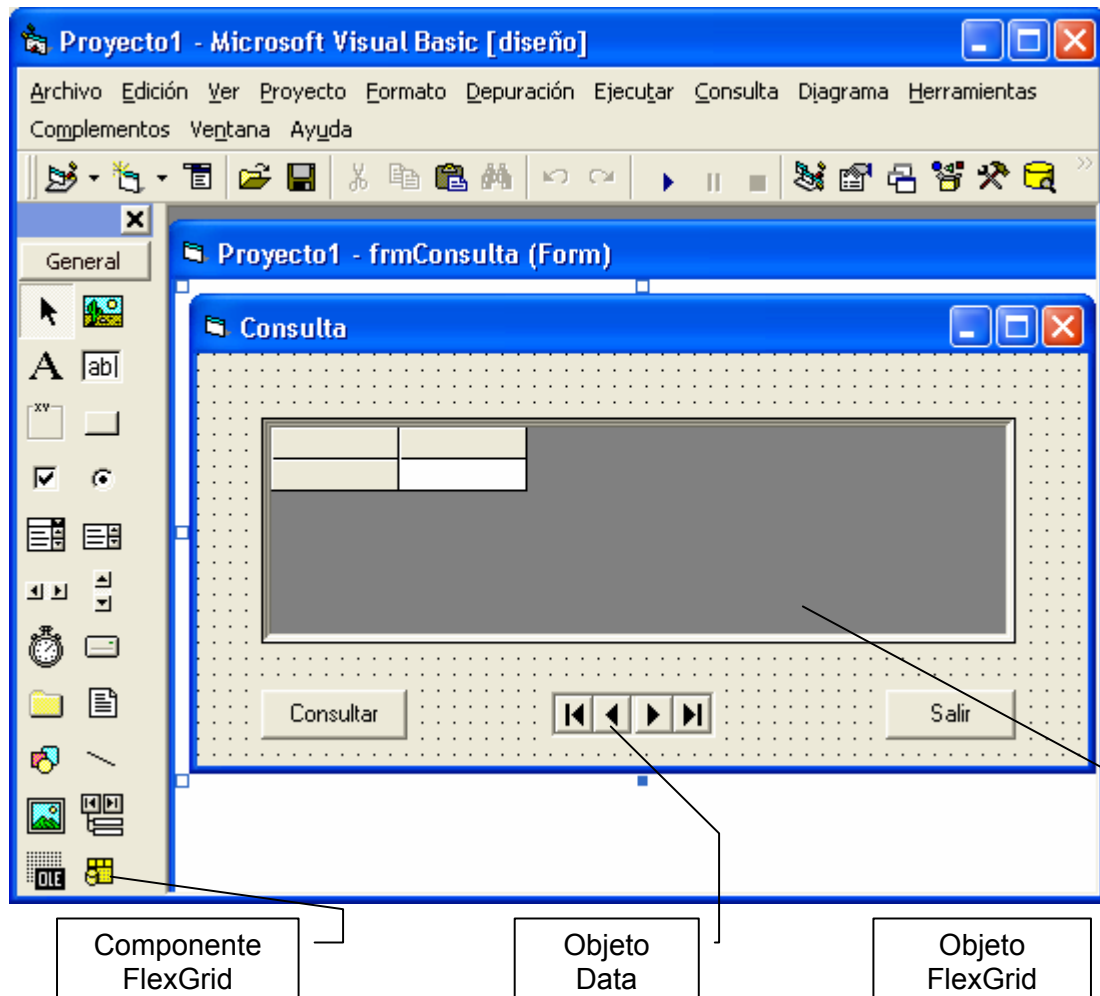


Busque y seleccione el Microsoft FlexGrid Control 6.0 (SP3), pulsando posteriormente sobre aplicar.

Observe como en el último lugar de su Cuadro de Herramientas de su proyecto le aparece un nuevo componente, el mencionado FlexGrid.

### 10.3. Objetos del formulario

Por defecto el proyecto se nos ha creado con un formulario. Modifiquemos dicho formulario insertando un par de botones, un objeto Data y un objeto FlexGrid hasta que presente el siguiente aspecto:



Los objetos tipo Data son intermediarios que conectan los controles que gestionan, reciben o envían, los datos y los diferentes elementos de la base de datos que se conecten a través de ella. Para la práctica que estamos desarrollando:

- El control Data deberá activar su propiedad DataBaseName con la ruta y el nombre de la base de datos a la que se quiera conectar. En nuestro caso seleccione la base datos: Gestion de Pedidos.
- La conexión a una de las tablas de la base de datos se consigue activando la propiedad RecordSource. Antes de hacerlo se debe haber indicado el valor a la propiedad DataBaseName. Para la práctica que estamos realizando, seleccione la tabla Provincias.

Pero el control Data sólo consigue establecer el enlace con la base de datos y la tabla que se ha elegido para el formulario en el que estamos trabajando. Para enlazar con

los campos de los registros de la tabla Provincias, deberemos activar una de las propiedades del objeto FlexGrid que hemos insertado en nuestro formulario, dicha propiedad es DataSource a la que le damos el valor Data1 que se nos ofrece.

Con esto habremos conectado el objeto FlexGrid de nuestro formulario con la tabla Provincias de la base de datos por lo que en principio el objeto FlexGrid nos mostrará todas las provincias existentes en nuestra tabla.

