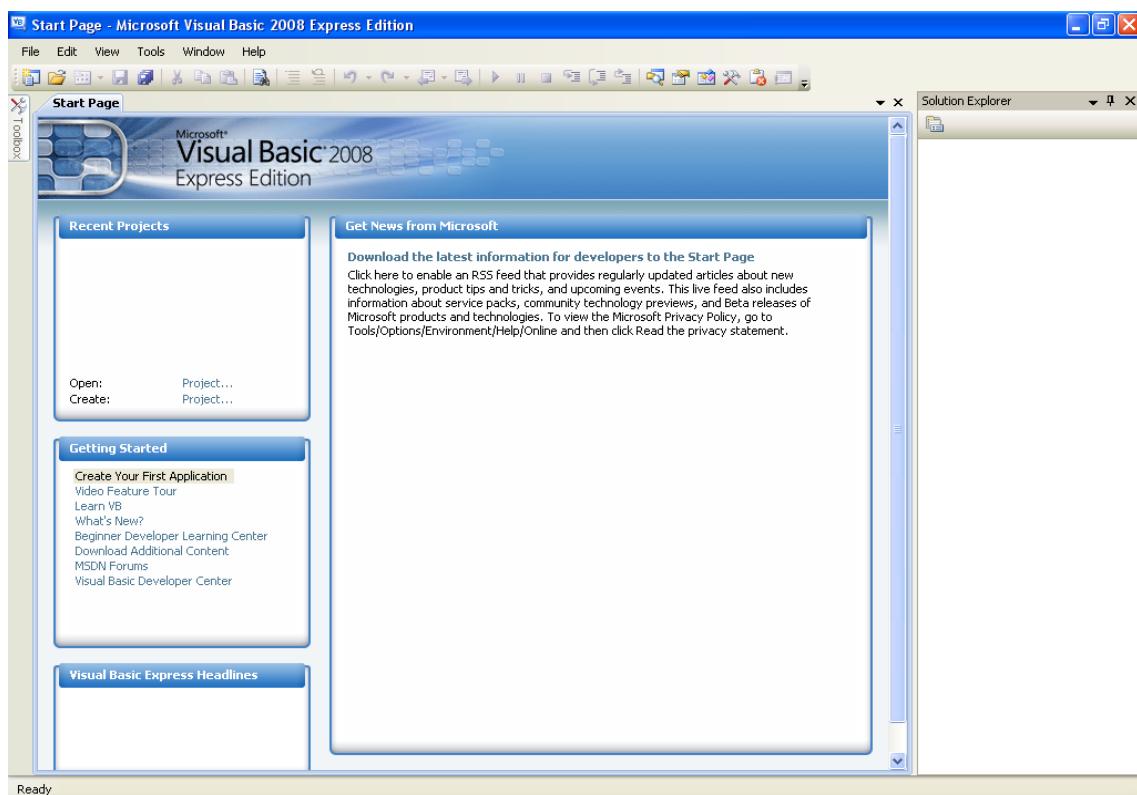


INGRESAR AL VISUAL BASIC.NET

Al ingresar por primera vez al visual basic.net observara la siguiente pantalla:



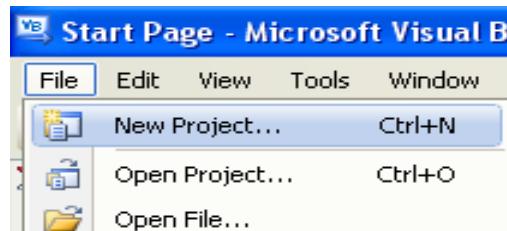
En esta pantalla se muestra principalmente la página de inicio con proyectos recientes y las opciones para abrir o crear un nuevo proyecto. También se muestra en cuadro de herramientas y el explorador de soluciones, pero, vacios porque no hay ningún proyecto activo.

Para visualizas todos los componentes de Visual Basic.Net debe crear una aplicación.

CREAR UNA APLICACIÓN

Para crear una aplicación puede seguir cualquiera de los siguientes pasos:

Elegir la opción Archivo/Nuevo Proyecto.



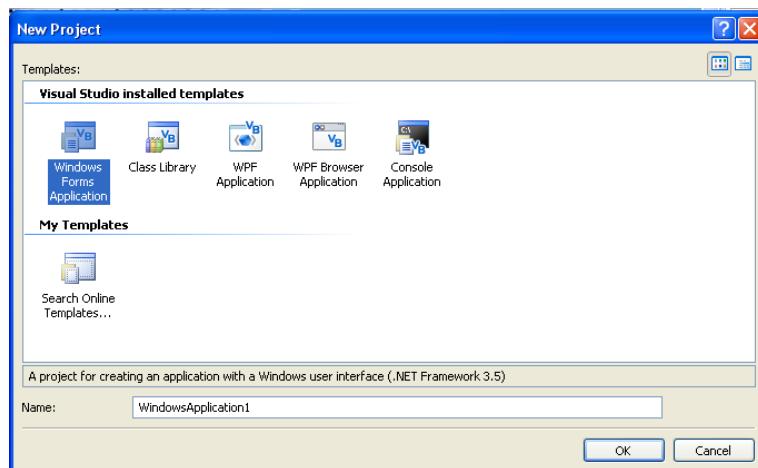
Hacer clic en el botón Nuevo Proyecto que se encuentra en la barra Estándar.



Hacer clic en el proyecto de la opción Crear que se encuentra en la página de inicio.



En cualquiera de los pasos anteriores se visualiza la siguiente ventana:



Esta ventana contiene las plantillas instaladas de Visual Studio.Net en forma predeterminada se elige Aplicación para Windows.

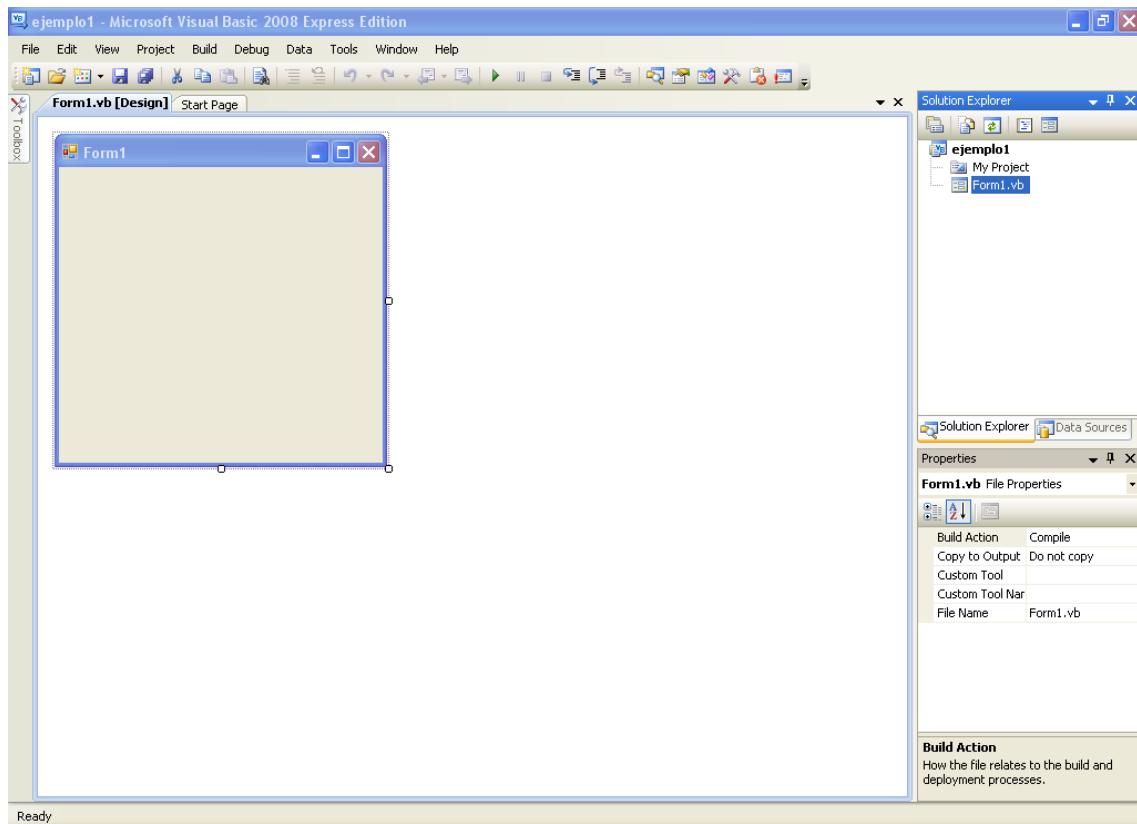


En la caja Nombre, escriba el nombre de su nueva aplicación y presione Enter o haga clic en el botón Aceptar.



ELEMENTOS DEL VISUAL BASIC.NET

Al crear una nueva aplicación se visualiza la siguiente ventana que consiste en el IDE del Visual Basic.Net en el ejemplo se ha creado una aplicación llamada ejemplo1.



Como puede observar, el Visual Basic 2008 express edition contiene todos los elementos de versiones anteriores como se muestra a continuación

BARRA DE TITULO

Contiene el nombre del proyecto y su estado. Cuando estamos en estado de diseño, es decir, dibujando los controles o escribiendo las instrucciones, la barra solo muestra el nombre del proyecto.



Cuando se ejecuta la aplicación se muestra la palabra ejecutando en la barra de título.



Durante este estado, la aplicación solo ejecutara las instrucciones indicadas y no se podrá modificarlas ni alterar el diseño de la aplicación.

Durante la ejecución de una aplicación, esta se puede interrumpir para modificar, agregar una nueva instrucción, en este estado se visualiza la palabra depurando.



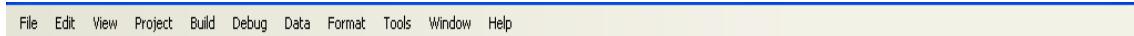
Los siguientes botones se encuentran en la barra estándar permiten iniciar, interrumpir, detener o depurar una aplicación.



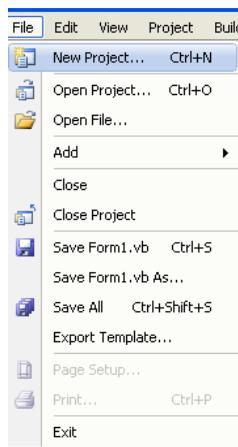
Como en versiones anteriores de Visual Basic, se puede ejecutar una aplicación pulsando la tecla F5.

LA BARRA DE MENU

La barra de menú contiene todas las opciones que permiten utilizar el Visual Basic 2008 Express.



Cada una de estas opciones contiene sub opciones, las cuales se pueden también elegir o activar pulsando las teclas que las acompañan, ejemplo Ctrl + O para abrir un proyecto o haciendo clic en su gráfico representativo que se encuentra en la barra estándar.



La barra estándar contiene los botones que permiten acceder de manera inmediata a las opciones más comunes del Visual Basic. Al pasar el puntero del mouse por cada uno de los botones se visualiza un mensaje indicando la función de cada uno de ellos.



Crea un nuevo proyecto.

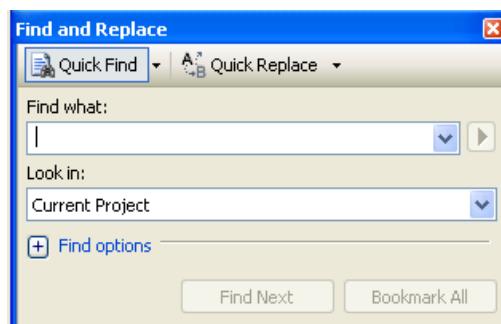


Abre un proyecto existente.



Permite buscar y/o reemplazar un texto.

Al hacer clic en este botón se visualiza la siguiente ventana:



En la caja buscar, escriba el texto que desea buscar y en buscar en, indique donde se debe buscar el texto. Si también desea reemplazar un texto haga clic en reemplazo rápido.



Este botón permite marcar como comentario la línea de instrucción actual o el bloque de líneas seleccionadas.

```

Form1.vb* Start Page Form1.vb [Design]*

    Button1
    Public Class Form1
        Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Ha
            'MsgBox("holo")
        End Sub
    End Class
  
```



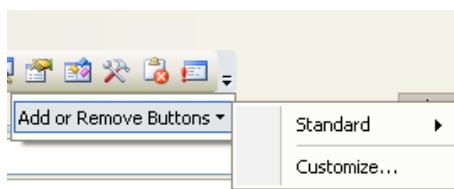
Este botón permite quitar la marca de comentario a la línea de instrucción actual o el bloque de líneas seleccionadas.

```

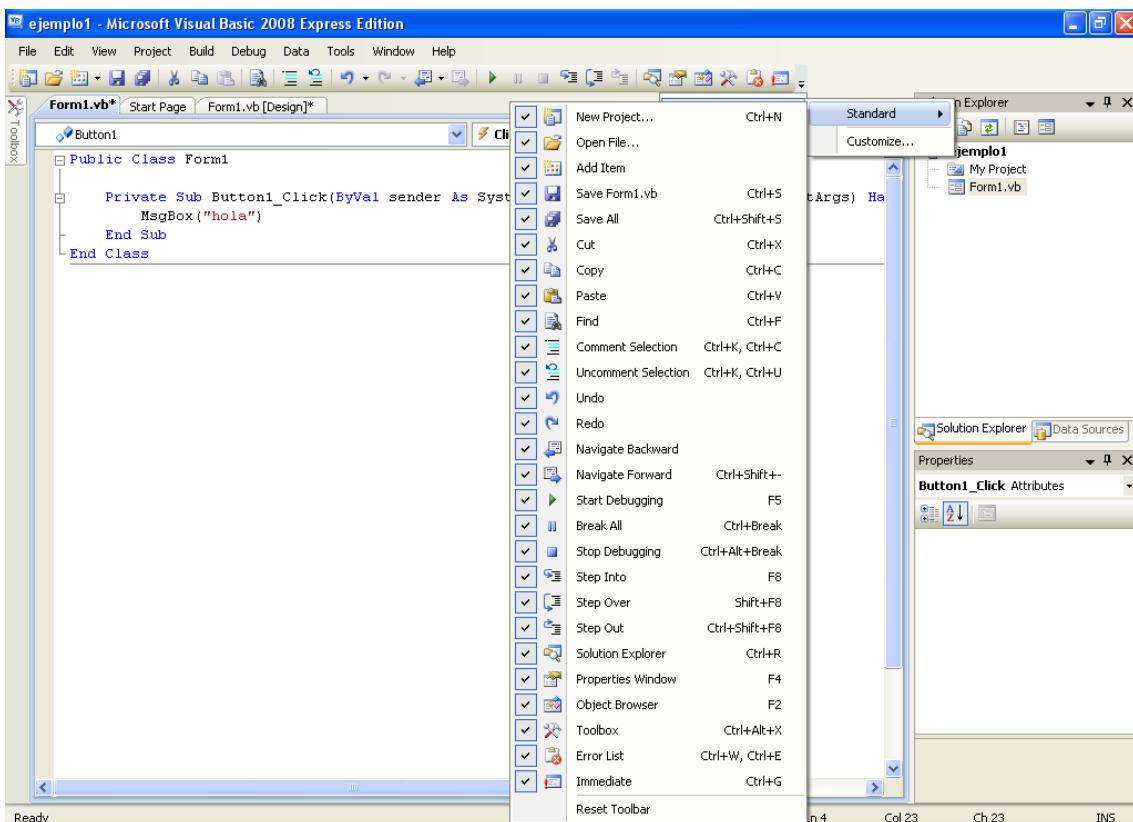
Form1.vb* Start Page Form1.vb [Design]*

    Button1 Click
    Public Class Form1
        Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Ha
            MsgBox("holo")
        End Sub
    End Class
  
```

El último botón de la barra estándar permite agregar o quitar botones:



Al hacer clic en estándar se visualiza la ventana con los botones. Usted puede activar o desactivar su casilla para visualizarlos o no.

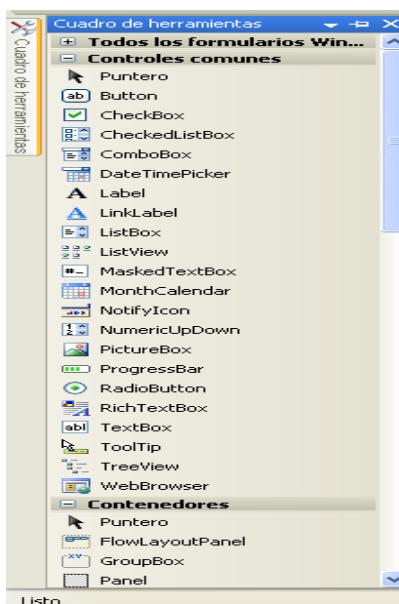


EL CUADRO DE HERRAMIENTAS

Como su nombre lo indica, este cuadro contiene todas las herramientas que se necesitan para desarrollar las aplicaciones.

Ala pasar el puntero del mouse por este cuadro se visualizan todas las Herramientas.

Algunas Herramientas del Visual Basic 2005 Express son similares a las de versiones anteriores de Visual Basic.



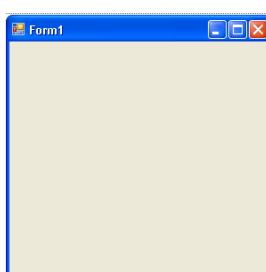
Si desea que el cuadro de herramientas este permanente en su pantalla debe de hacer clic en el botón Ocultar Automáticamente.



El Formulario

El Formulario se utiliza para crear la interfaz del usuario, es decir, la ventana donde se realizará comunicación entre el usuario y la aplicación.

Aquí se dibujan las herramientas o controles del cuadro de herramientas, se le asignan propiedades y se escriben las instrucciones necesarias.

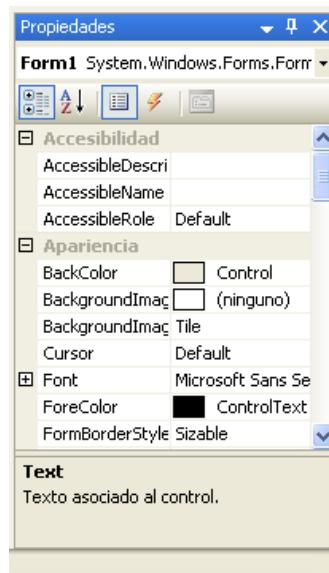


Para dibujar un control que se encuentra en el cuadro de herramientas, se puede hacer doble clic sobre el control o arrastrarlo desde el cuadro de herramientas hacia el formulario.