#### 10.4. Código del formulario

Vamos a programar ahora los botones que hemos creado, para ello en la pantalla de código fuente ponga lo siguiente:

En la sección destinada a las declaraciones:

```
Dim strNombre As String
```

Para el botón de Consultar el código será:

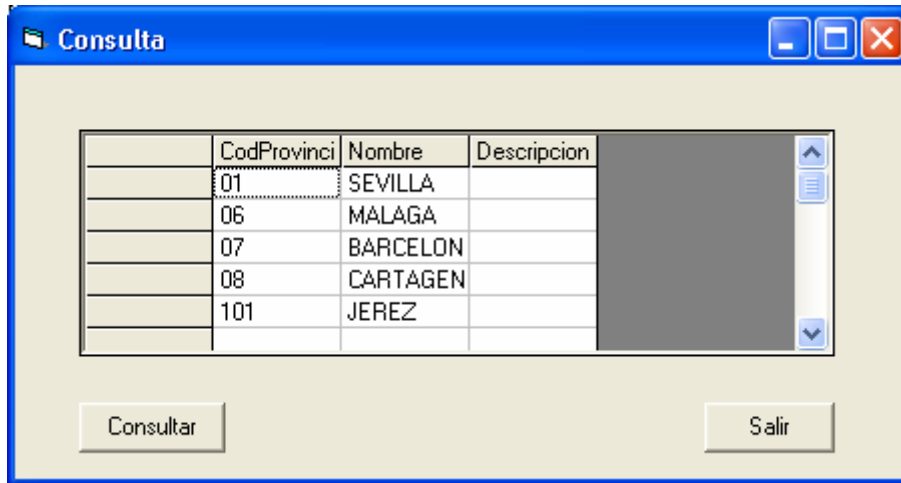
```
Private Sub cmdConsultar_Click()  
strNombre = ""  
strNombre = InputBox("Introduzca el nombre de la provincia", "Búsqueda  
Proyectos")  
If strNombre <> "" Then  
strNombre = Trim(UCCase(strNombre))  
Data1.RecordSource = "SELECT * FROM PROVINCIAS WHERE  
PROVINCIAS.NOMBRE like "*" & strNombre & "*"  
Data1.Refresh  
Else  
If MsgBox("Debe introducir el motivo de búsqueda. ¿Desea continuar sus  
búsquedas?", 20, "AVISO") = vbNo Then  
End  
End If  
End If  
End Sub
```

Para el botón Salir el código fuente será:

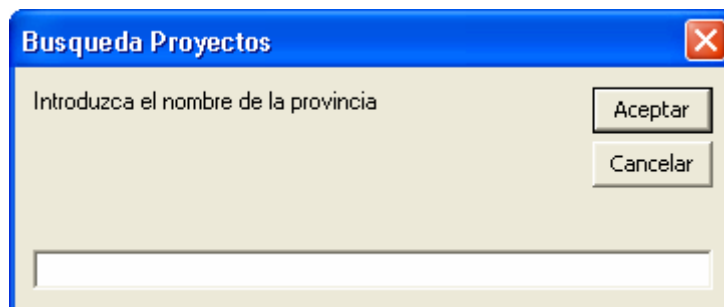
```
Private Sub cmdSalir_Click()  
End  
End Sub
```

## 10.5. Resultado

Habrá debido obtener un formulario con el siguiente aspecto:



Cuando pulse sobre el botón Consultar le aparecerá el siguiente cuadro de diálogo:



En el deberá introducir el nombre de la provincia que está buscando y pulsar aceptar para que la búsqueda se ejecute.

## 11. Práctica 7. Crear informe de datos

La edición empresarial de Visual Basic 6.0 permite acceder desde su entorno de desarrollo a una serie de utilidades visuales de acceso a datos para ayudarnos a conectarnos a cualquier base de datos compatible con ODBC, diseñar, ejecutar y guardar consultas y diseñar informes con los datos extraídos de la base de datos con la que hallamos conectado.

Las utilidades que nos van a valer para realizar un informe en VB son:

**a) Diseñador del entorno de datos:** proporciona un entorno interactivo durante el diseño para crear aplicaciones que durante la ejecución accedan a bases de datos. Con el diseñador del entorno de datos podemos llevar a cabo varias tareas, las más importantes son:

- Agregar un diseñador del entorno de datos a un proyecto realizado en Visual Basic.
- Crear objetos Connection.
- Crear objetos Command basados en procedimientos almacenados, tablas, vistas e instrucciones SQL.
- Crear jerarquías de objetos Command, o asociar entre sí a uno o varios objetos Command.
- Arrastrar y colocar campos de un objeto Command del diseñador de entorno de datos al diseñador de informes de datos.

**b) Diseñador de informes de datos:** es un generador de informes que puede utilizarse con un origen de datos como el diseñador de entorno de datos para crear informes con datos procedentes de tablas relacionales. Algunas de sus características son:

- Permite colocar campos arrastrados desde el diseñador del entorno de datos.
- Posee un conjunto de controles propios análogos a los controles de Visual Basic
- Permite mostrar una vista preliminar del informe e imprimirlo y exportar el informe a HTML y texto.

## 11.1. Objetivo

Como ejemplo vamos a crear un informe para mostrar los datos existentes en la tabla Artículos.

Recordemos que la tabla Articulos presentaba el siguiente diseño:

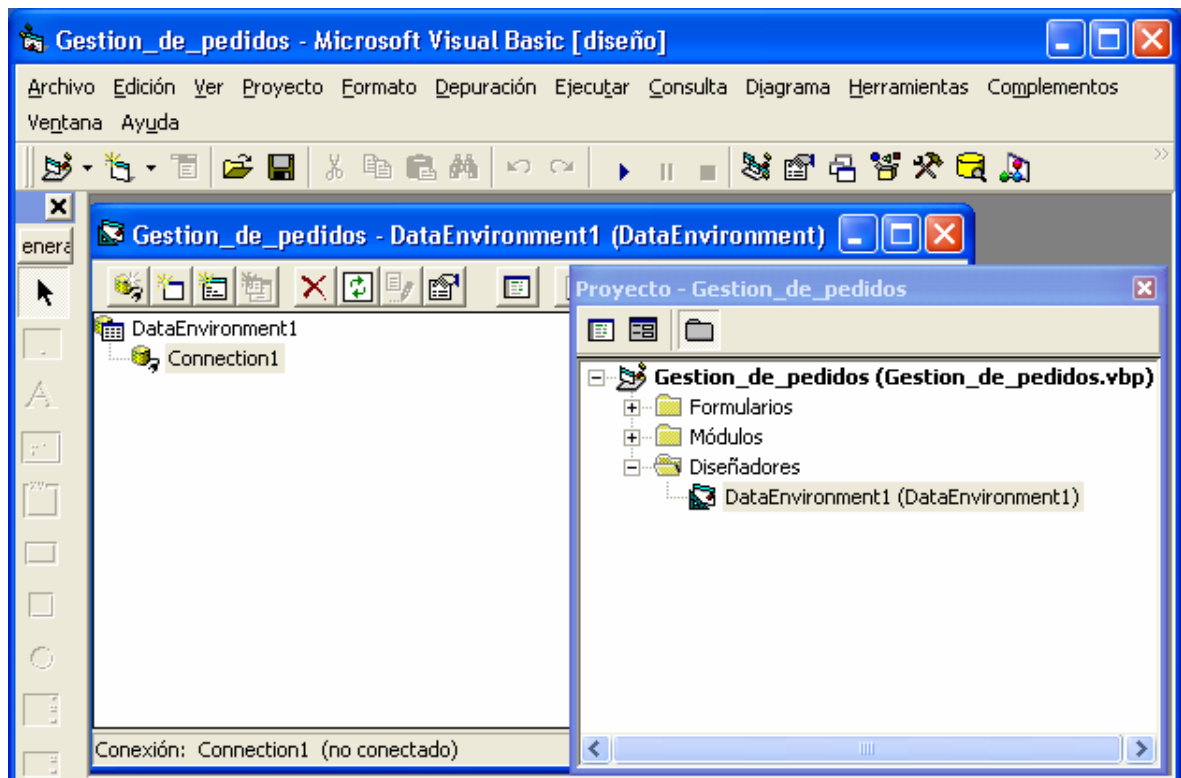
Nombre del Campo	Tipos de datos	Descripción
CodArticulo	Texto	Código del artículo
Descripción	Texto	Descripción del artículo
Comentario	Demo	Comentario sobre el artículo

Vamos a crear el informe en el proyecto Gestion\_de\_pedidos que ya hemos realizado.

Tras crearlo modificaremos el menú existente en dicho proyecto creando una nueva opción Informes y dentro de esta la de Informe de artículos, desde la que llamaremos al informe que hayamos creado.

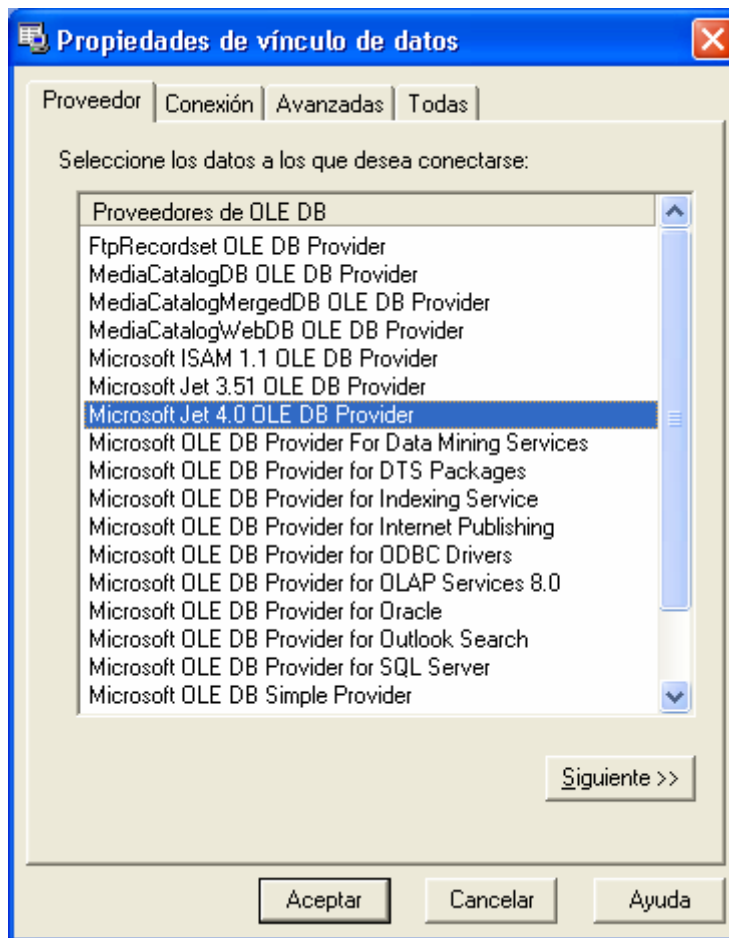
## 11.2. Agregar Data Enviroment y establecer conexión con base de datos

Desde el menú principal de VB pulse Proyecto - Agregar Data Enviroment



Junto con el Data Enviroment se le ha creado un objeto Connection. Un objeto Connection representa una conexión a una base de datos

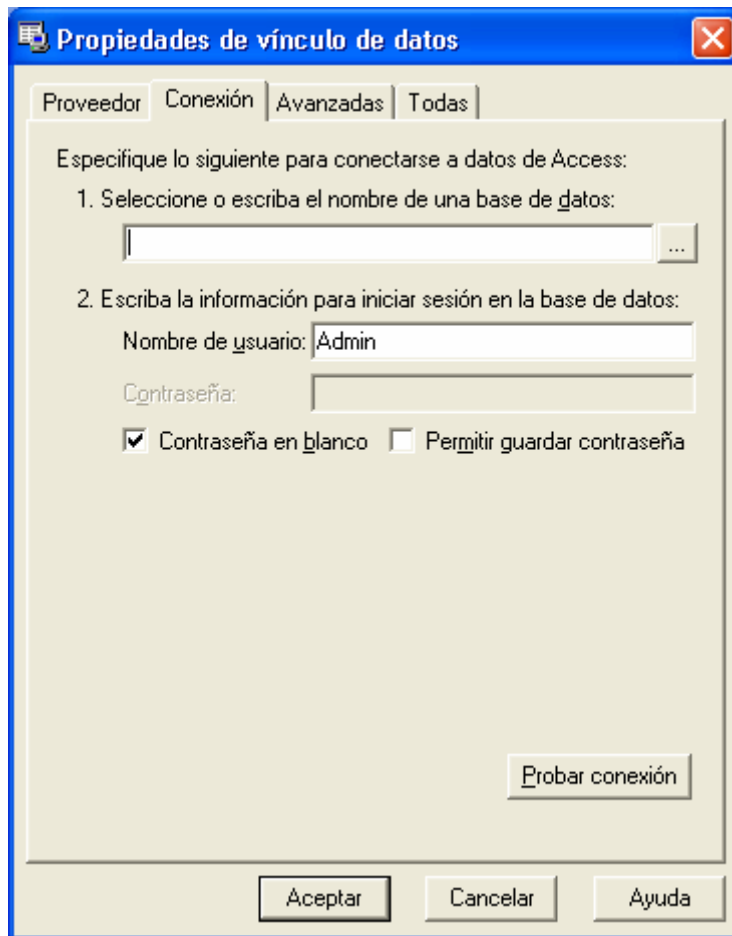
Pulse sobre el objeto Connection con el botón derecho del ratón y elija la opción Propiedades. Se le muestra la siguiente ventana:



En esta pantalla puede elegir el proveedor de datos que desea utilizar para conectarse a la base de datos.

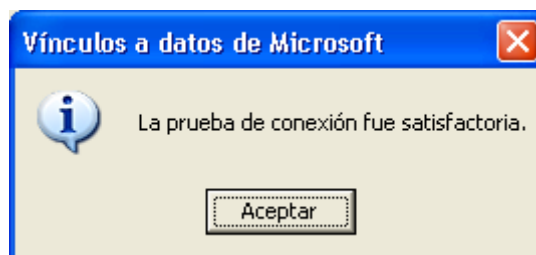
En nuestro caso seleccione la opción Microsoft Jet 4.0 OLE DB Provider y pulse Siguiente.

La siguiente opción presenta el siguiente aspecto:



En nuestro caso no vamos a utilizar contraseña para acceder a la base de datos y el usuario que vamos a usar en la base de datos es el que viene en Visual Basic por defecto: Admin (no lo modifique).

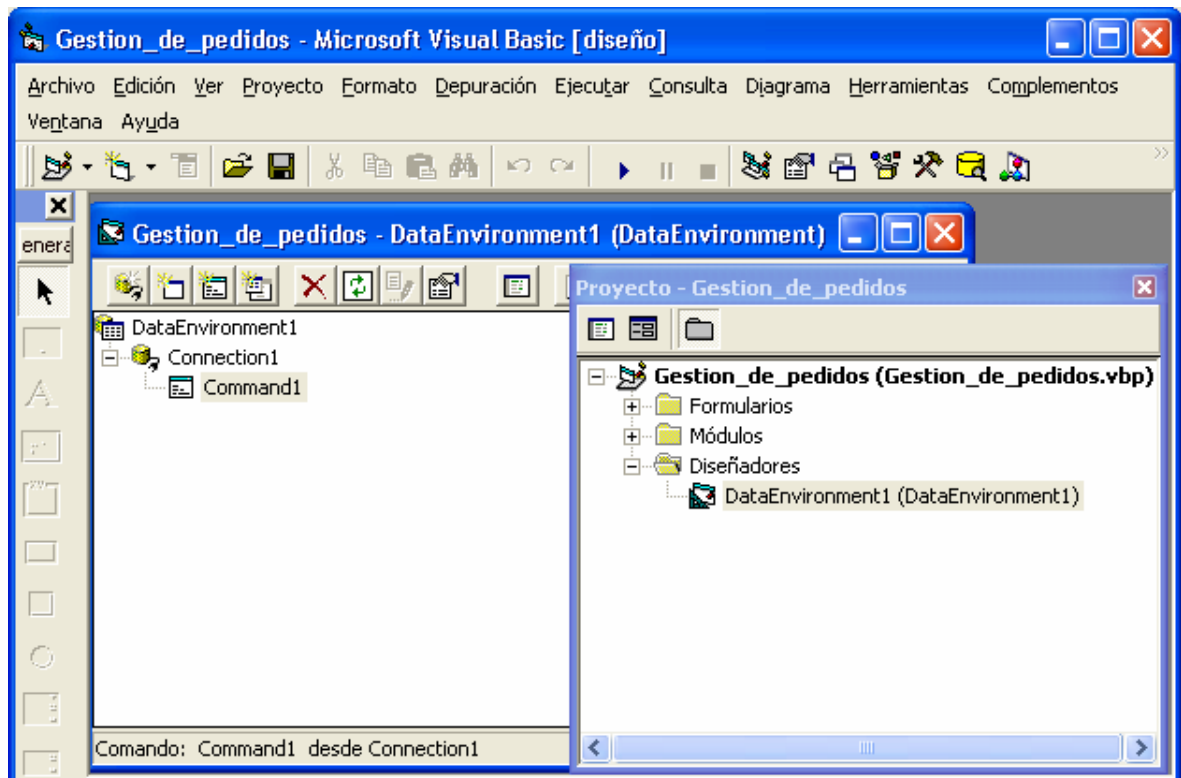
Seleccione la base de datos Gestion\_de\_pedidos tras buscarla y pulse el botón Probar conexión, al hacerlo le debe salir el mensaje:



A continuación pulse el botón Aceptar.

Tras estos pasos habrá creado la conexión con la base de datos desde Visual Basic. Sin embargo el objeto Connection por si mismo no proporciona un vínculo con una tabla específica tal y como necesitamos.