La Ventana de Propiedades

Esta ventana, como su nombre lo indica, contiene todas las propiedades o características que se les pueden asignar a los controles que se dibujan el formulario.



Como en casi todas las ventanas, en la barra de título se visualizan los botones que permiten:

- Indicar posición de la ventana en la pantalla.
- Indicar si la ventana se oculta de forma automática.
- Cerrar la ventana.



Después de la barra de título, la venta de propiedades muestra el nombre y tipo de control al cual se le está asignando las propiedades.

En el siguiente ejemplo indica que se está asignando propiedades al formulario llamado Form1.



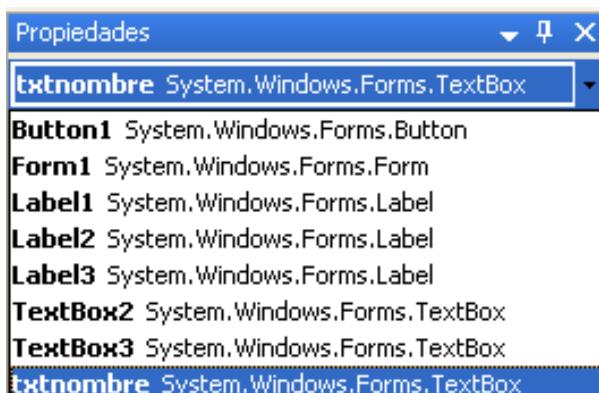
En el siguiente ejemplo indica que se está asignando propiedades al control llamado Button1 y es un botón de comandos.



En el siguiente ejemplo indica que se está asignando propiedades al control llamado TxtNombre y es una caja de textos.



Esta sección de la ventana de propiedades contiene una flecha donde usted puede seleccionar otro control que ah dibujado en su formulario para asignarle propiedades.

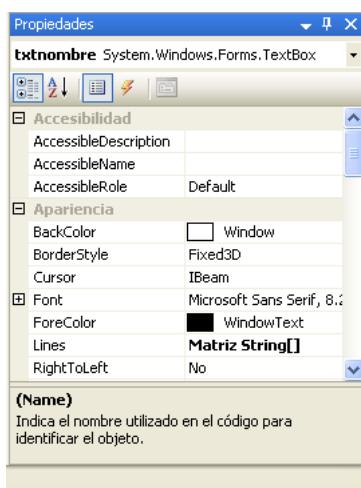


En la parte inferior del nombre del control se muestran los botones que permiten lo siguiente:

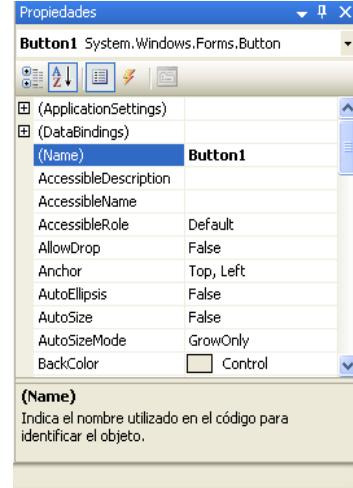
- Mostrar las propiedades o eventos ordenadas por su categoría.
- Mostrar las propiedades o eventos ordenadas alfabéticamente por su nombre.
- Ver solo las propiedades del objeto seleccionado.
- Ver solo los eventos del objeto seleccionado.



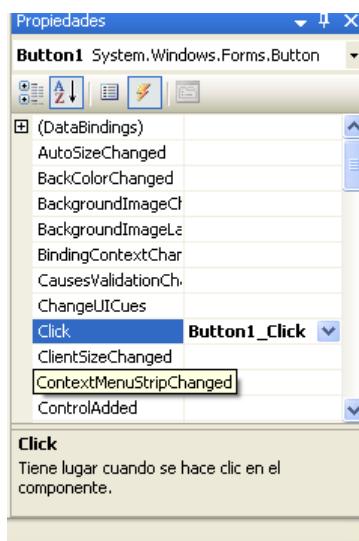
Ordenación por Categoría



Ordenación por Nombre



En la siguiente ventana se muestran los eventos del control Button1 ordenados alfabéticamente.

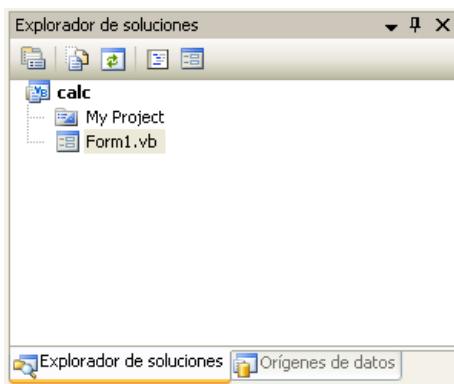


Cuando se tiene la lista de los eventos de un control, se puede hacer doble clic en su nombre para abrir la ventana de código.

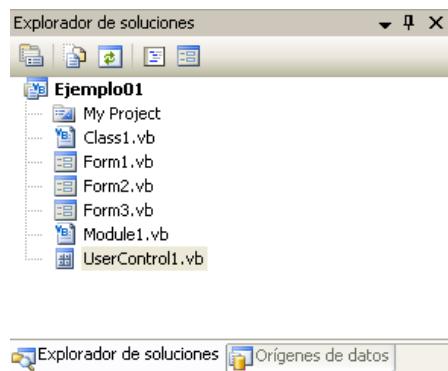
```

    ' Button1
    Public Class Form1
        Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
            ' Evento Click del botón
        End Sub
    End Class
  
```

Para regresar al formulario, puede pulsar las teclas Shift + F7 o hacer clic en la hoja: Form1.vb [diseño]* que se encuentra en la parte superior.



El siguiente ejemplo muestra el explorador de soluciones con una clase, tres formularios, un modulo con control de usuario.



El explorador de soluciones muestra los siguientes botones.

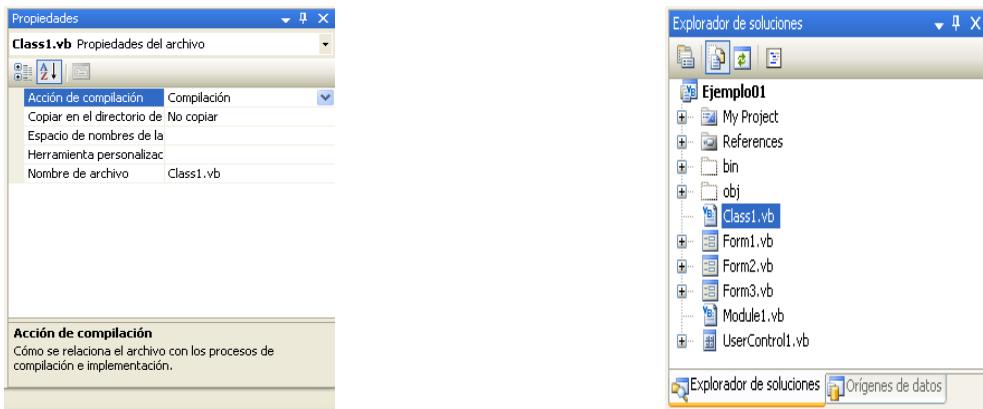


Estos botones permiten:

- Mostrar las propiedades del archivo seleccionado.
- Mostrar todos los archivos del proyecto.
- Actualizar el explorador de soluciones.
- Ingresar a la ventana de código el objeto seleccionado.
- Ver la ventana de diseño.

Propiedad del Archivo

Propiedad de Proyecto



Ventana De Código

```

Form1.vb [Form1.vb [Diseño] Página de inicio]
    btnnuevo Click
    Public Class Form1
        Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnnuevo.Click
            txtnombre.Text = ""
            txtedad.Text = ""
            txtdireccion.Text = ""
            txtnombre.Focus()
        End Sub
    End Class

```

Ventana De Diseño



Como en versiones anteriores de Visual Basic, también se puede ingresar a la ventana de código de un objeto pulsando la tecla F7 y para regresar a la ventana de diseño Shift + F7.

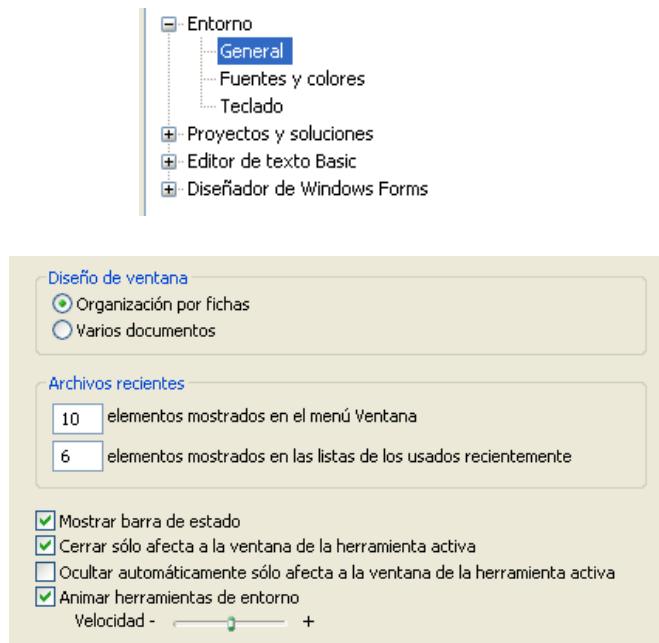
CONFIGURAR EL VISUAL BASIC .NET

Si desea configurar el Visual Basic .Net, como por ejemplo, cambiar el tipo, tamaño o color de la letra des las instrucciones que escribe o el color del texto de cada comentario, etc., debe elegir la opción Herramientas/Opciones. Se visualiza una ventana con las secciones:

En la primera venta debe de seleccionar el elemento de Visual Basic que desea configurar.

En la *segunda ventana se muestran los datos* que puede configurar del elemento seleccionado.

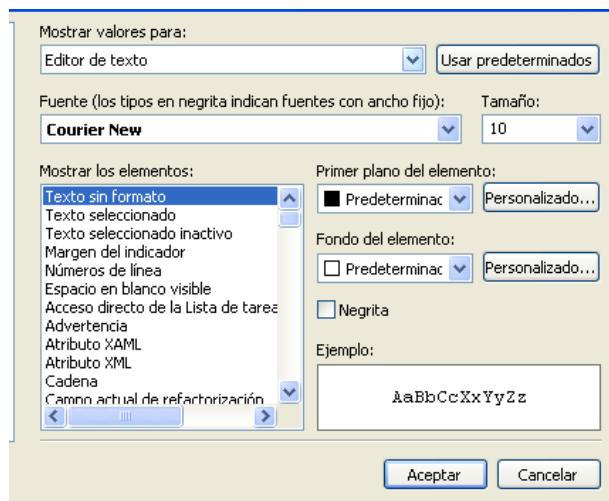
La siguiente sección se muestra cuando elige el elemento Entorno/General donde por ejemplo, puede indicar que se muestre o no la barra de estado.



La barra de estado indica si el Visual Basic .Net esta Listo para trabajar o esa realizando algún proceso, el numero de fila y columna donde se encuentra el cursor dentro de un programa en la ventana de código y si el teclado está en el modo Insert.



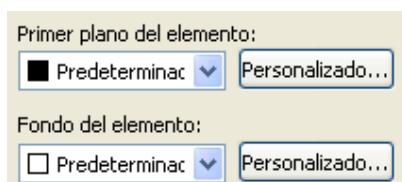
Si desea modificar el tipo de letra y los colores que utiliza el Visual Basic .Net, debe elegir de la primera sección la opción Entorno/Fuentes y colores. Se visualiza la siguiente sección:



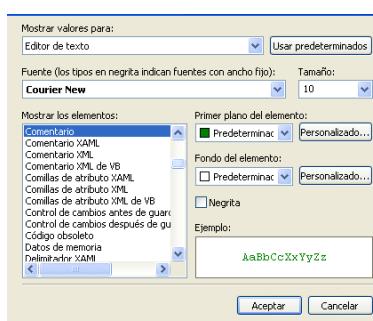
Para cambiar el tipo y el tamaño de letra que se utilizará para las instrucciones que escriba, seleccione Texto sin formato de la lista de elementos y luego seleccione Fuente y Tamaño.



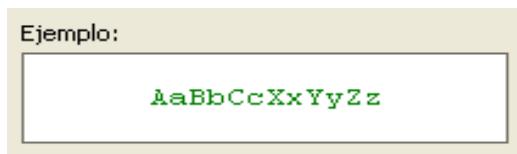
Para asignar colores utilice Primer Plano o Fondo.



En el siguiente ejemplo se ha seleccionado de la lista elementos la opción {comentario para configurar la letra de cada comentario que hagamos de un programa.



En la parte inferior de la sección se muestra un ejemplo del tipo, tamaño o color de la letra que usted asignó.

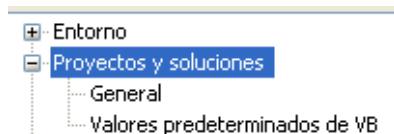


Si desea restaurar los valores predeterminados del Visual Basic .Net, haga clic en el botón Usar Predeterminados.

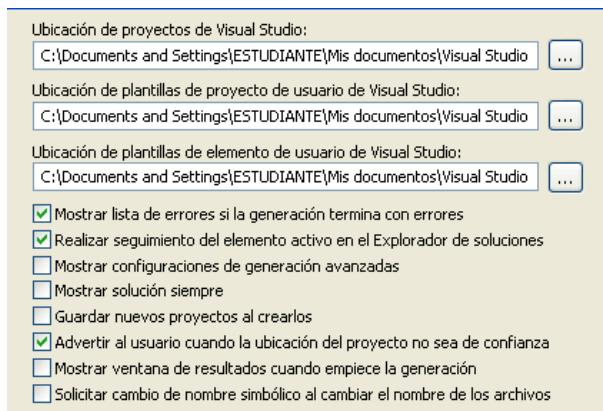
Usar predeterminados

Para guardar los cambios haga clic en el botón Aceptar.

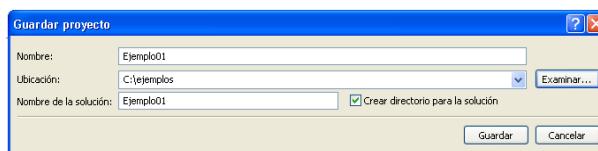
En la opción Proyectos y Soluciones/General puede indicar la carpeta donde se grabaran en forma predeterminada los proyectos que desarrolle así como las plantillas que use.



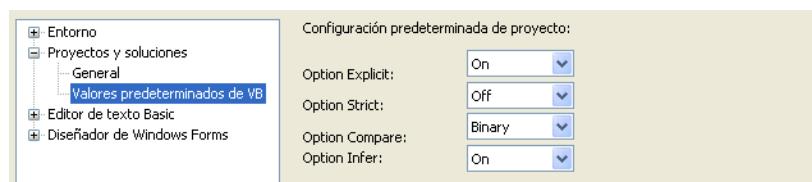
Al elegir esta opción se visualiza la siguiente sección:



Puede hacer clic en el botón Buscar (...) de cada una de las cajas para indicar la carpeta predeterminada. Cuando grabe un proyecto se visualizará la ventana con la carpeta que ha configurado.



La opción Proyectos y Soluciones/Valores predeterminados de Visual Basic .Net permite configurar los valores On u Off para:



Option Explicit

Permite indicar si es obligatorio o no declarar las variables que utilizamos dentro de un programa. El valor On indica que es obligatorio declarar las variables. Cuando el Visual Basic .Net esta configurado para que se declaren las variables, estas se visualizan subrayadas cuando no las declaramos.

```

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        N = 10.5
    End Sub

```

Si pasamos el puntero del mouse por la variable, se muestra un texto explicativo.

Al ejecutar el proyecto también se mostrara un mensaje de error.

```

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        N = 10.5
    End Sub

```

Option Strict

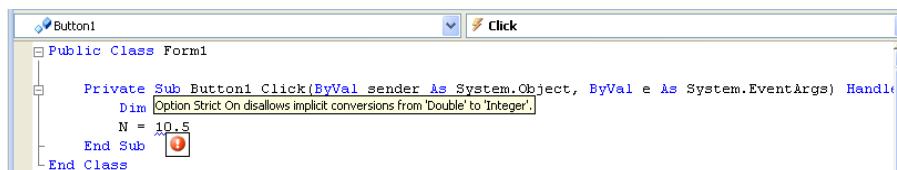
Permite configurar el Visual Basic .Net para que controle la conversación de datos. Cuando está en On no permite conversaciones donde se pierdan datos. Por ejemplo, en las siguientes instrucciones ha declarado la variable N para almacenar valores enteros pero se le está asignando un valor decimal, por lo que se muestra subrayada, indicando error.

```

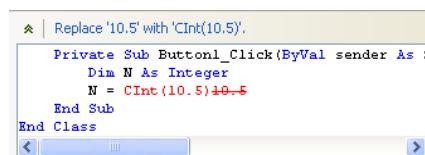
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        Dim N As Integer
        N = 10.5
    End Sub

```

Al pasar el mouse por el valor subrayado, se muestra el texto explicativo.



Junto al texto explicativo se muestra un ícono de admiración y al pulsar las teclas Shift + Alt F10 se muestra la solución al error cometido.



Option Compare

Permite configurar al Visual Basic .Net para el tipo de comparación de cadenas de caracteres. El tipo de comparación puede ser Binary o Text.

El tipo de comparación Binary compara cada carácter por su valor binario que lo representa. La característica principal de este tipo de comparación es que las letras mayúsculas con diferentes a las minúsculas.

Por ejemplo, las siguientes instrucciones muestran el mensaje: No son iguales si se ha configurado Comparación Binary.

```
If "AMOR" = "amor" Then
    MsgBox("Si son Iguales", MsgBoxStyle.Information, "Comparación")
Else
    MsgBox("No son Iguales", MsgBoxStyle.Information, "Comparación")
```

El tipo de comparación Text compara carácter por carácter.

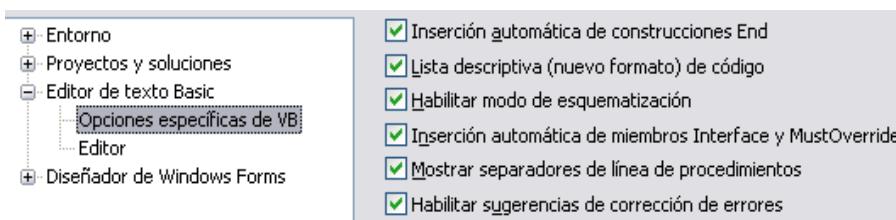


Característica principal de este tipo de comparación es que las letras mayúsculas son iguales a las minúsculas.

Las instrucciones muestran el mensaje S son Iguales, si se ha configurado el Visual Basic .Net para comparación de Text.



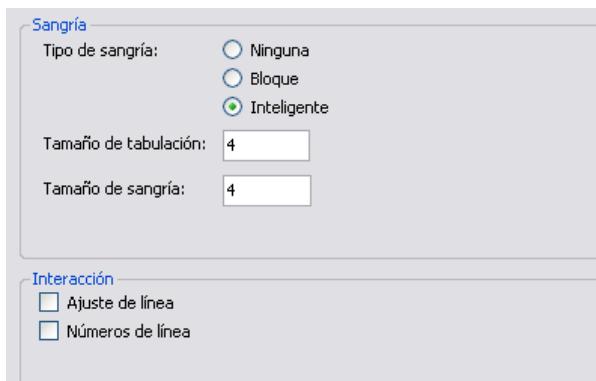
La opción Editor de texto Basic/Opciones específicas de VB .Net permite configurar el comportamiento del editor cuando escribimos las instrucciones.



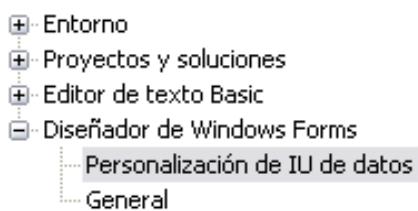
Por ejemplo, cada vez que forma la condición en una instrucción if y presiona la tecla Enter, automáticamente se escribe la instrucción End if. Lo mismo sucede con la instrucción For Next y similares.

```
If N > 10 Then          For a = 1 to 10
End If                  Next
```

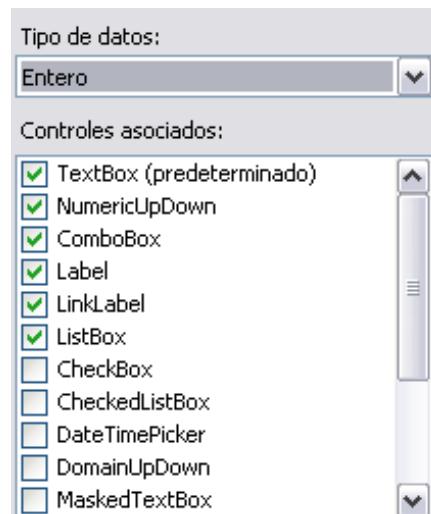
La opción editor de textos Basic/Editor permite configurar la sangría e interacción, es decir, las líneas de instrucciones que escribimos, por ejemplo, podemos activar la casilla para que se enumere cada línea.



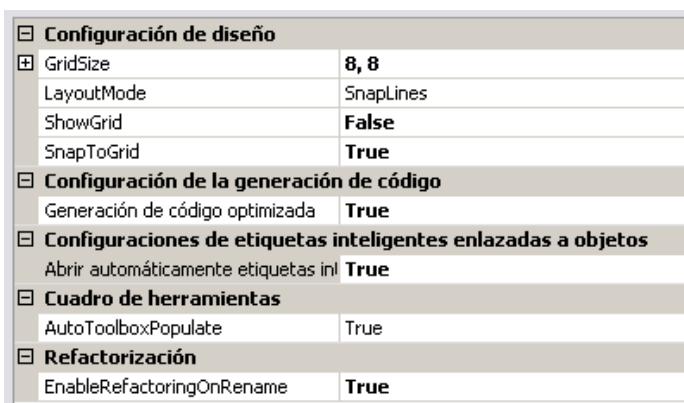
La opción Diseñador de Windows Forms/Personalización de IU permite configurar los controles que están asociados a cada uno de los tipos de datos de Visual Basic .Net



En el siguiente ejemplo se muestran los controles, cuyo contenido puede devolver un dato tipo entero.



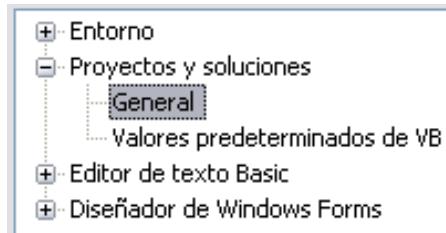
La opción diseño de Windows Forms/General permite configurar el formulario. Por ejemplo, puede asignar el valor true a la opción ShowGrid para que en los formularios se muestren con líneas en el modo de diseño.



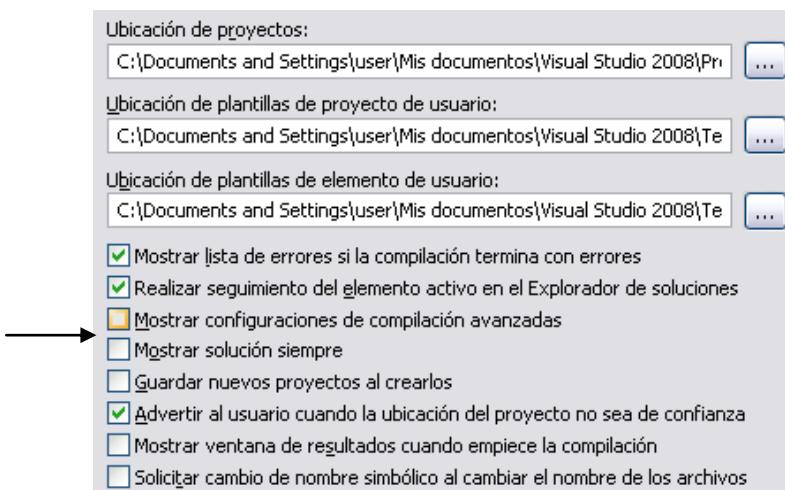
GRABAR UN PROYECTO

La forma de grabar un proyecto depende de cómo está configurado el Visual Basic .Net.

Este tipo de configuración se realiza ingresando a la opción Herramientas/Opciones del menú principal y luego ingresando a Proyectos y soluciones/General.



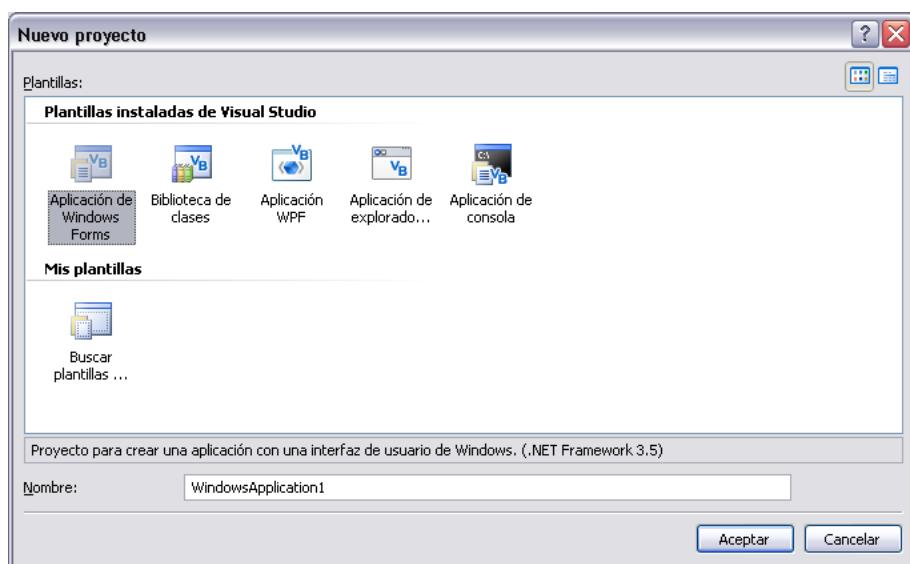
En la ventana que se visualiza existe una casilla llamada guardar nuevo proyecto al crearlo.



Si la casilla indica está desactivada, el proyecto se grabara sólo después de crearlo.



En este caso crear un nuevo proyecto, el Visual Basic .Net sólo le pedirá el nombre del proyecto.



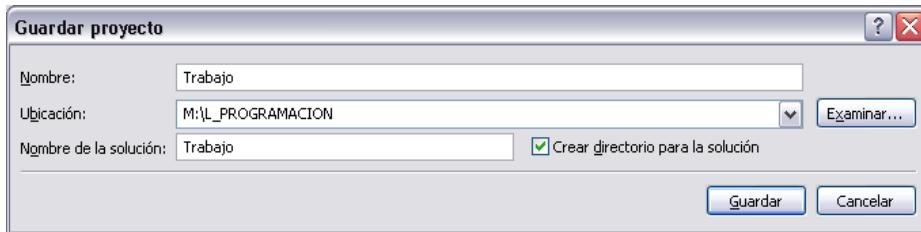
El nombre predeterminado de una aplicación es WindowsApplication, reemplácelo por el nombre que usted le desea asignar a su nueva aplicación y luego pulse la tecla Enter o haga clic en el Botón Aceptar.

Johan Guerreros Montoya

De esta manera se crea y se visualiza una nueva aplicación, pero no se grabará. Si desea grabar la aplicación después de crearla, debe hacer clic en la opción o en el botón Grabar Todo o pulsar las teclas Ctrl + Shift + S. si sólo desea grabar el formulario, puede hacer clic en la opción o en el botón Grabar Form.Vb o pulsar las teclas Ctrl. + S.

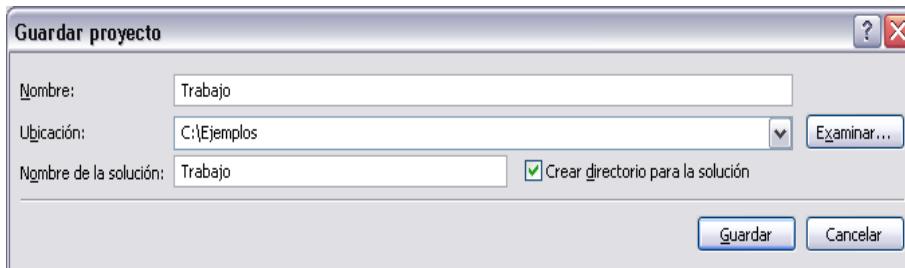


Con cualquiera de los pasos anteriores se visualiza la siguiente ventana:



En esta ventana puede hacer clic en el botón Examinar para buscar y/o seleccionar una nueva carpeta donde grabará el proyecto.

En la siguiente ventana de ejemplo se ha seleccionado la carpeta ejemplos en la Unidad C:\ y el nombre del proyecto es: Trabajo.



Visual Basic .Net crea en forma automática una carpeta con el nombre del proyecto dentro del cual graba todos los archivos de ese proyecto. En la ventana de ejemplo se observa que se ha creada la carpeta Trabajo dentro de C:\Ejemplos.



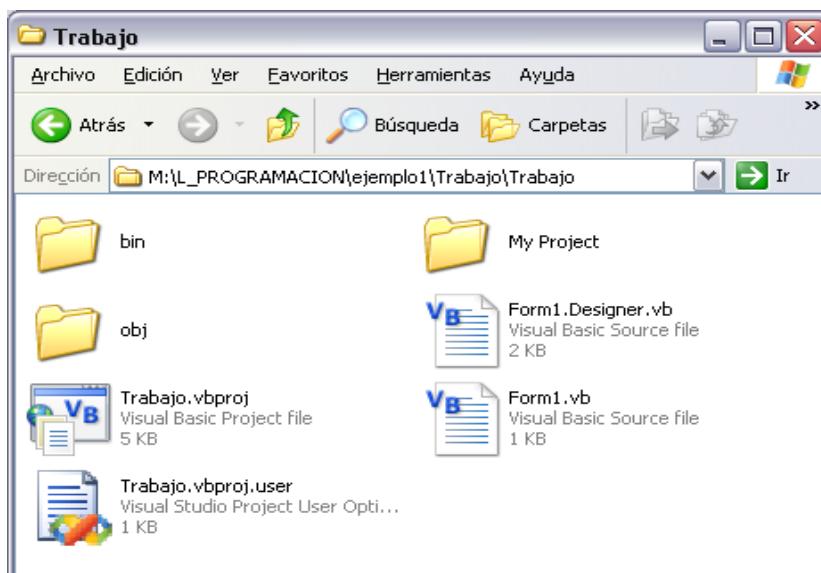
El contenido de la carpeta que se crea con el nombre del proyecto depende de activar o no la casilla: Crear directorio para solución.



Si la casilla esta activada se creará dentro de esa carpeta un archivo y una carpeta con el mismo nombre. El archivo es la solución y que permite abrir de manera directa el proyecto. La carpeta contiene todos los archivos del proyecto.

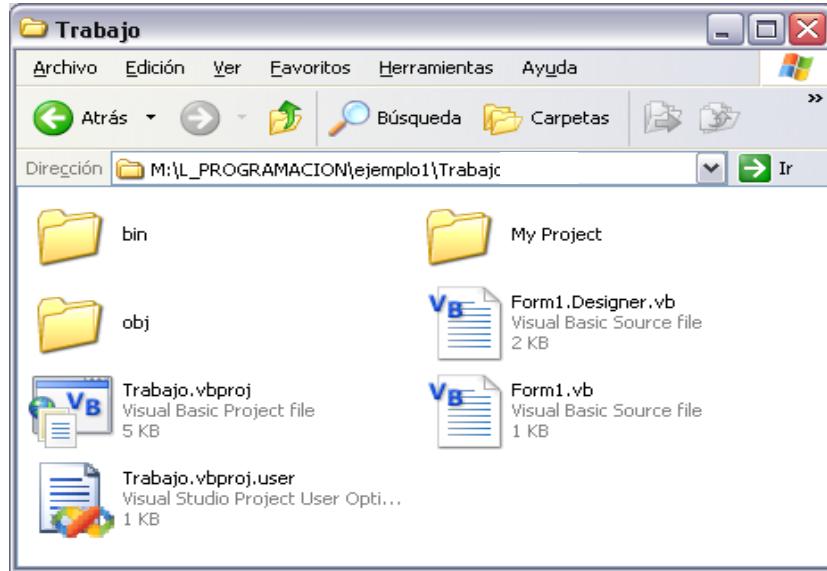


Contenido de la carpeta



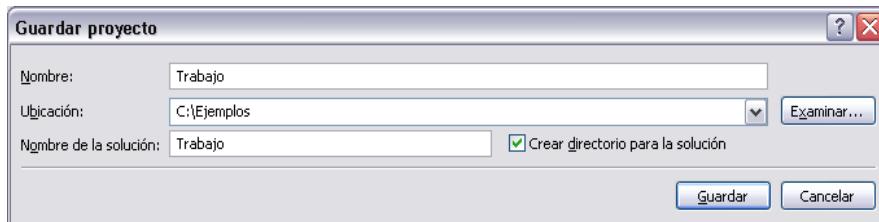
Si la casilla no esta activa se grabara dentro de la carpeta creada con el nombre del proyecto los archivos de esa aplicación junto con la solución, es decir, no se crea una carpeta adicional.



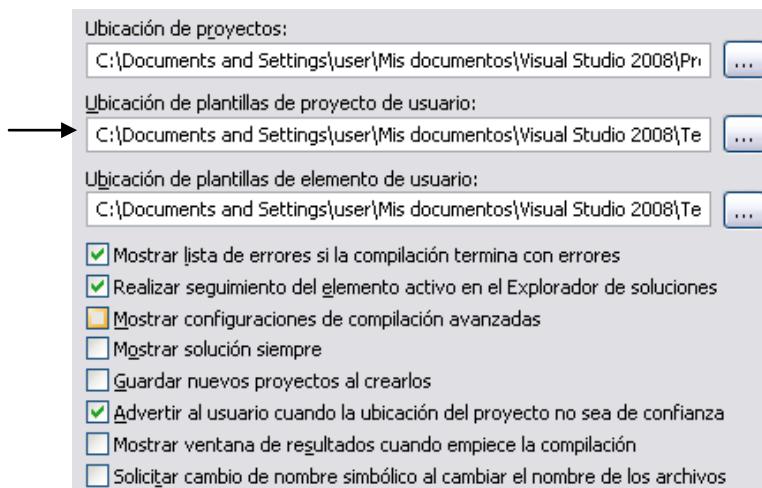


Algunos archivos del proyecto se encuentran en las carpetas Bin, Obj y My Proyect.

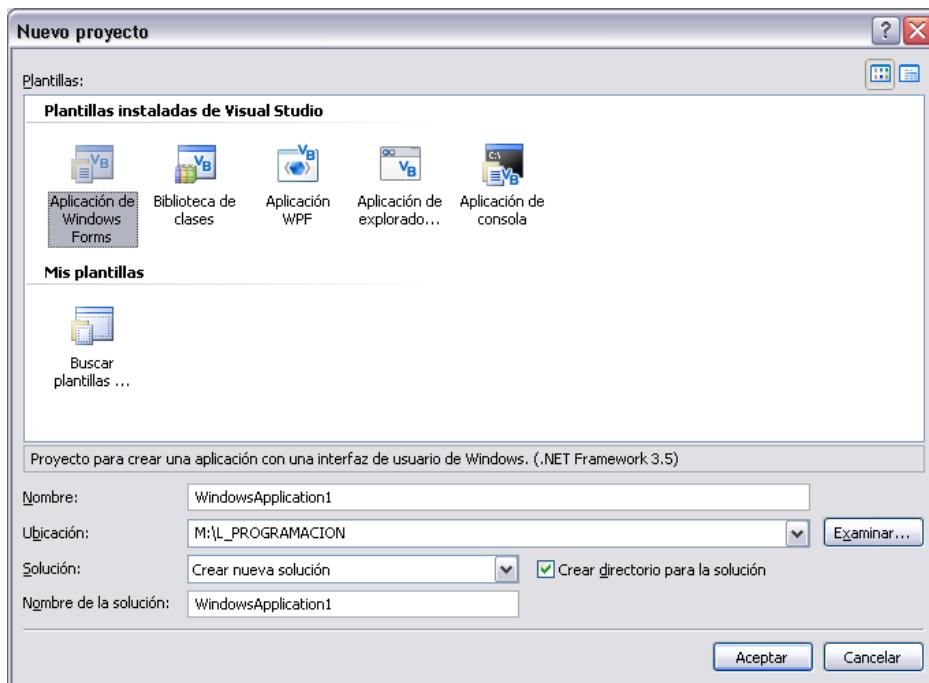
La carpeta que se visualiza es formal predeterminada al momento de grabar un proyecto (Ubicación) como se muestra en la siguiente ventana:



Depende de la ventana de configuración, en la caja: Ubicación de Proyectos de Visual Studio.



Si en la ventana de configuración anterior activamos la casilla: Guardar nuevos proyectos al crearlos.