Vamos a indicarle ahora a VB la tabla de la base de datos que necesitamos para nuestro informe. Para ello pulse nuevamente con el botón derecho sobre el objeto Connection1 y elija la opción Agregar comando:

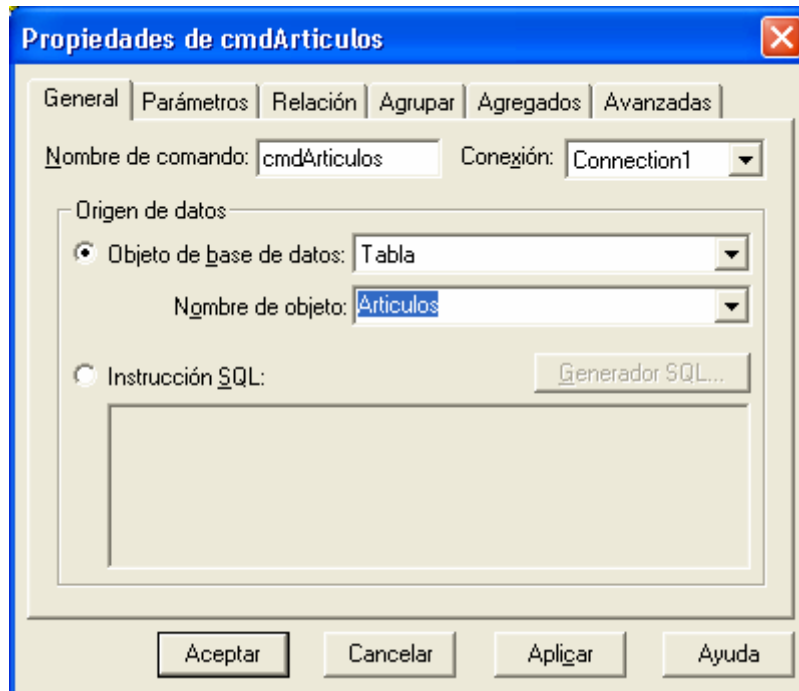


Habrá creado un objeto Command, los objetos Command definen información de detalle específica acerca de cómo deben recuperarse los datos a través de una conexión.

Cada objeto Command representa un objeto de la base de datos (una tabla, una consulta SQL,... etc.).

Pulse con el botón derecho sobre el objeto Command que acaba de crear y elija la opción Cambiar nombre, póngale cmdArticulos.

Vuelva a pulsar con el botón derecho sobre el objeto Command ahora denominado cmdArticulos, y elija la opción Propiedades. Le debe aparecer el siguiente cuadro de diálogo:



En este cuadro de diálogo le aparecen dos posibilidades:

- Indicar a VB que los datos van a ser sacados de un objeto de la base de datos
- Informar a VB que va a poder acceder a los datos necesarios para el informe mediante una instrucción SQL, en cuyo caso podremos utilizar el Generador SQL si lo consideramos necesario para crear nuestra consulta SQL.

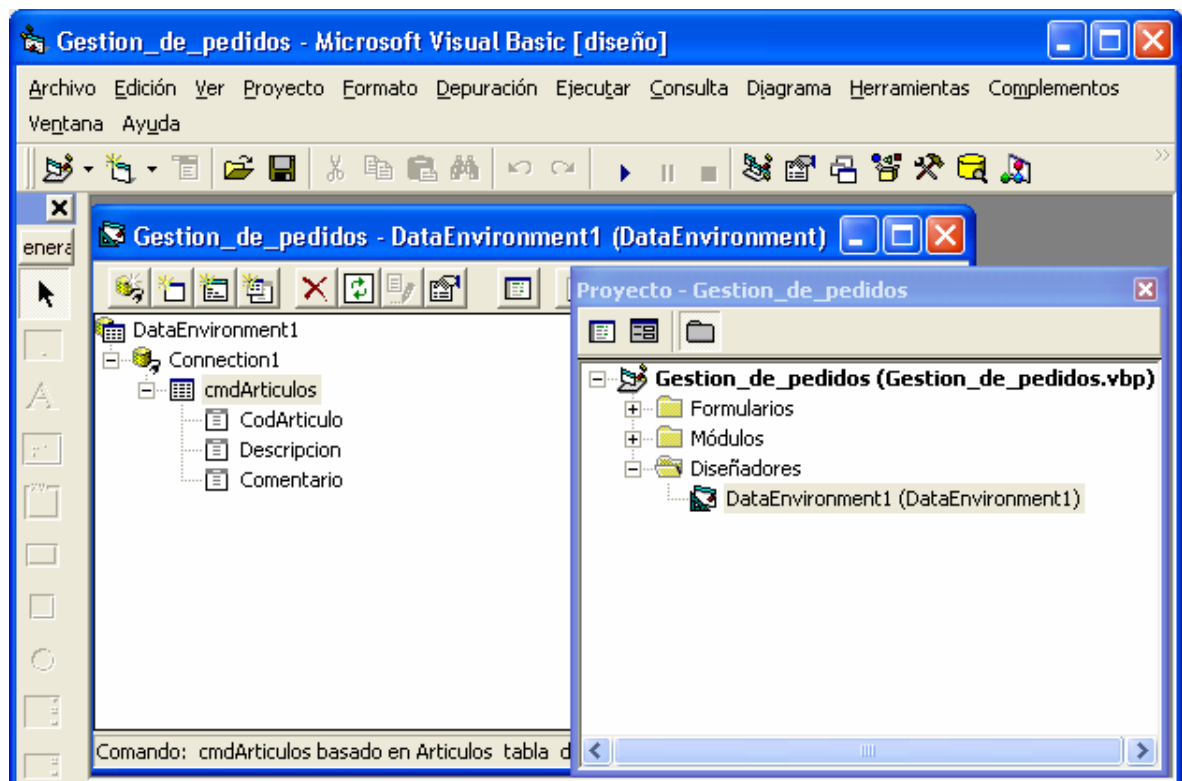
Para la práctica que estamos llevando a cabo elija Objeto de base de datos y dentro de estos el tipo Tabla, y seleccione de entre estas la tabla Articulos.

A continuación pulse sobre la pestaña Avanzadas y elija el tipo de bloqueo: 3 - Optimistic.

Pulse Aceptar.