El proyecto se grabara al momento de crearlo. La siguiente ventana visualiza cuando el Visual Basic .Net está configurado para grabar proyectos al momento de crearlo.



Los controles de aplicaciones en esta nueva ventana son los siguientes y los pasos para grabar el proyecto son los mismos.

ABRIR UN PROYECTO

Para abrir un proyecto lo puede hacer desde el explorador de Windows haciendo doble clic en el nombre de la solución o del proyecto que se crea al grabarlo.



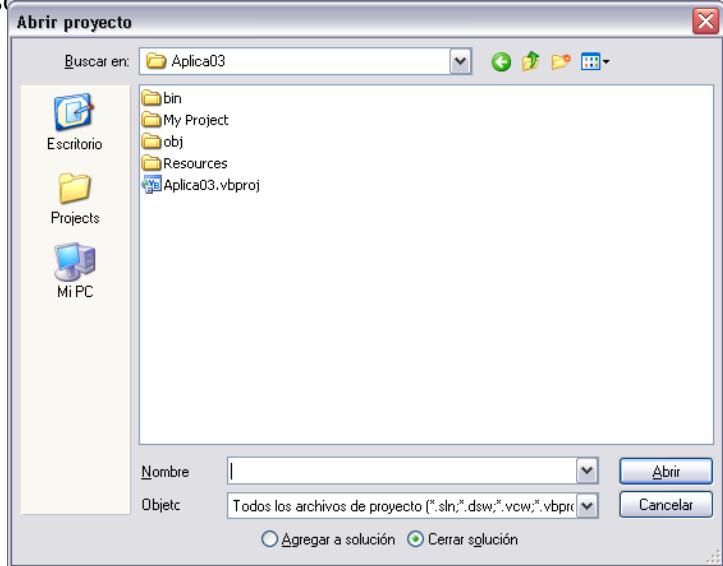
Nombre de la Solución

Nombre del Proyecto

Si se encuentra dentro del Visual Basic .Net y desea abrir un proyecto, puede hacer clic en el botón Abrir Archivo o en la opción Archivo/Abrir Archivo. Se visualiza la siguiente ventana donde debe seleccionar la carpeta donde gravo el proyecto.

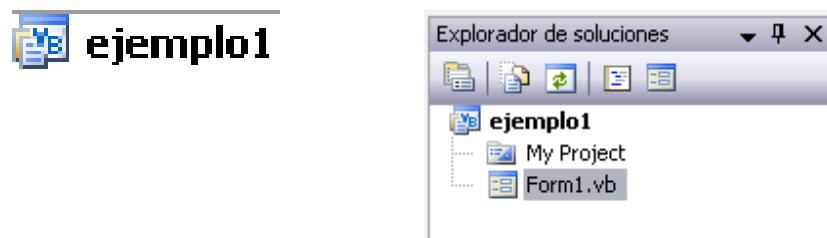
Botón Abrir Archivo



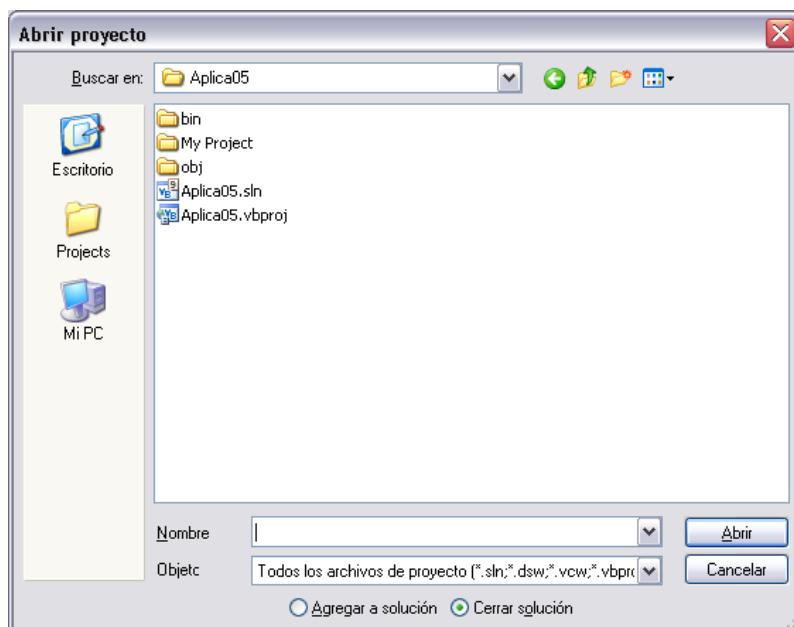


En esta ventana haga doble clic en el nombre de la solución o selecciónelo y luego haga clic en el botón Abrir. Al mostrarse el proyecto en la pantalla, haga doble clic en el nombre del formulario que desea visualizar en el explorador de soluciones.

Nombre de la Solución:



Una forma mas directa de abrir un proyecto es haciendo clic en la opción Archivo/Abrir Proyecto. En esta ventana, después de seleccionar la carpeta donde grabó el proyecto, visualiza el nombre del proyecto y de la solución creada. Para abrir el proyecto, haga doble clic en cualquiera de los archivos o seleccione cualquiera de ellos y haga clic en el botón Abrir.



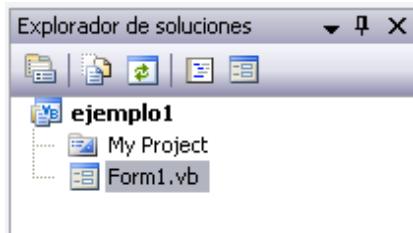
Nombre del Proyecto



Nombre de la Solución



Como en caso anterior, al mostrarse el proyecto en la pantalla, haga clic del explorador de soluciones, en el nombre del formulario que desea visualizar.



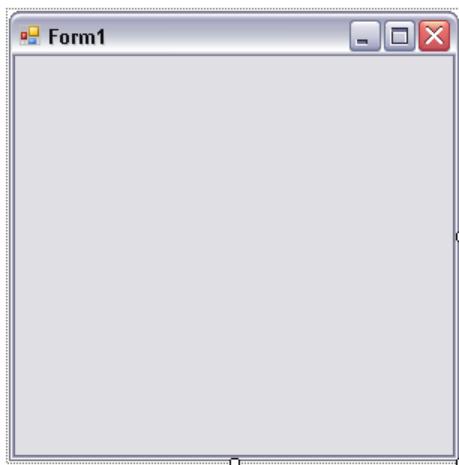
CONTROLES BASICOS DEL VISUAL BASIC .NET

Contenido:

n este capítulo, usted aprenderá a desarrollar sus primeras aplicaciones en EVisual Basic .Net y a utilizar sus controles Básicos.

- ***El Formulario***
- ***El Control Button***
- ***El Control Label***
- ***El Control TextBox***
- ***El Control ToolTip***
- ***El Control ContextMenuStrip***.

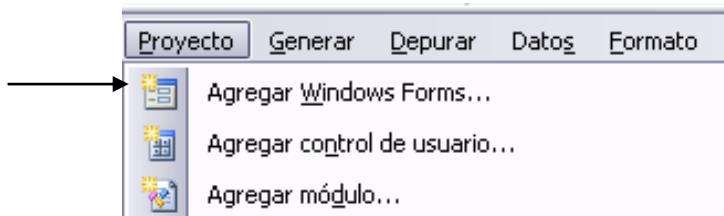
EL FORMULARIO



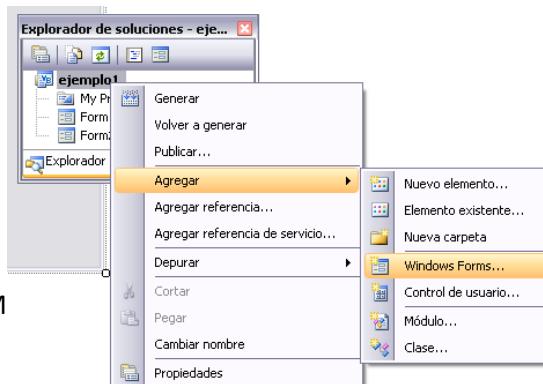
El formulario es el objeto principal de cada aplicación porque aquí se construye la interfaz del usuario, es decir, la comunicación del usuario con la aplicación que se desarrolla.

El desarrollo de una aplicación consiste en agregar los formularios necesarios y asignarles sus propiedades luego dibujar los controles que cada uno de ellos necesitan y asignarles también sus propiedades y finalmente escribir las instrucciones requeridas por la aplicación.

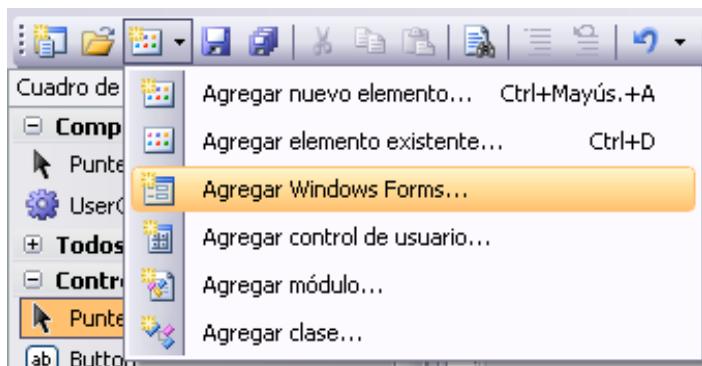
Cuando se crea una aplicación, en forma predeterminada se crea un formulario llamado Form1. Para agregar nuevos formularios a su aplicación puede elegir la opción Proyecto/Agregar Windows Forms.



Otra forma de agregar formularios a una aplicación es haciendo clic en el nombre del proyecto del explorador de soluciones y eligiendo la opción Agregar/Windows Forms.



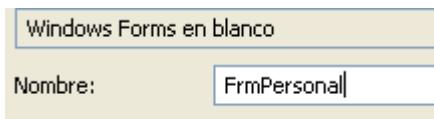
El botón Agregar elemento de la barra Estándar también tiene la opción Agregar Windows Forms.



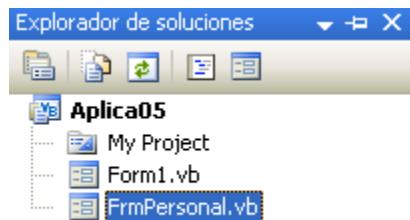
En la ventana que se visualiza seleccione Windows Forms y haga clic en el botón Agregar.



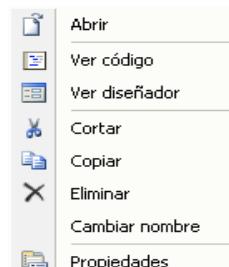
Si desea, antes de hacer clic en el botón Agregar puede asignarle un nombre diferente al predeterminado del nuevo formulario. En el siguiente ejemplo se está agregando un nuevo formulario con el nombre FrmPersonal



Cuando se agrega un nuevo formulario al proyecto, su nombre se visualiza en el explorador de soluciones.



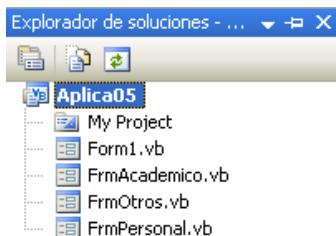
Para eliminar un formulario haga clic derecho en el explorador de soluciones y del menú contextual la opción Eliminar.



su nombre en el que se visualiza elija

Del mensaje que se visualiza, haga clic en el botón Aceptar para confirmar la eliminación

En la siguiente ventana de ejemplo, el proyecto está compuesto por cuatro formularios:

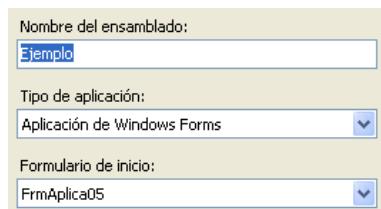


Como en versiones anteriores, al ejecutar el proyecto, se visualiza el primer formulario. Para indicar el formulario inicial puede elegir la opción Herramientas/Propiedades o haciendo clic derecho en el nombre de la aplicación de explorador de soluciones y eligiendo Propiedades.

La ventana que se visualiza tiene dos secciones. La primera sección contiene un conjunto de opciones de las cuales debe elegir Aplicación.



Cuando se elige la opción Aplicación, se visualiza dentro de la segunda sección lo siguiente:



En esta sección debe elegir en Formulario de Inicio el formulario que desea ejecutar primero.

Principales propiedades de los formularios

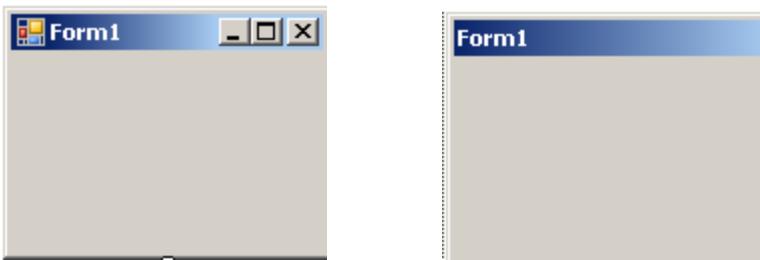
Name

Se utiliza para asignarle un nombre al formulario. Este nombre también se le puede asignar al momento de agregar el formulario



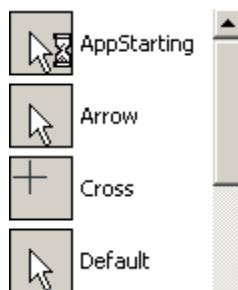
ControlBox

Esta propiedad permite mostrar o no los botones de control del formulario.



Cursor

Se utiliza para seleccionar el tipo del puntero del mouse cuando se pase por el formulario. Ejemplo.



FormBorderStyle

Esta propiedad permite configurar el estilo del borde del formulario. Del estilo que se elija depende los botones que se visualice y su comportamiento cuando se ejecuta. Ejemplos:

FixedSize (Normal) None FixedToolWindow



MaximizeBox

Se utiliza para indicar si el formulario debe mostrar el botón maximizar.

MaximumSize

Se utiliza para indicar el tamaño máximo que puede tener el formulario.

MinimizeBox

Se utiliza para indicar si el formulario debe mostrar el botón minimizar.

MinimumSize

Se utiliza para indicar el tamaño mínimo que puede tener el formulario.

StartPosition

Se utiliza para indicar la posición del formulario cuando se ejecute.

Text

Se utiliza para escribir el título del formulario.

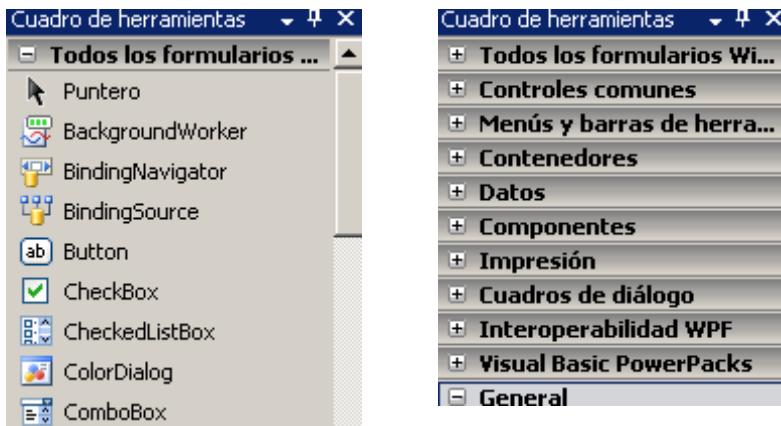
WindowState

Se utiliza para indicar el tamaño iniciar del formulario cuando se ejecute.

Dibujar Controles en el Formularios

Los controles se encuentran en el cuadro de herramientas agrupados en fichas. Usted puede expandir la ficha según el tipo de control que necesita. Si desea visualizar todos los controles expanda la ficha:

Todos los formularios Windows Forms



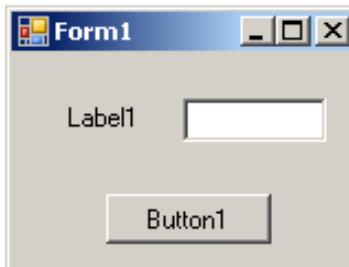
Existen dos formas de dibujar controles en un formulario:

- 1. Haciendo doble clic en el control.**

En este caso los controles que va dibujando se van ubicando en la parte superior izquierda del formulario.

- 2. Arrastrando el control desde el cuadro de herramientas al formulario.**

Después de dibujar los controles debe asignarles sus propiedades.



EL CONTROL BUTTON

Este control, como en versiones anteriores del Visual Basic .Net, permite escribir instrucciones, las cuales se ejecutan normalmente cuando se hace clic en dicho control.

Algunas de sus propiedades han cambiado y se han agregado nuevas.

Name

Esta propiedad permite asignarle un nombre al control. Se recomienda que su nombre empiece con las letras Btn.

Image

Esta propiedad permite asignarle un grafico al botón.



ImageAlign

Esta propiedad permite linear el grafico dentro del control. En Visual Basic .Net existen 09 formas de alinear, como se muestra a continuación.



Text

Esta propiedad permite asignarle un título al botón. En esta propiedad se puede utilizar también el símbolo & para poder ejecutar las instrucciones pulsando la tecla Alt y la letra subrayada del título del botón.

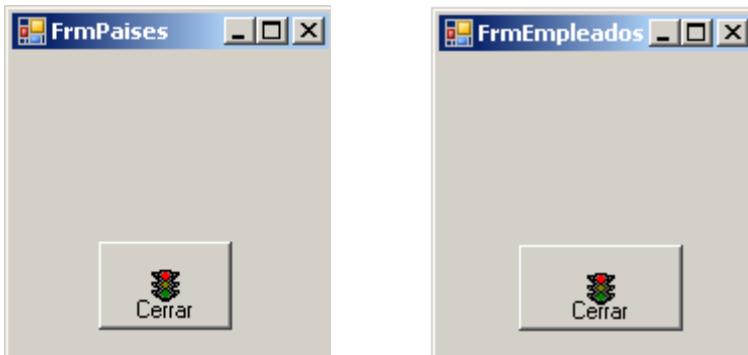
TextAlign

Esta propiedad permite alinear el título del botón.

Aplicación Desarrollada Nº II-01



Esta aplicación permite llamar a 03 formularios desde un formulario principal. El formulario principal se llama FrmPrincipal y los otros: FrmPaises, FrmEmpleados y FrmProductos. Ejemplo:



En esta aplicación de ejemplo, se utiliza el método Show para mostrar formularios.

El formulario principal esta compuesto por 04 botones de comandos llamados: BtnPaises, BtnEmpleados, BtnProductos y BtnFinalizar.



Cada uno de los botones tiene asociado un grafico utilizando su propiedad image. Los tres primeros botones tiene el grafico alineado a la mitad y lado izquierdo (MiddleLeft). El texto de cada uno de ellos están alineados a la mitad y centro (MiddleCenter).

El botón BtnFinalizar tiene el grafico alineado a la mitad y centro (MiddleCenter) y su texto en el centro de la parte inferior (BottomCenter).

El resto de formularios solo tiene un botón llamado BtnCerrar con un grafico a la mitad y centro (MiddleCenter) y su texto en el centro de la parte inferior (BottomCenter).



Instrucciones de los botones del formulario FrmPrincipal.

```

(General) (Declaraciones)
Public Class Form1
    Private Sub BtnPaises_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BtnPaises.Click
        FrmPaises.Show()
    End Sub

    Private Sub BtnEmpleados_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BtnEmpleados.Click
        FrmEmpleados.Show()
    End Sub

    Private Sub BtnProductos_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BtnProductos.Click
        FrmProductos.Show()
    End Sub

    Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
        Me.Close()
    End Sub
End Class

```

Dentro de cada uno de los botones BtnCerrar del resto de formularios se utiliza el método Close.

Close()

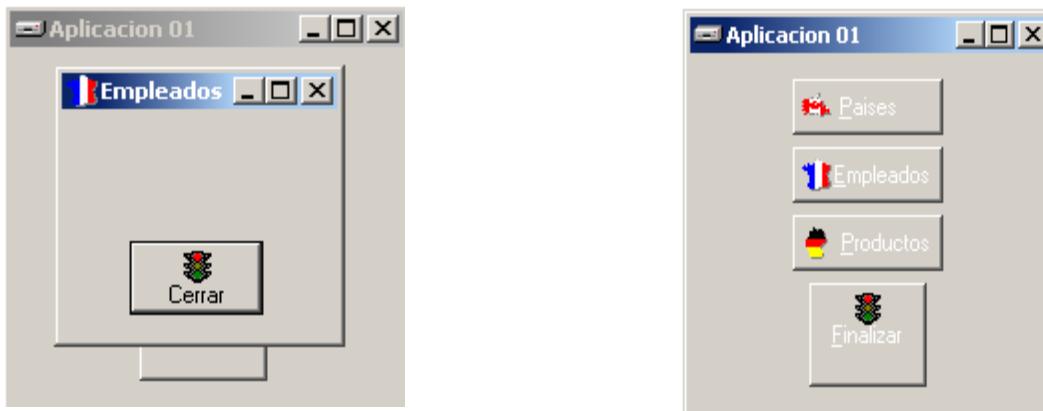
Para llamar y utilizar un formulario, también se le puede crear una referencia. Por ejemplo, para llamar al formulario FrmEmpleados se pueden escribir las siguientes instrucciones.

Dim F as New FrmEmpleados

F.Show()

El método Show muestra el formulario en modo No Modal, esto quiere decir, que después de llamar a un formulario principal sin cerrar el formulario secundario.

Por ejemplo, si se tiene los dos formularios y haces clic en cualquier parte del formulario principal, éste queda activado y el formulario FrmEmpleados minimizado en la barra de tareas.



Barra de Tareas

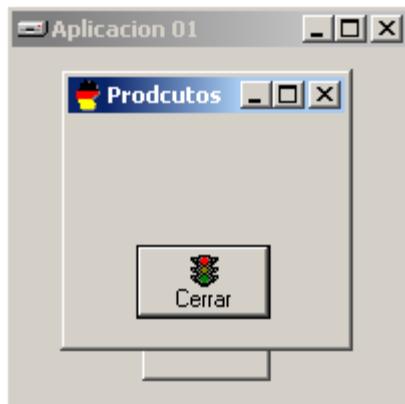


Si queremos llamar a los formularios en modo Modal, es decir, que solo se active el formulario principal cuando cerramos el formulario secundario, debemos utilizar el método ShowDialog.

Las siguientes instrucciones muestra el formulario FrmProductos en modo Modal.

```
Private Sub Btnproductos_Click(ByVal sender As System.Object
    Frmproductos.ShowDialog()
End Sub
```

De esta manera cuando se haga clic en cualquier parte del formulario principal no se activará hasta cerrar el formulario FrmProductos



A cada uno de los botones podemos asignarle un texto explicativo para que se visualice cuando se pase el puntero del mouse por el control.

Para esto se debe utilizar el control **ToolTip**.

EL CONTROL TOOLTIP

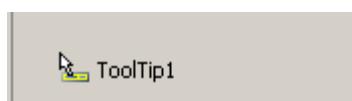
El control **ToolTip** permite establecer establecer un texto explicativo para otros controles cuando el usuario pase el puntero del mouse por el control.

En el siguiente ejemplo se ha asignado un texto explicativo a los botones **BtnPaises** y **BtnFinalizar**.



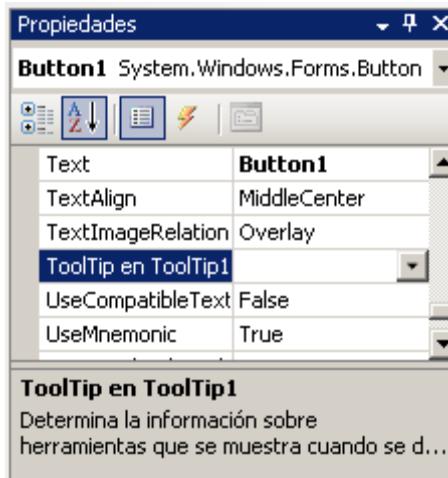
Para poder asignarle a los controles el texto explicativo, debe dibujar en su formulario el control **ToolTip**.

Cuando dibuja el control **ToolTip**, Se ubica en la parte inferior del formulario.



Después de dibujar el control **ToolTip**, en su formulario se agrega automáticamente la propiedad **ToolTip** en **ToolTip1** a los controles que contiene el formulario.

La siguiente ventana de propiedades pertenece al botón BtnPaises después de haber dibujado el control ToolTip en su formulario.



Antes de dibujar el control ToolTip, no existía la propiedad ToolTip en ToolTip1.



Adicionalmente el texto explicativo asignado a cada botón, podemos hacer que el color de fondo de cada uno de ellos cambie de color cuando el usuario pase el puntero del mouse sobre cualquier botón.

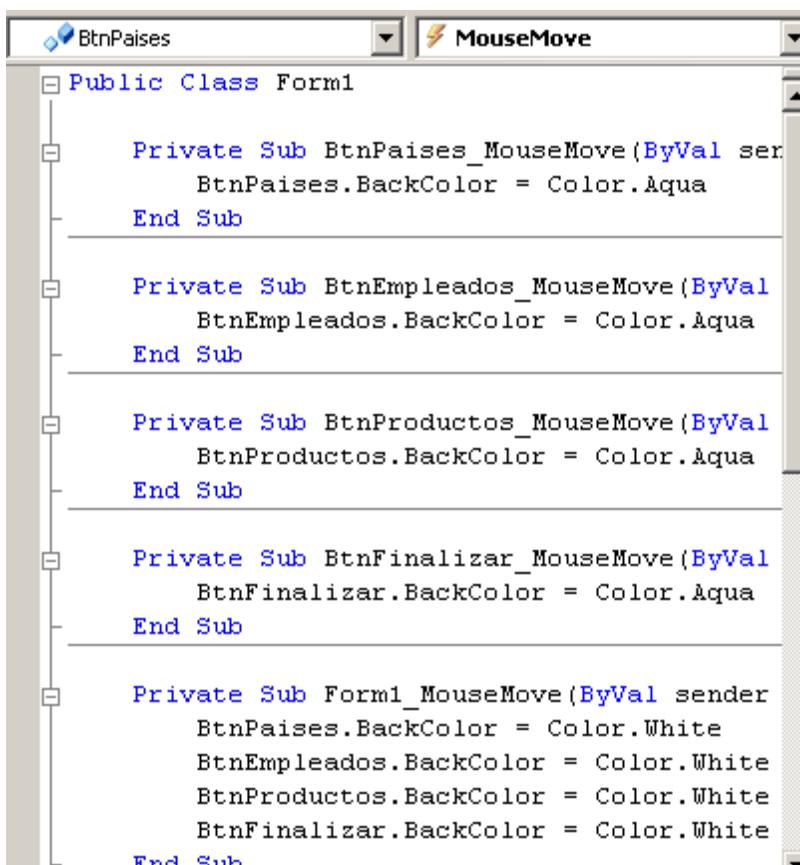


En los siguientes ejemplos se muestra el cambio del color de cada botón en el formulario principal.



Para lograr la activación de colores utilizamos la propiedad BackColor de cada botón y el evento MouseMove del formulario y de los botones.

Instrucciones del evento MouseMove de cada botón y del formulario.



```

BtnPaises
MouseMove

Public Class Form1

    Private Sub BtnPaises_MouseMove(ByVal sender As System.Object, ByVal e As System.EventArgs)
        BtnPaises.BackColor = Color.Aqua
    End Sub

    Private Sub BtnEmpleados_MouseMove(ByVal sender As System.Object, ByVal e As System.EventArgs)
        BtnEmpleados.BackColor = Color.Aqua
    End Sub

    Private Sub BtnProductos_MouseMove(ByVal sender As System.Object, ByVal e As System.EventArgs)
        BtnProductos.BackColor = Color.Aqua
    End Sub

    Private Sub BtnFinalizar_MouseMove(ByVal sender As System.Object, ByVal e As System.EventArgs)
        BtnFinalizar.BackColor = Color.Aqua
    End Sub

    Private Sub Form1_MouseMove(ByVal sender As System.Object, ByVal e As System.EventArgs)
        BtnPaises.BackColor = Color.White
        BtnEmpleados.BackColor = Color.White
        BtnProductos.BackColor = Color.White
        BtnFinalizar.BackColor = Color.White
    End Sub

```

EL CONTROL LABEL

El control label permite mostrar mensaje o resultados de algún cálculo numérico en el formulario.

Sus principales propiedades son:

Name

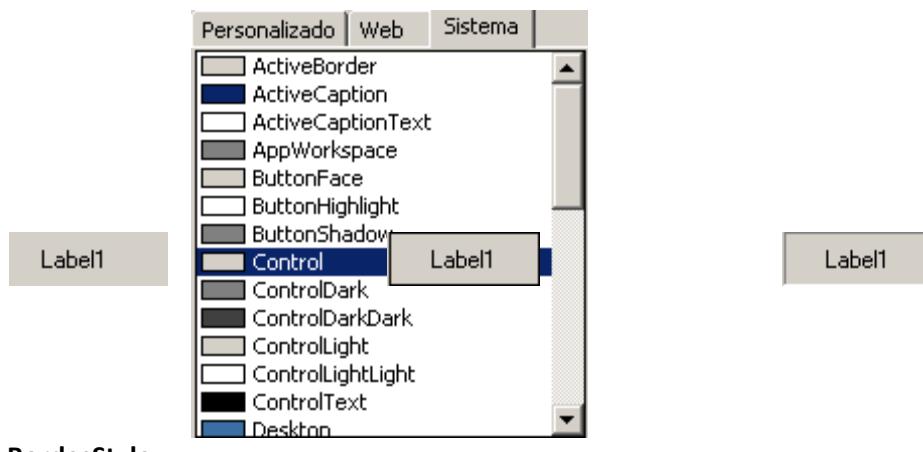
Esta propiedad permite asignarle un nombre al control. Se recomienda que su nombre empiece con las letras Lbl.

AutoSize

Permite indicar si el tamaño del control debe ajustarse en forma automática a la calidad y tamaño de las letras que contiene. En forma predeterminada tiene el asignado el valor True. Para poder cambiarle su tamaño en tiempo de diseño debe asignarle el valor False a esta propiedad.

BackColor

Permite asignar un color de fondo. El color se puede asignar desde tres hojas y el predeterminado se llama Control que se encuentra en la hoja Sistema.



BordesStyle

Esta propiedad permite cambiar el estilo del borde del control. Son tres estilos:

None

FixedSingle

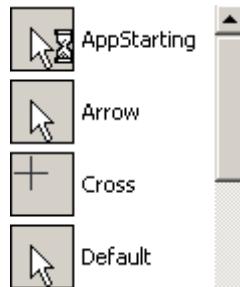
Fixed3D

ContextMenuStrip

Se utiliza cuando deseamos mostrar un menú contextual cuando el usuario haga clic derecho sobre el control. El menú contextual debe estar creado previamente, utilizando el control ContextMenuStrip. En el siguiente ejemplo se muestra un menú contextual sobre una etiqueta.

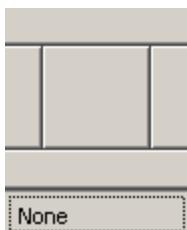
Cursor

Se utiliza para seleccionar el tipo del puntero del mouse cuando se pase por el control. Algunos tipos de punteros son:



Dock

Esta propiedad permite indicar la forma de acoplamiento del cursor. Las formas son las siguientes:

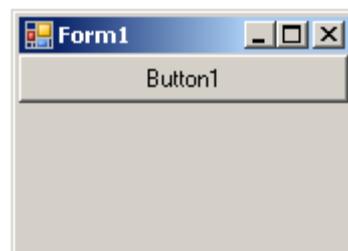


La opción None ubica el control donde se dibujo. Los siguientes ejemplos se muestran utilizando un botón de comando:

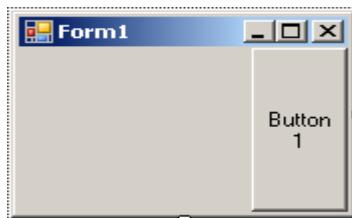
None



Top



Right

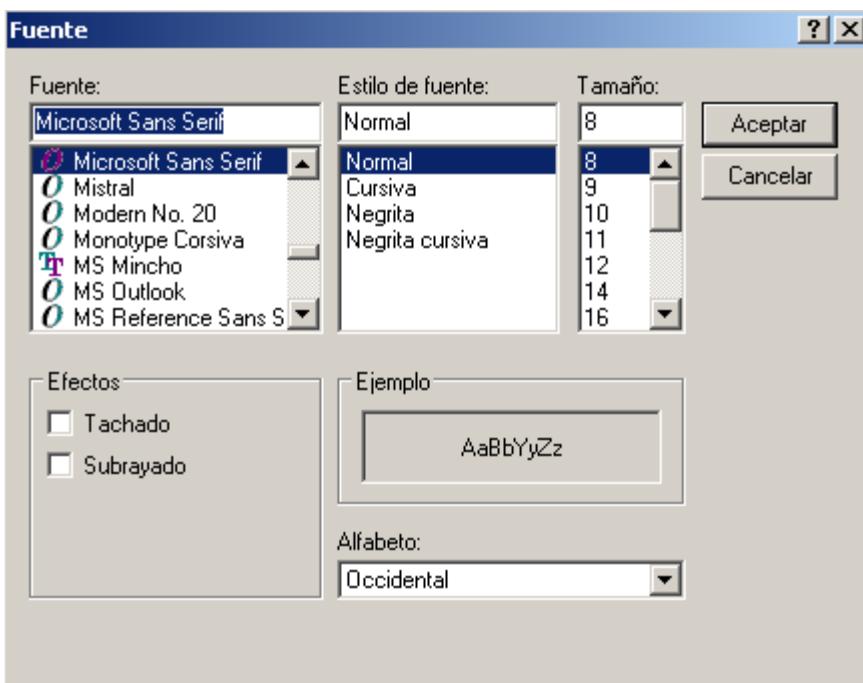


Fill



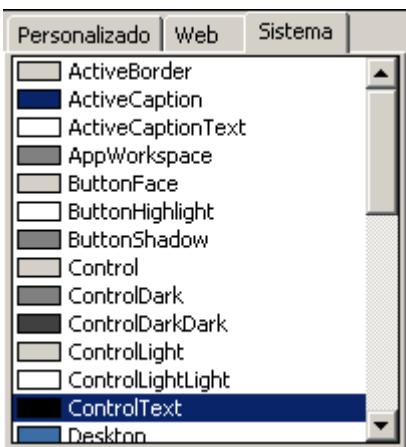
Font

Esta propiedad permite asignar el tipo, estilo y tamaño de letra, así como algunos efectos con la que el control mostrara la información.



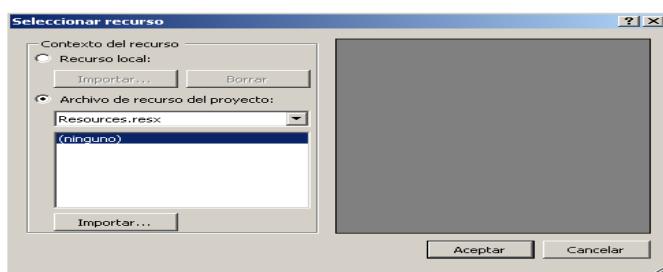
ForeColor

Permite asignar un color a la letra del control. El color se puede seleccionar desde tres hojas y el predeterminado se llama ControlText que se encuentra en la hoja Sistema. La ventana que visualiza es la misma que la propiedad BackColor.



Image

Esta propiedad permite seleccionar un grafico para que se muestre en el control. Al ingresar a



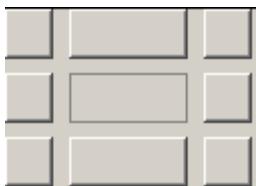
esta propiedad se muestra la siguiente ventana:

El botón Importar permite seleccionar el grafico. Si elegimos la segunda opción, los gráficos que seleccionemos formaran parte del proyecto y se agregarán al explorador de soluciones.

El ejemplo muestra 2 imágenes seleccionadas con la opción: Archivo de cursos del proyecto.

ImageAlign

Esta propiedad permite alinear el grafico dentro del control. Tiene las siguientes opciones de alineación:



ImageIndex

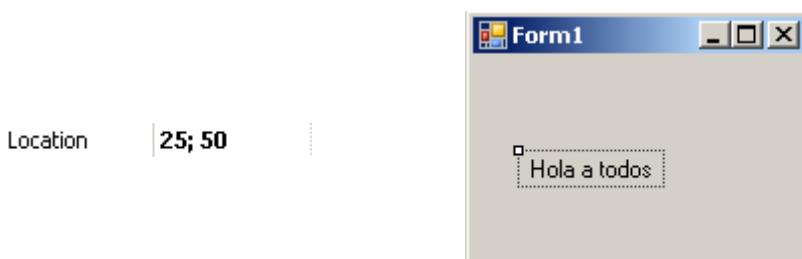
Esta propiedad permite indicar el numero del grafico almacenado en un control ImageList que debe mostrar el control Label. El primer elemento es el numero cero.