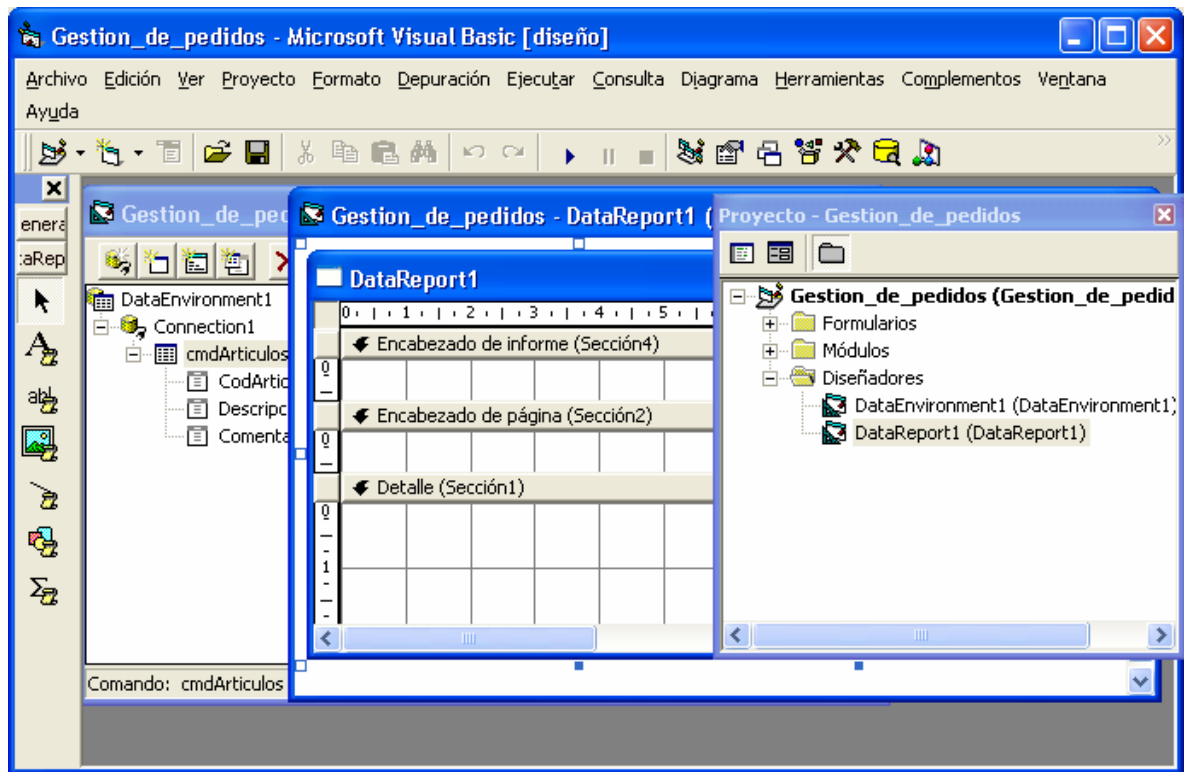
Observe como en el objeto Command, cmdArticulos, aparecen ahora los campos de la tabla Articulos. En nuestro caso estos campos eran:

- CodArticulo
- Descripcion
- Comentario



### 11.3. Creación del informe

Vaya al menú principal de VB y elija la opción Proyecto - Agregar Data Report, le aparece una nueva pantalla llamada DataReport1 en la que podrá diseñar su informe:



Esta pantalla DataReport1 es un objeto, como si fuese un nuevo formulario, y como tal tendrá sus propiedades, modifique la propiedad Caption poniéndole como valor: Informe de artículos.

A continuación asigne a su propiedad DataSource el valor DataEnviroment1 y a su propiedad DataMember el valor cmdArticulos.

Pulse con el botón derecho sobre la pantalla aún en blanco del informe DataReport1 y seleccione Obtener estructura del menú contextual. Responda Sí a la pregunta que se le formula sobre si desea reemplazar el diseño existente.

Observe que ahora los detalles corresponden a los Artículos.

Asigne a las propiedades GridX y GridY del objeto DataReport1 el valor 4.

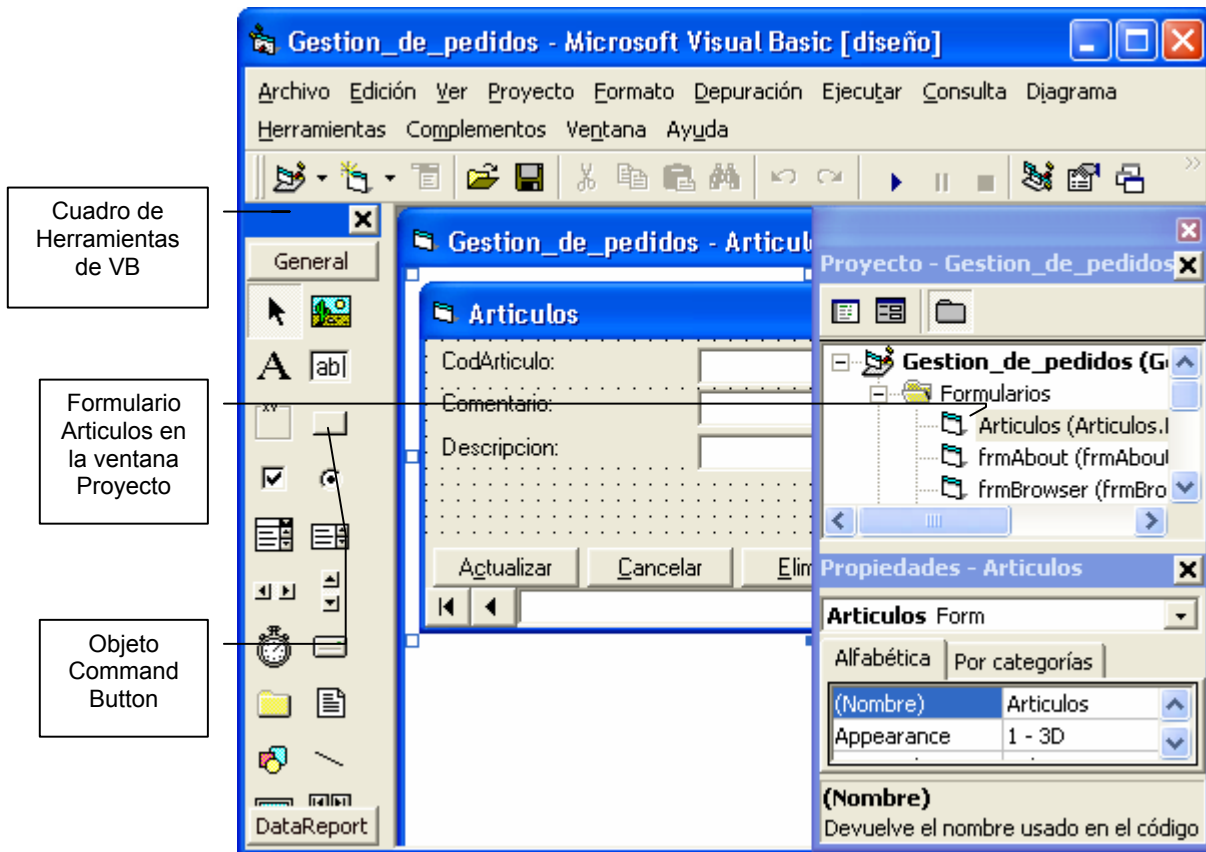
Por último arrastre los campos existentes en el objeto Command, cmdArticulos, sobre la sección cmdArticulos\_detail (detalle). Ajuste los tamaños de los campos según sus necesidades.

Por último pulse el botón Guardar de la barra principal de herramientas y guarde el DataEnviroment y el DataReport.

## 11.4. Visualizar el informe

Por último y para acabar la práctica vamos a crear un botón en nuestra aplicación que permita visualizar el informe que acabamos de crear al pulsarlo.

Abra el formulario Artículos pulsando dos veces sobre el en la ventana Proyecto. Nos aparece el formulario en el modo Objeto:



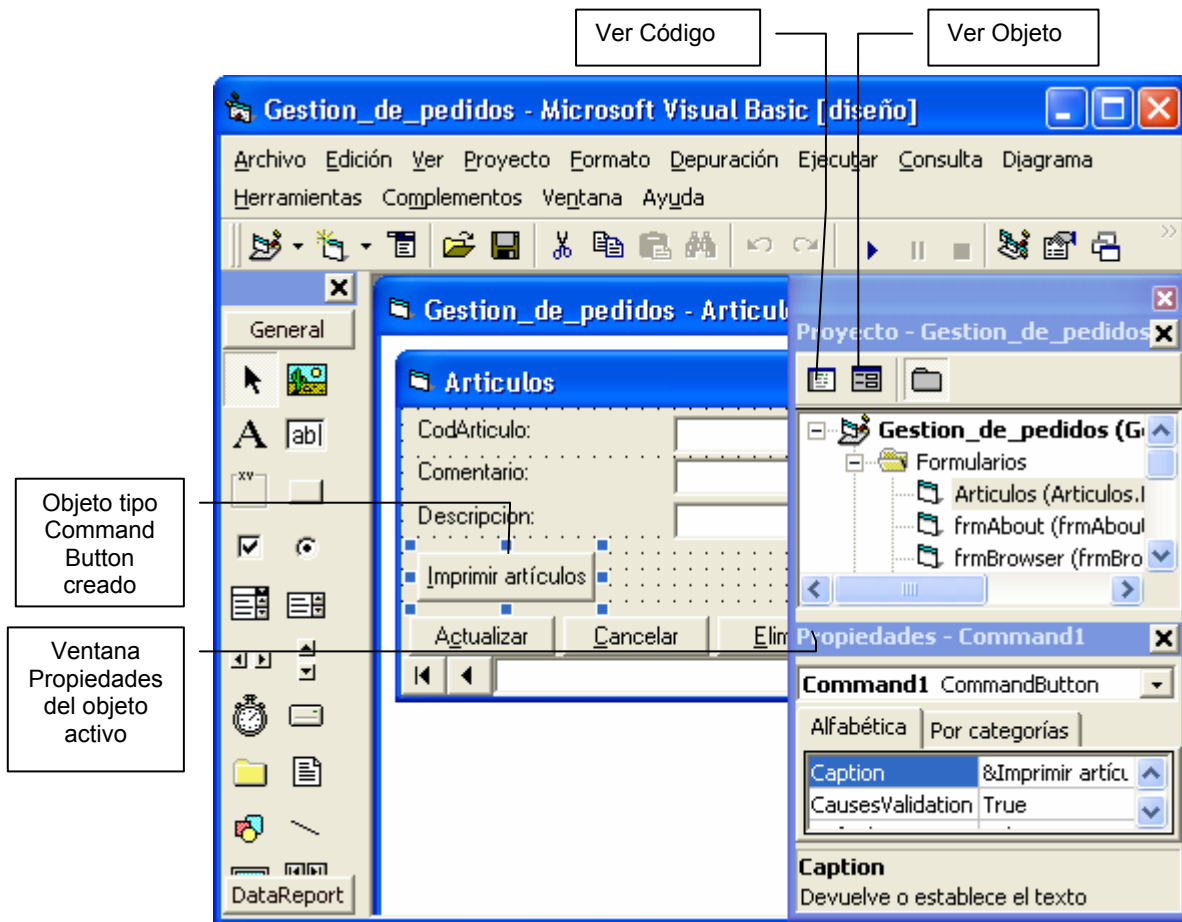
A la derecha de la ventana del entorno de desarrollo tenemos el Cuadro de herramientas de VB.

Dentro de este cuadro tenemos el objeto Command Button, pulse sobre el, observe como al volver sobre el formulario Articulos el cursor ha cambiado su forma, pasando de la flecha a un símbolo igual a: +. Esto indica que si pulsamos sobre el formulario en la ventana de diseño podremos crear un objeto igual al pulsado, es decir un objeto Command Button.

Pulse sobre la ventana de diseño en el formulario y arrastre el cursor sin dejar de pulsar, creando de esa manera un botón.

Modifique en la ventana de Propiedades las propiedad Caption, poniéndole el valor "&Imprimir artículos".