Para que esta propiedad funcione debe de agregar un control ImageList con algunos gráficos al proyecto y asignar el nombre de esa ImageList en la propiedad ImageList del control Label.
Ejemplo:



Location

Se utiliza para indicar la posición del control en el formulario. Se le debe asignar dos valores separados por un punto y coma. Ambos valores se refieren a la esquina superior derecha del control. El primer valor indica la posición horizontal y el segundo la posición vertical.



MaximumSize

Se utiliza para indicar el tamaño máximo que puede tener el control. Acepta dos valores separados por un punto y coma. El primer valor indica el tamaño máximo horizontal y el

segundo el tamaño mínimo vertical. Si no se desea asignar límite, ambos valores deben ser ceros.

MaximumSize | 0; 0

MinimumSize

Se utiliza para indicar el tamaño mínimo que puede tener el control. Acepta dos valores separados por un punto y coma. El primer valor indica el tamaño mínimo horizontal y el segundo el tamaño mínimo vertical. Si no se desea asignar límite, ambos valores deben ser ceros.

MinimumSize | 0; 0

Padding

Esta propiedad se utiliza para indicar el espacio interior del control, es decir, el espacio entre los márgenes y el texto gráfico que visualicen.

En el siguiente ejemplo se muestra un control Label con espacios interiores:



Size

Esta propiedad se utiliza para indicar el tamaño del control. Aceptar dos valores separados por punto y coma que representan la esquina inferior derecha. Para poder cambiar estos valores le debe asignar el valor False a la Propiedad AutoSize.

AutoSize	False	Size	35; 50
----------	-------	------	--------

TabIndex

Esta propiedad se utiliza para indicar el orden de ubicación del cursor en los controles cada vez que se pulse la tecla Tab.

Tag

Se utiliza para almacenar algún valor que podemos usar dentro de la ejecución de un programa.

Text

Esta propiedad permite almacenar el texto o valor que el control debe mostrar. En versiones anteriores los controles Label usaban la propiedad Caption.

TextAlign

Esta propiedad permite alinear la información que se muestra en un control. Tiene las siguientes opciones de alineación:



UseMnemonic

Esta propiedad permite indicar si se puede acceder a la etiqueta o control dibujado, inmediatamente después de la etiqueta al pulsar la tecla precedida por el símbolo &.

UseWaitCursor

Esta propiedad permite indicar si se cambia la propiedad Cursor del control al valor WaitCursor. Ejemplo:



Visible

Esta propiedad permite indicar si el control se debe visualizar o no cuando se ejecute la aplicación.

Aplicación Desarrollada Nº II-02



Esta aplicación permite mostrar el nombre, edad y distrito de una persona. Se utiliza un control Label llamado LblData y cuatro botones de comandos llamados: BtnNombre, BtnEdad, BtnDistrito y BtnFinalizar.

El control LblDatos tiene en su propiedad AutoSize el valor False, en su propiedad TextAlign el valor MiddleCenter y en su propiedad Font el tamaño 12 y negrita.

(Name)	LblDatos	AutoSize	False	ImageAlign	MiddleCenter
--------	-----------------	----------	-------	------------	--------------

Al hacer clic en un botón se muestra el dato respectivo.



Instrucciones de los botones de comandos.

```

(General) (Declaraciones)

Public Class Form1

    Private Sub BtnNombre_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BtnNombre.Click
        LblDatos.Text = "Juan Jose"
    End Sub

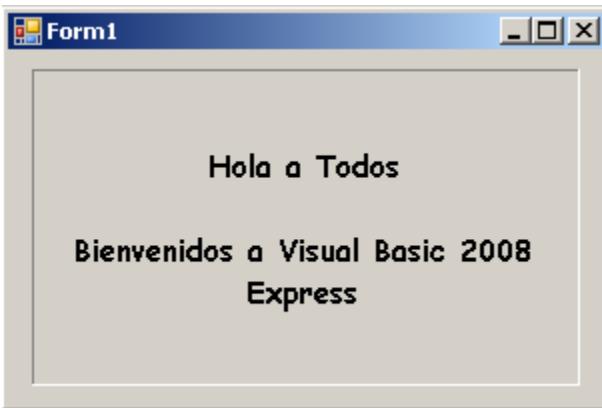
    Private Sub BtnEdad_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BtnEdad.Click
        LblDatos.Text = "33 Años"
    End Sub

    Private Sub BtnDistrito_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BtnDistrito.Click
        LblDatos.Text = "Cerro Colorado"
    End Sub

    Private Sub BtnFinalizar_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BtnFinalizar.Click
    End
    End Sub
End Class

```

Aplicación Desarrollada Nº II-03



Este programa utiliza una etiqueta (LblSaludo) y el evento Form_Load para mostrar un saludo al ejecutarlo.

Para desarrollar este programa solo debe dibujar en un formulario y una etiqueta llamada LblSaludo y asignarle en su propiedad AutoSize el valor False.

Instrucciones del evento Load del formulario.

'Asigna el titulo al formulario

Me.Text= "Mi programa de Ejemplo"

"Alinea el texto del saludo al centro.

LblSaludo.TextAlign= ContentAlignment.MiddleCenter

"Muestra el saludo en dos líneas diferentes

LlblSaludo.Text= "Hola a Todos" &Chr(13) & Chr(13) & "Bienvenidos a Visual Basic 2008 Express"

EL CONTROL TEXT BOX



Este control es utilizado para ingresar datos de una manera muy sencilla en una aplicación.

Sus principales propiedades son:

Name

Esta propiedad permite asignarle un nombre al control. Se recomienda que su nombre empiece con las letras Txt.

AcceptsReturn

Esta propiedad se utiliza para indicar si en este control se debe pasar a la siguiente línea cuando se pulse la tecla Enter si esta configurado para escribir en múltiples líneas.

AcceptsTab

Esta propiedad se utiliza para indicar si el cursor debe pasar al siguiente control cuando el usuario pulse la tecla Tab.

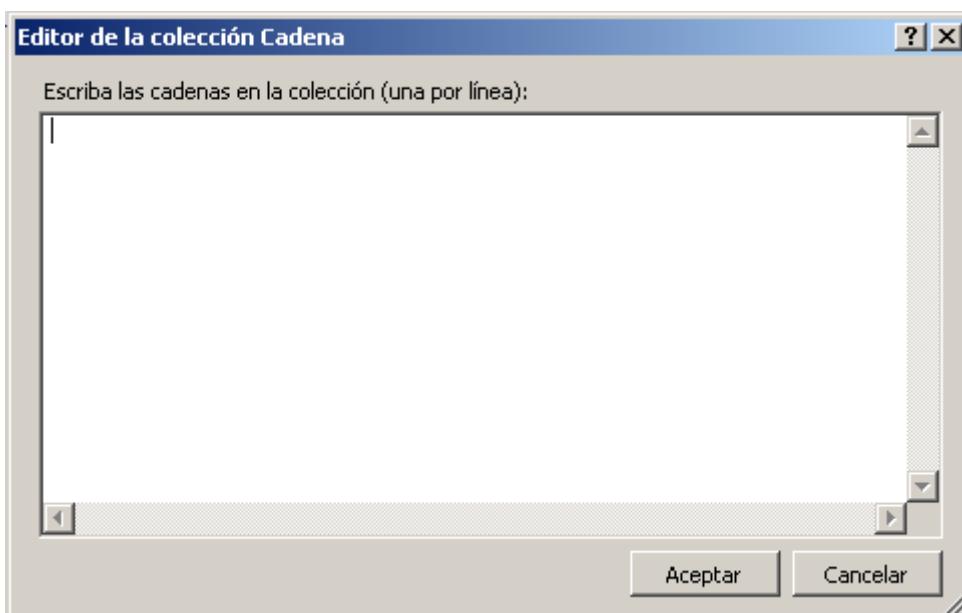
AutoCompleteCustomSource

Esta propiedad se utiliza cuando deseamos que el control TextBox autocomplete alguna palabra que podemos escribir la letra J se auto completa con los meses que empiecen con esa inicial o iniciales.

Al activar esta propiedad se visualizar la palabra Colección y un botón con tres puntos.

AutoCompleteCustomSource **(Colección)**

Al hacer clic en el botón con tres puntos se visualiza la siguiente ventana donde debe escribir sus iniciales.



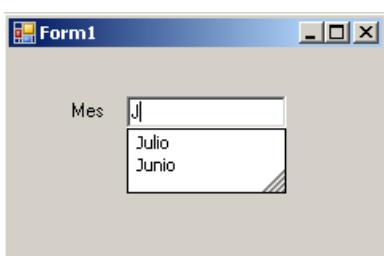
AutoCompleteMode

Esta propiedad se utiliza para indicar la forma como se deben auto completar las palabras en el control TextBox y son las siguientes:

Suggest

Append

SuggestAppend



AutoCompleteSource

Esta propiedad se utiliza para indicar el origen de las palabras que se deben auto completar en el control TextBox.

Johan Guerreros Montoya

Elija CustomSource para que se utilicen las palabras que ha escrito en la propiedad AutoCompleteCustomSource.

Para el programa de ejemplo se han escrito los meses del año en la propiedad AutoCompleteCustomSource.



Y se han asignado los siguientes valores a las propiedades:



BorderStyle

Esta propiedad permite asignar un borde al control TextBox. Los tipos de bordes son los siguientes:



CharacterCasing

Esta propiedad permite indicar si las letras que el usuario escribe en el control TextBox deben quedar como las escribe o convertirse a mayúsculas o minúsculas.

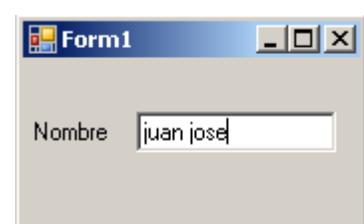
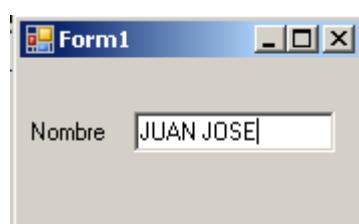
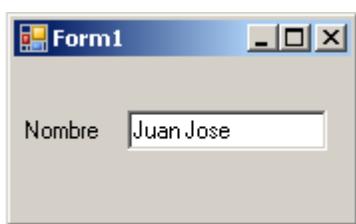


La opción Normal deja las letras como el usuario las escribe. Upper las convierte a mayúsculas y Lower las convierte a minúsculas.

Normal

Upper

Lower



ContextMenuStrip

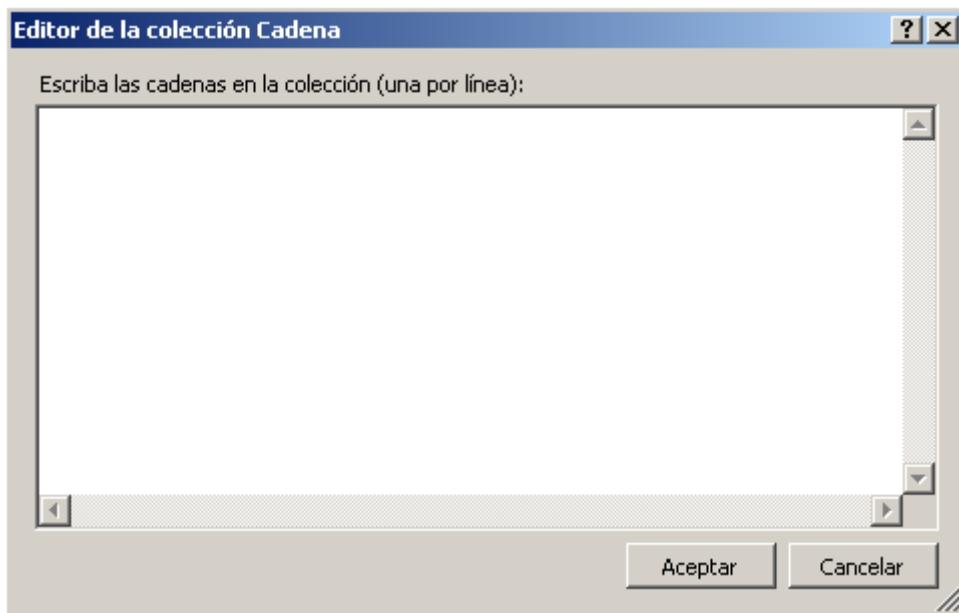
Se utiliza cuando deseamos mostrar un menú contextual cuando el usuario haga clic derecho sobre el control. El menú contextual debe estar creado previamente utilizando el control ContextMenuStrip.

Cursor

Se utiliza para seleccionar el tipo del puntero del mouse cuando se pase por el control. El predeterminado en este control es IBeam.

Lines

Esta propiedad se utiliza para ingresar el texto en modo de diseño cuando el control TextBox esta configurado para utilizar múltiples líneas. Al ingresar a esta propiedad se muestra la siguiente venta donde debe escribir el texto deseado.



Locked

Esta propiedad se utiliza para indicar si se puede mover, cambiar el tamaño del control en tiempo de diseño. Cuando se le asigna el valor True a esta propiedad, el control no se podrá mover ni cambiar de tamaño y el control se visualizara con un candado en la parte superior derecha.

MaxLength

Esta propiedad se utiliza para establecer la cantidad máxima de caracteres que debe aceptar el control TextBox. La cantidad predeterminada de caracteres que acepta el control es: 32,767.



Es muy importante establecer la cantidad máxima de caracteres para los datos que se ingresan en un control texto y que se van a grabar en una base de datos según la longitud asignada a cada campo.

Multiline

Esta propiedad se utiliza para indicar si el control de aceptar varias líneas de texto. Cuando se asigna el valor True a esta propiedad se debe indicar las barras de desplazamiento que debe tener el control. Para ello debe utilizar su propiedad ScrollBars.

PasswordChar

Se utiliza cuando queremos evitar que se visualicen los caracteres que escribimos en el control TextBox, como por ejemplo cuando ingresamos contraseñas. Acepta solo un carácter, el cual reemplazara a todo lo que se escribe en el control. Ejemplo:

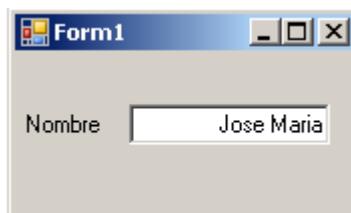


ReadOnly

Se utiliza para indicar si el usuario puede escribir o modificar los datos que se encuentran en el control TextBox. En versiones anteriores de Visual Basic, esto se hacia con la propiedad Locked.

RightToLeft

En esta propiedad se puede indicar si el texto se puede escribir de derecha a izquierda.



ScrollBars

Esta propiedad se utiliza para indicar la barra de desplazamiento que debe acompañar el control TextBox cuando esta configurado para escribir en múltiples líneas, es decir, cuando la propiedad MultiLine tiene el valor True. La opción Both muestra ambas barras: Vertical y Horizontal.



El siguiente formulario muestra una caja de textos configurada para que acepte varias líneas de texto y con la barra de desplazamiento Vertical.



TabIndex

Esta propiedad se utiliza para indicar el orden de ubicación del cursor en los controles TextBox cada vez que se pulse la tecla Tab.

TabStop

Esta propiedad se utiliza para indicar si el cursor se debe ubicar en el control cuando el usuario pulse la tecla Tab en el control anterior.

Text

En esta propiedad se almacena el texto que el usuario escribe en el control TextBox.

TextAlign

Esta propiedad permite alinear un texto en el control TextBox. Tiene las siguientes opciones:



UseWaitCursor

Esta propiedad permite indicar si se cambia la propiedad Cursor del control al valor WaitCurose. Ejemplo:



Visible

Esta propiedad permite indicar si el control se debe visualizar o no cuando se ejecuta la aplicación.

WordWrap

Se utiliza para indicar si al llenarse una línea, el cursor debe pasar automáticamente a la siguiente, cuando el control esté configurado en múltiples líneas.

Aplicación Desarrollada Nº II-04

Este programa permite ingresar un nombre y mostrar un saludo.

Johan Guerreros Montoya



Controles Utilizados



Si al hacer clic en el botón Saludo no se ha ingresado el nombre, se visualiza el mensaje error:
Por favor, Ingrese el Nombre.



Instrucciones del botón BtnSaludo:

'Pregunta si se ha ingresado el nombre

If TxtNombre.Text.Trim = ""Then

'Muestra el mensaje de error

```

MessageBox.Show("Ingrese el nombre", "Por Favor")

'Ubica el cursor en el control TxtNombre

TxtNombre.Focus()

Exit Sub

End If

'Define la variable nombre y le asigna el valor ingresado

Dim Nombre As String = TxtNombre.Text

'Alinea el texto del aludo al centro de la etiqueta

LblSaludo.TextAlign = ContentAlignment.MiddleCenter

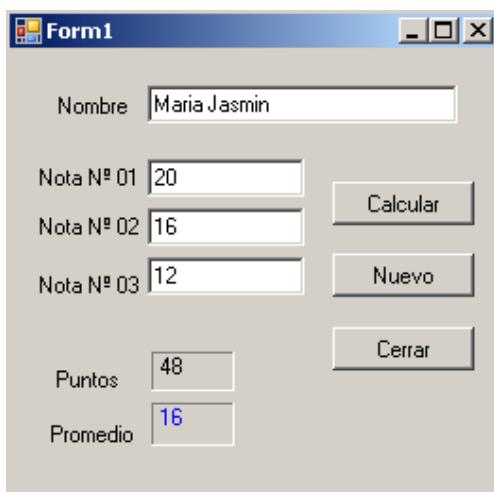
'Muestra el saludo

LblSaludo.Text = "Hola: " & Chr(13) & Nombre & Chr(13) & Chr(13) & "Bienvenido a Visual
Basic 2008 Express"

```

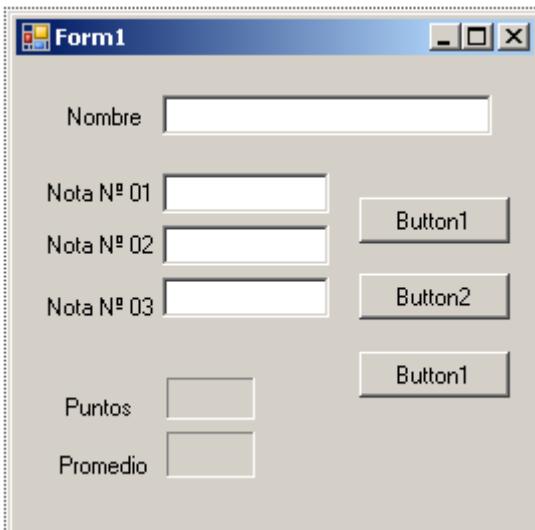
Aplicación Desarrollada Nº II-05

Permite ingresar el nombre y tres notas y mostrar los puntos y el promedio. Si esta aprobado, su promedio se muestra en azul, de lo contrario en rojo.