La pantalla debe mostrar el siguiente aspecto:

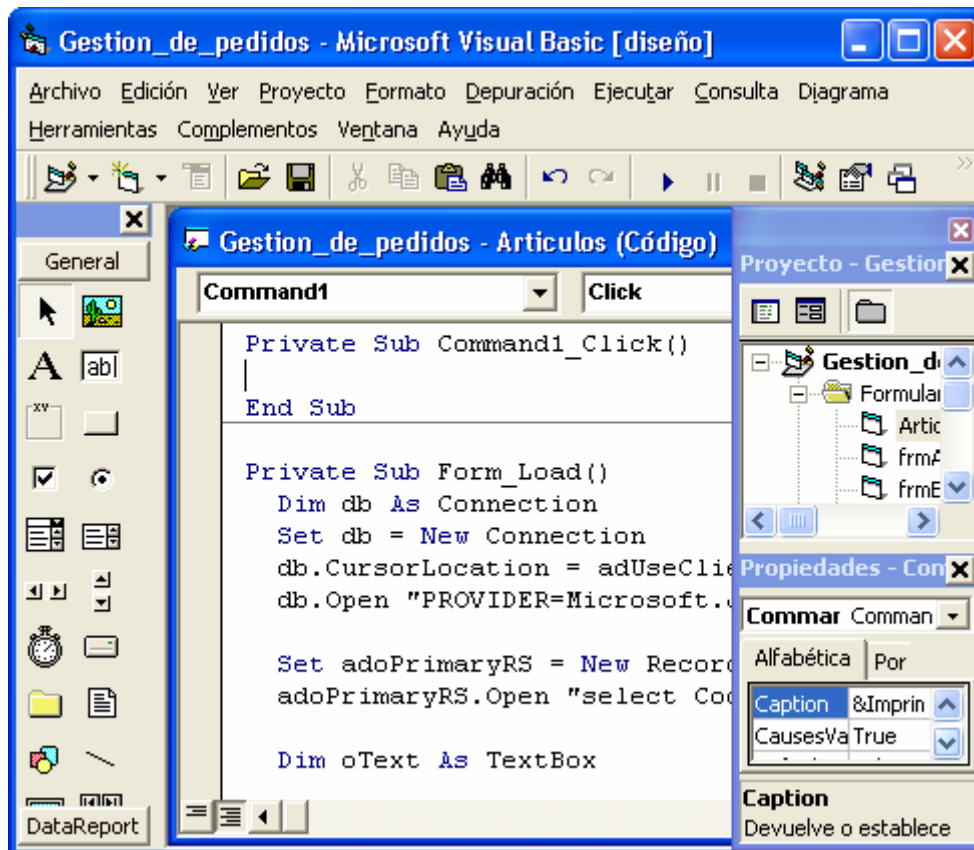


Programemos ahora el botón que hemos creado de forma que cuando ejecutemos la aplicación y pulsemos sobre el, nos muestre el informe que acabamos de realizar. Para ello haga doble clic sobre el botón para pasar al modo Código de VB.

**Nota:** Para pasar del modo Diseño al modo Código en VB y viceversa disponemos de dos botones situados en la parte superior de la ventana Proyecto.

En la ventana de Código es donde se realiza la escritura del código fuente de los módulos que responden a los eventos de cualquiera de los controles que componen la interfaz que hayamos creado.

La ventana de código que se nos presenta es:



El cursor se nos presenta en el punto al que irá Visual Basic cuando esté ejecutando nuestra aplicación y por lo tanto será aquí donde tengamos que poner la línea de código que vamos a programar para que aparezca el informe creado.

Escriba lo siguiente:

```
Private Sub Command1_Click()  
DataReport1.Show  
End Sub
```

Guarde los cambios efectuados en el formulario.

**Nota:** La ventana código presenta en su parte superior dos listas desplegables. La de la izquierda permite desplegar toda la lista de los objetos que componen el formulario activo. La de la derecha visualiza los sucesos posibles del objeto elegido en la ventana de la izquierda para los que se puede escribir código.

## El autor

Licenciado en Matemáticas. Universidad de Sevilla. (2001)

### Experiencia docente:

Profesor en la Universidad de Sevilla en el Master de Tecnologías de Análisis para la Sociedad de la información.

Profesor de Matemáticas y Ciencias de la Naturaleza de Educación Secundaria Obligatoria en el IES Mariana de Pineda (Dos Hermanas/Sevilla).

Amplia experiencia impartiendo clases en academias y a particulares de matemáticas a distintos niveles educativos, principalmente a niveles de secundaria obligatoria, bachillerato y universidad.

Elaboración de programaciones y unidades didácticas. Conocimiento de la estructura, objetivos y contenidos del sistema educativo.

Impartición de cursos sobre aprendizaje de distintas aplicaciones informáticas a sus usuarios finales en la Junta de Andalucía.

### Otra Experiencia Profesional:

Septiembre 2001 / Junio 2005: Tareas de análisis y programación de aplicaciones en entorno Oracle 9i para la Junta de Andalucía.

Junio 2005 / Septiembre 2007: Tareas de análisis y programación de aplicaciones en entorno Oracle 9i y Cobol para el Servicio Andaluz de Salud.

Marzo 2001 / Febrero 2004: Tareas de consultaría para la aplicación S.R.P. de la Consejería de Justicia de la Junta de Andalucía.

Formación:

Octubre 2004 - Marzo 2005: Curso de Adaptación Pedagógica (C.A.P.) en el Instituto de Ciencias de la Educación de la Universidad Complutense de Madrid.

Máster en Tecnologías de Análisis para la Sociedad de la Información – Universidad de Sevilla (Edición 2003- 2004)

Título de Experto Universitario en Tecnologías de Análisis para la Sociedad de la información (Edición 2001 - 2002).

Curso Superior de Capacitación en las Nuevas Tecnologías de la Información y la Comunicación. (Periodo: 01/10/2001-23/01/2002, Duración: 400 horas) Organizado por la Consejería de Empleo y Desarrollo Tecnológico de la Junta de Andalucía).

**Correo electrónico:** [f\\_flores\\_gil@hotmail.com](mailto:f_flores_gil@hotmail.com)