Controles utilizados

Johan Guerreros Montoya



A los controles LblPuntos Y LlblPromedio asígneles en su propiedad Autosize el valor False.

Instrucciones del Botón BtnCalcular:

```
Dim nota1, nota2, nota3, puntos, promedio As Single
nota1 = Single.Parse(txtnota1.Text)
nota2 = Single.Parse(txtnota2.Text)
nota3 = Single.Parse(txtnota3.Text)
puntos = nota1 + nota2 + nota3
promedio = puntos / 3
lblpuntos.Text = puntos.ToString
lblpromedio.Text = promedio.ToString
If promedio >= 10.5 Then
    lblpromedio.ForeColor = Color.Blue
Else
    lblpromedio.ForeColor = Color.Red
End If
```

Instrucciones del Botón BtnNuevo:

```
txtnombre.Text = ""
txtnota1.Text = ""
txtnota2.Text = ""
txtnota3.Text = ""
lblpuntos.Text = ""
lblpromedio.Text = ""
txtnombre.Focus()
```

Instrucciones del Botón BtnCerrar:

```
Close()
```

Podemos modificar las instrucciones del botón calcular para que verifique el ingreso correcto de las notas.

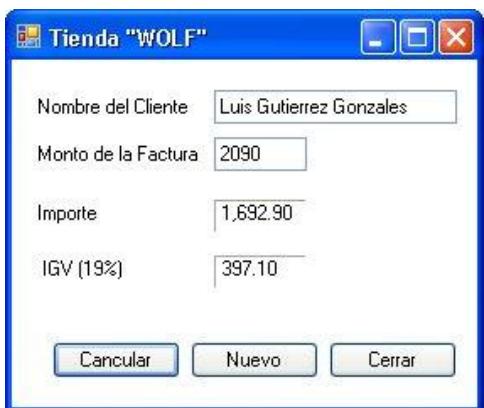
```

If txtnombre.Text.Trim = "" Then
    MessageBox.Show("Ingrese el Nombre del Alumno", "Por Favor",
    MessageBoxButtons.OK, MessageBoxIcon.Information)
    txtnombre.Focus()
    ExitSub
EndIf
If txtnota1.Text.Trim = "" OrSingle.Parse(txtnota1.Text.Trim) < 0
OrSingle.Parse(txtnota1.Text.Trim) > 20 Then
    MessageBox.Show("Ingrese la Nota 1", "Entre 0 y 20",
    MessageBoxButtons.OK, MessageBoxIcon.Error)
    txtnota1.Text = ""
    txtnota1.Focus()
    ExitSub
EndIf
If txtnota2.Text.Trim = "" OrSingle.Parse(txtnota2.Text.Trim) < 0
OrSingle.Parse(txtnota2.Text.Trim) > 20 Then
    MessageBox.Show("Ingrese la Nota 2", "Entre 0 y 20",
    MessageBoxButtons.OK, MessageBoxIcon.Error)
    txtnota2.Text = ""
    txtnota2.Focus()
    ExitSub
EndIf
If txtnota3.Text.Trim = "" OrSingle.Parse(txtnota3.Text.Trim) < 0
OrSingle.Parse(txtnota3.Text.Trim) > 20 Then
    MessageBox.Show("Ingrese la Nota 3", "Entre 0 y 20",
    MessageBoxButtons.OK, MessageBoxIcon.Error)
    txtnota3.Text = ""
    txtnota3.Focus()
    ExitSub
EndIf
Dim nota1, nota2, nota3, puntos, promedio AsSingle
    nota1 = Single.Parse(txtnota1.Text)
    nota2 = Single.Parse(txtnota1.Text)
    nota3 = Single.Parse(txtnota1.Text)
    puntos = nota1 + nota2 + nota3
    promedio = puntos / 3
    lblpuntos.Text = puntos.ToString
    lblpromedio.Text = promedio.ToString
If promedio >= 10.5 Then
    lblpromedio.ForeColor = Color.Blue
Else
    lblpromedio.ForeColor = Color.Red
EndIf

```

APLICACIÓN DESARROLLADA Nº II-06

Este programa permite ingresar el nombre de un cliente y el monto total de su factura y calcula y muestra por separado importe e IGV (19%). La suma del importe e IGV debe dar el monto de la factura.



Controles utilizados



Instrucciones del botón btncalcular:

```
Dim monto, importe, igv AsDouble
monto = Double.Parse(txtmonto.Text)
igv = monto * 0.19
    importe = monto - igv
    lblimporte.Text = importe.ToString("###,##0.00")
    lbligv.Text = igv.ToString("###,##0.00")
```

Instrucciones del botón btnnuevo:

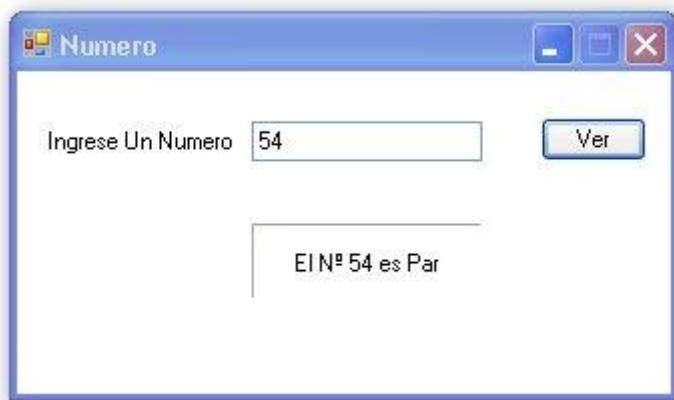
```
txtcliente.Text = ""
txtmonto.Text = ""
lblimporte.Text = ""
lbligv.Text = ""
txtcliente.Focus()
```

Instrucciones del botón btncerrar

End

APLICACIÓN DESARROLLADA Nº II-07

Este programa permite ingresar un número e indicar si es par o impar. Contiene otra forma de borrar el contenido de una caja de textos y de convertir valores.



En los programas anteriores se utiliza parse para convertir los valores ingresados en controles textbox al tipo de dato que se ha definido la variable donde se almacenan.

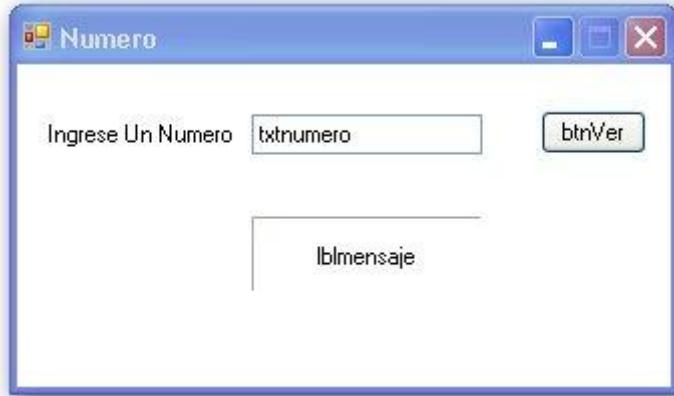
En este programa se utiliza la función ctype que también permite realizar conversiones de datos.

Para limpiar el contenido de un control textbox se le puede asignar un valor vacío entre comillas. Por ejemplo:

```
Txtdato.text = ""
```

En este programa se utiliza la función clear de los controles textbox que permiten limpiar su contenido.

Controles utilizados



Instrucciones del botón btnver:

```
If txtnumero.Text.Trim = "" Then
    MessageBox.Show("Ingrese un Numero", "Por Favor",
    MessageBoxButtons.OK, MessageBoxIcon.Error)
    txtnumero.Clear()
    txtnumero.Focus()
ExitSub
EndIf
Dim Numero As Integer
Numero = CType(txtnumero.Text, Integer)

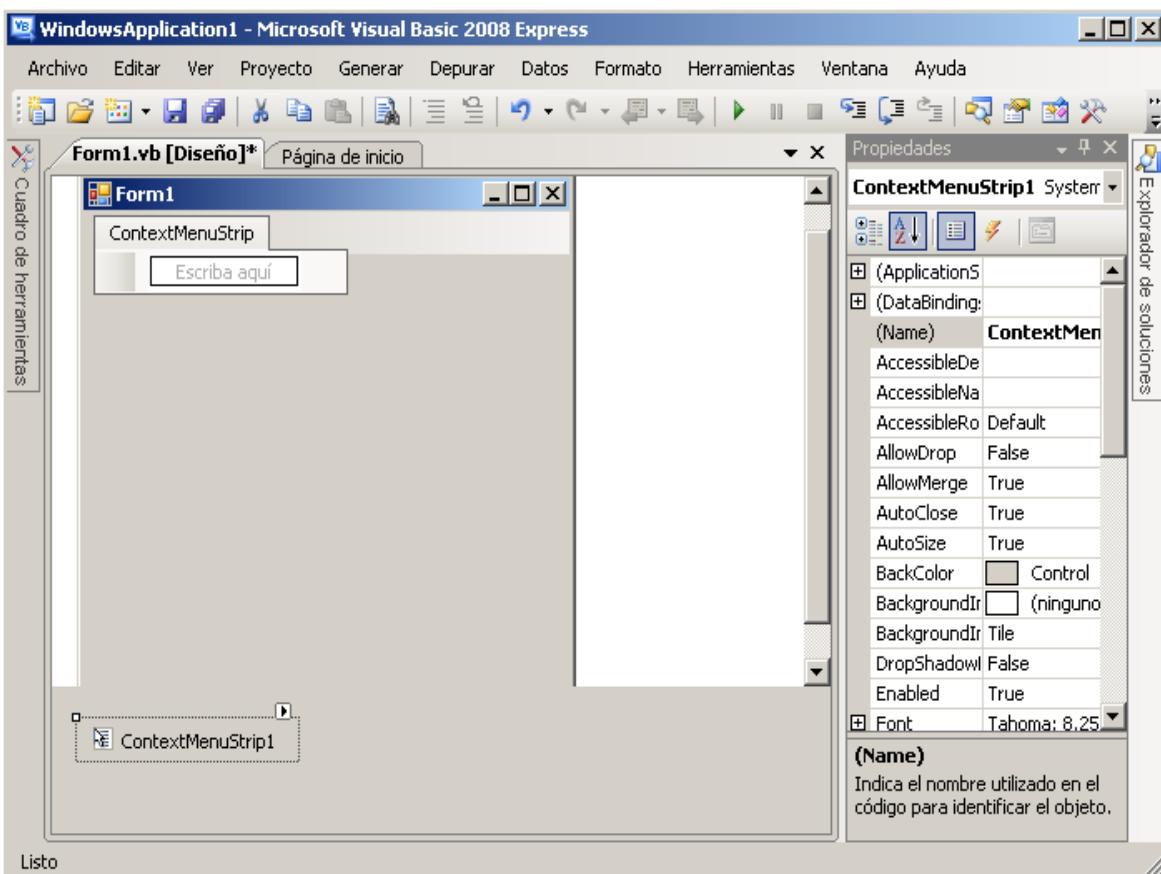
If (Numero Mod 2) = 0 Then
    lblmensaje.Text = "El N° " & Numero & " es Par"
Else
    lblmensaje.Text = "El N° " & Numero & " es Impar"
EndIf
```

EL CONTROL CONTEXTMENUSTRIP



Permite crear un menu contextual en un formulario, el cual se podra enlazar a cualquier control para que se visualiza cuando se hace clic derecho.

Cuando se dibuja en el formulario o se selecciona, se muestra la ventana para escribir las opciones que debe mostrar el menu contextual. Ubique el cursor en cada caja donde dice: Escriba aquí y escriba las opciones:



Las propiedades que se visualizan dependen si se ha seleccionado el control ContextMenuStrip o una de las opciones del menu.

Las principales propiedades cuando se selecciona el control son:

Name

Esta propiedad permite asignarle un nombre al control.

AutoClose

Esta propiedad permite indicar si el menu contextual se debe cerrar automaticamente despues de elegir una opción.

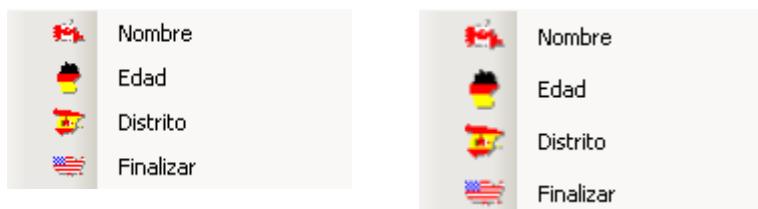
BackgroundImage

Esta propiedad permite seleccionar un grafico que servira de fondo al menu contextual.

ImageScalingSize

Esta propiedad permite indicar el tamaño de los graficos que pueden acompanar a las opciones del menu contextual. Los valores predeterminados son: 16;16.

ImageScalingSize | 16; 16

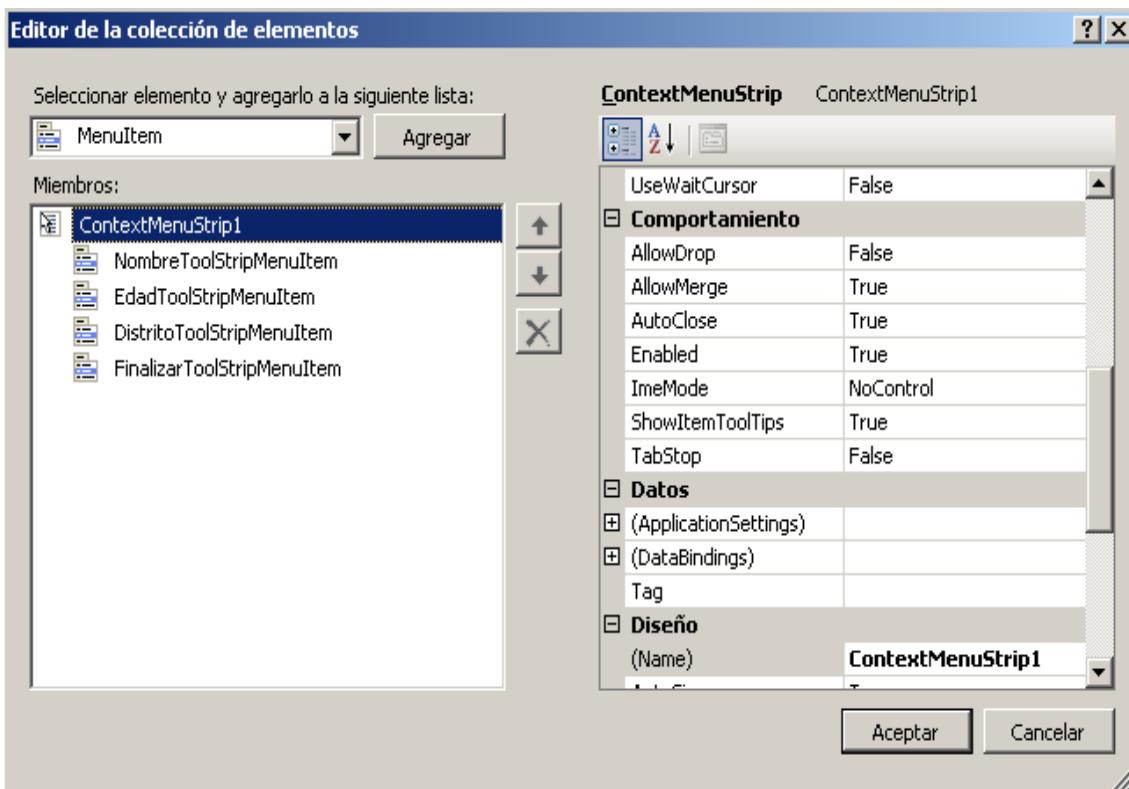


Items

Esta propiedad permite acceder a la ventana donde se puede agregar y/o editar las opciones del menu contextual.

Tambien se visualizan las propiedades de cada una de ellas.

La ventana que se visualiza al ingresar a esta propiedad es:



ShowCheckMargin

Permite indicar si se debe mostrar el margen izquierdo del menu contextual donde se muestra el check que indica si la opcion esta elegia o no.

ShowImageMargin

Permite indicar si se debe mostrar los graficos asignados a cada una de las opciones del menu contextual.

ShowItemToolTips

Permite indicar si se debe mostrar el texto de la propiedad ToolTipText.

TextDirection

Permite indicar la direccion del texto de las opciones del menu.

Las principales propiedades cuando se selecciona una opcion son:

Name

Esta propiedad permite sginarle un nombre a la opcion. En forma predeterminada toma el nombre del texto escrito para la opcion segura de: ToolStripMenuItem.

Checked



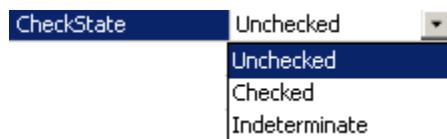
Esta propiedad permite indicar si la casilla de verificación de la opción debe estar activada o desactivada. Para que se visualice la casilla, debe asignar el valor

CheckOnClick

Esta propiedad permite indicar si la casilla de verificación de la opción debe activar y desactivar cuando se haga clic en la opción.

CheckState

Permite indicar el estado de la casilla de verificación de la opción.



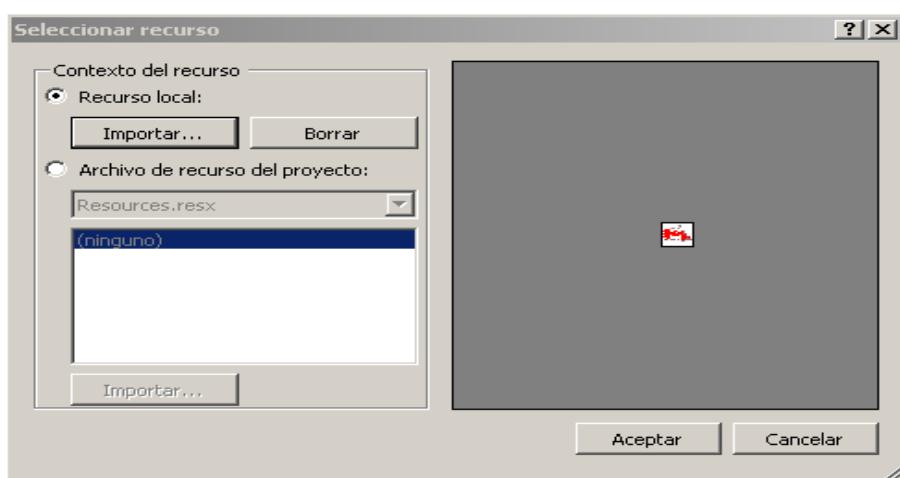
DisplayStyle

Permite indicar lo que se debe mostrar en la opción .



Image

En esta propiedad se puede seleccionar un gráfico que se mostrara junto a la opción. Se visualiza la siguiente ventana donde lo puede seleccionar.



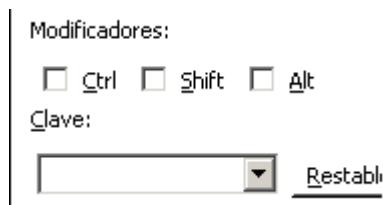
ShortCutKeyDisplayString

En esta propiedad se puede escribir un texto que reemplazara al texto que representa a la combinación de teclas de acceso rápido en la opción.

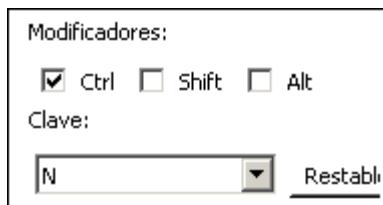
ShortCutKeys

Permite indicar la combinación de teclas de acceso rápido a la opción del menú contextual.

Al ingresar a esta opción se visualiza la siguiente ventana:



En la ventana anterior, debe activar la tecla principal y luego seleccionar en el combo la tecla secundaria. Por ejemplo, en la siguiente ventana se ha elegido CRTL + N para la opción seleccionada.



ShowShortCutKeys

Permite indicar si la combinación de teclas de acceso rápido a la opción se debe visualizar junto a la opción.

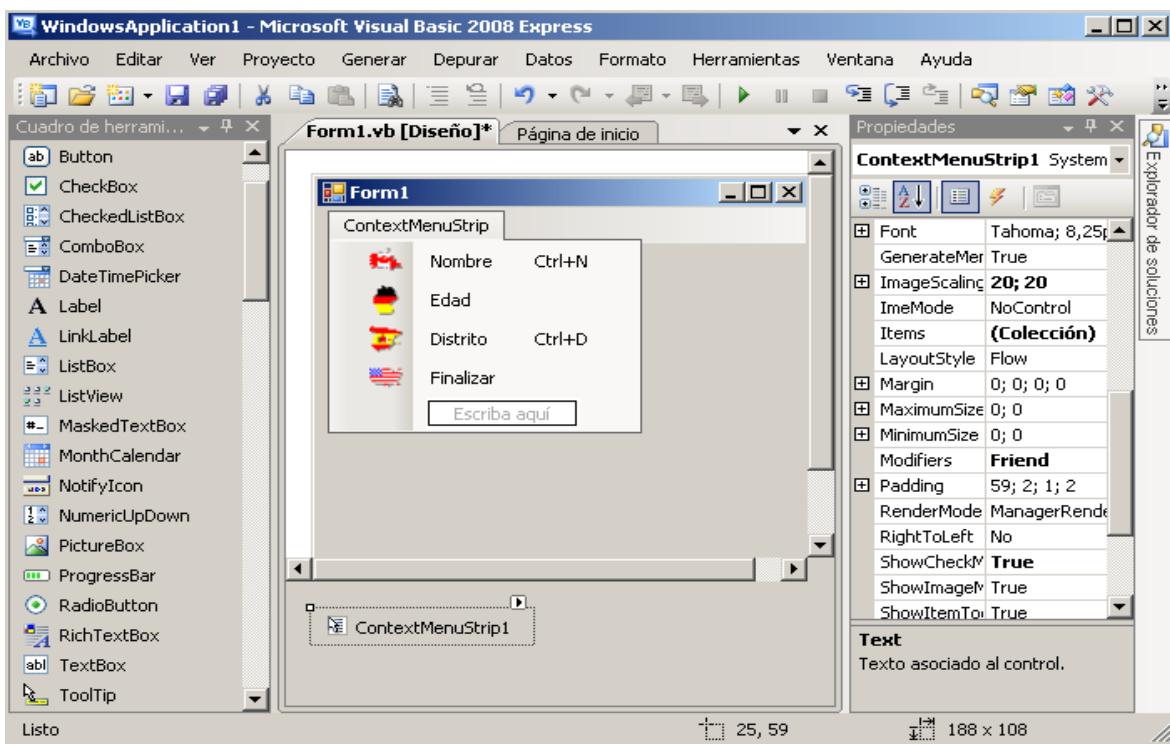
ToolTipText

Esta propiedad permite escribir un texto que se visualizara cuando se pase el puntero del mouse por una opción.

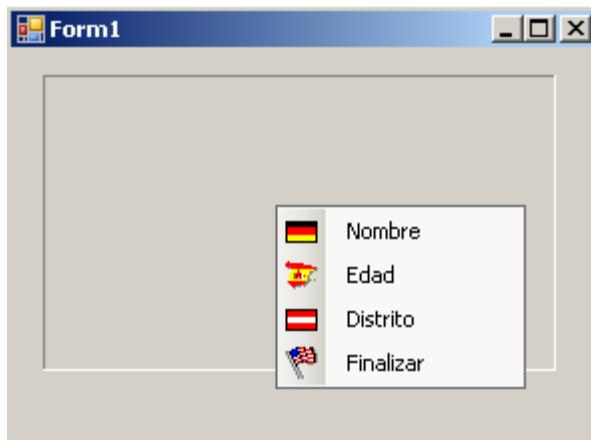
Después de crear el menú contextual, como se muestra a continuación:

Debe hacer doble clic en cada una de las opciones para escribir las instrucciones que deben ejecutar. También puede hacer clic en el botón Ver Código. En el siguiente ejemplo se ha hecho doble clic en la opción Edad.

```
Private Sub EdadToolStripMenuItem_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles EdadToolStripMenuItem.Click
    Dim edad As Integer = InputBox("Por favor ingrese su edad")
    If edad < 18 Then
        MessageBox.Show("No eres mayor de edad")
    Else
        MessageBox.Show("Eres mayor de edad")
    End If
End Sub
```

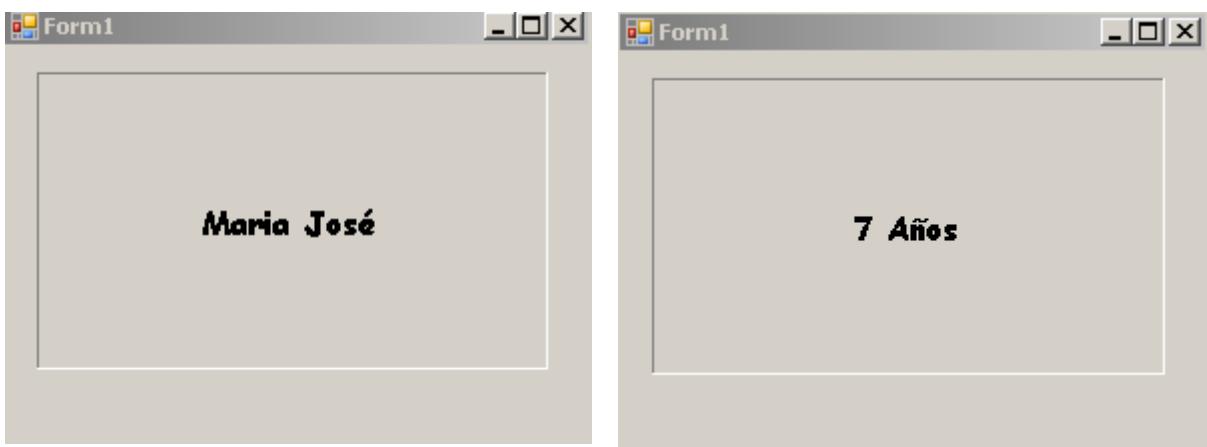


Aplicación Desarrollada Nº II-08



Este programa permite mostrar el nombre, edad o distrito de una persona utilizando un menú contextual que funciona dentro de un control Label y cual se utiliza para mostrar el dato seleccionado. El menú contextual también tiene la opción para finalizar el programa.

Ejemplo:



Para desarrollar este programa debe dibujar en su formulario un control Label llamado LblData y un control ContextMenuStrip1.



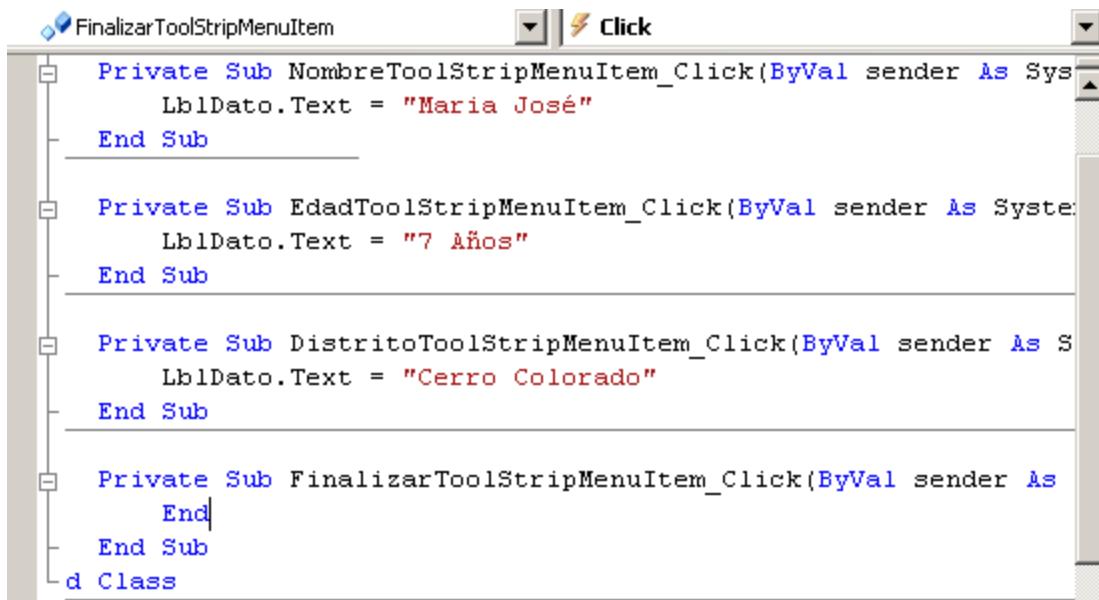
Al control Label le debe asignar valores en las siguientes propiedades:

(Name)	LblData
AutoSize	False
BorderStyle	Fixed3D
ContextMenuStrip	ContextMenuStrip1
TextAlign	MiddleCenter

En el menú contextual debe tener las siguientes opciones:



Instrucciones de la opción del menú contextual.



```

Private Sub NombreToolStripMenuItem_Click(ByVal sender As System.Object)
    LblDato.Text = "Maria José"
End Sub

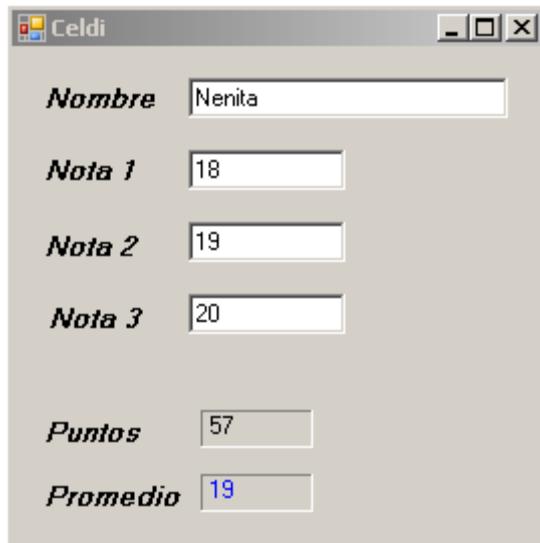
Private Sub EdadToolStripMenuItem_Click(ByVal sender As System.Object)
    LblDato.Text = "7 Años"
End Sub

Private Sub DistritoToolStripMenuItem_Click(ByVal sender As System.Object)
    LblDato.Text = "Cerro Colorado"
End Sub

Private Sub FinalizarToolStripMenuItem_Click(ByVal sender As System.Object)
    End
End Sub
End Class

```

Aplicación Desarrollada № II-09



Este programa permite ingresar el nombre y 3 notas de un alumno y mostrar los puntos obtenidos y su promedio.

Este programa no tiene ningún botón de comando, las opciones para Clacular el Promedio, Ingresar los datos de un nuevo alumno y finalizar un programa se eligen de un menú contextual que se esta enlazando al formulario, es decir, el menú contextual se muestra cuando se hace clic derecho en cualquier parte del formulario. El menú contextual es el siguiente:



Para desarrollar este programa debe de dibujar los siguientes controles incluyendo el control ContextMenuStrip para crear el menú contextual



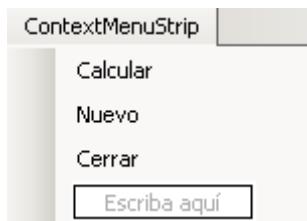
El formulario tiene como título: CETDI y lo debe enlazar con el control ContextMenuStrip1 como se muestra a continuación:

ContextMenuStrip	ContextMenuStrip1
------------------	--------------------------

Los controles LblPuntos y LblPromedio deben tener las siguientes propiedades:

AutoSize	False
BorderStyle	Fixed3D

El menú contextual debe tener las siguientes opciones:



Instrucciones de la opción Calcular:

```

Dim promedio AsSingle
    LblPuntos.Text = Val(TxtN1.Text) + Val(TxtN2.Text) + Val(TxtN3.Text)
    promedio = LblPuntos.Text / 3
If promedio >= 10.5 Then
    LblPromedio.ForeColor = Color.Blue
Else
    LblPromedio.ForeColor = Color.Red
EndIf
LblPromedio.Text = promedio
EndSub

```

Instrucciones de la opción Nuevo:

```

TxtNombre.Text = ""
    TxtN1.Text = ""
    TxtN2.Text = ""
    TxtN3.Text = ""
    LblPromedio.Text = ""
    LblPuntos.Text = ""
    TxtNombre.Focus()

```

Instrucciones de la opción Cerrar:

```
End
```

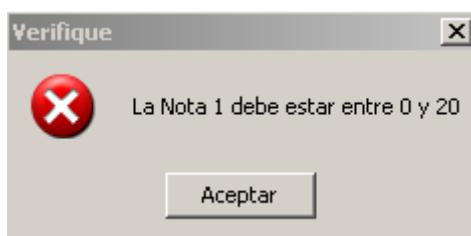
Aplicación Desarrollada Nº II-10

Este programa permite ingresar los mismos datos y mostrar los mismos resultados del programa anterior.

Nombre	Luz
Nota 1	12
Nota 2	13
Nota 3	16
Puntos	41
Promedio	13,66667

Se diferencian porque tiene las siguientes características:

1. *Los resultados se calculan y muestran en forma automáticas cuando se han ingresado el nombre del alumno y sus tres notas.*
2. *Las notas deben ser números entre 0 y 20 de los contrario se muestra el*



siguiente mensaje de error:

3. *El cursor pasa a la siguiente caja de textos al pulsar la tecla Enter.*
4. *Al pulsar la tecla Enter en la nota Nº 3 los datos se limpian para ingresar los datos de un nuevo alumno.*
5. *El programa finaliza cuando se pulsa la tecla ESC.*

Los controles para este programa son los mismos que se usaron en el programa anterior a excepción del control ContextMenuStrip.



Al formulario le debe asignar el valor True en su propiedad KeyPreView.



Este valor permitirá que el formulario detecte la pulsación de la tecla ESC para que el programa finalice.

A los controles TxtNota1, TxtNota2 y TxtNota3 debe asignarle el valor 2 en su propiedad MaxLength



Este valor permitirá que solo se ingresen dos caracteres. La verificación que sean numéricos y entre 0 y 20 se realiza dentro del programa.

Los controles LblPuntos y LblPromedio deben tener las mismas propiedades del programa anterior:



Después de dibujar los controles y asignar las propiedades debe importar el espacio de nombre: Microsoft.VisualBasic para utilizar la función IsNumeric que permite saber si un valor es numérico o no. Esto se realiza en la sección de declaración del formulario:

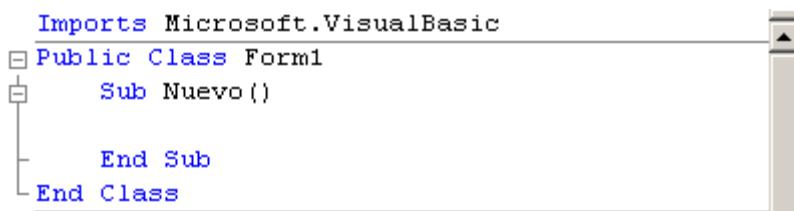
```
(General) (Declaraciones)
Imports Microsoft.VisualBasic
Public Class Form1
    Private Sub Form1_Load(ByVal sender As System.EventArgs)
        End Sub
    End Class
```

Para este programa también debe crear cuatro procedimientos llamados: Calcular, Nuevo, Limpiar y Comprobar.

Para crear un procedimiento, ubique el cursor debajo de cualquier palabra End Sub y escriba Sub y el nombre del procedimiento que desea crear.



Después de escribir Sub y el nombre del procedimiento, pulse Enter.



Las instrucciones de los procedimientos son los siguientes:

Procedimiento Nuevo: Este procedimiento limpia el contenido de los controles para ingresar los datos de un nuevo alumno.

```
Sub Nuevo()
    TxtNombre.Text = ""
    TxtN1.Text = ""
    TxtN2.Text = ""
    TxtN3.Text = ""
    LblPromedio.Text = ""
    LblPuntos.Text = ""
    TxtNombre.Focus()
EndSub
```

Procedimiento Limpiar: Este procedimiento limpia los resultados del programa es decir los puntos y el promedio del alumno. Este procedimiento es llamado cuando por ejemplo se ha ingresado en forma incorrecta una de las notas.

```
Sub Limpiar()
    LblPuntos.Text = ""
    LblPromedio.Text = ""
EndSub
```

Procedimiento Comprobar: Este procedimiento se ejecuta cuando se ingresan o modifican los datos. Si todos los datos se han ingresado, llama al procedimiento Calcular, de lo contrario llama al procedimiento Limpiar.

```
Sub Comprobar()
```

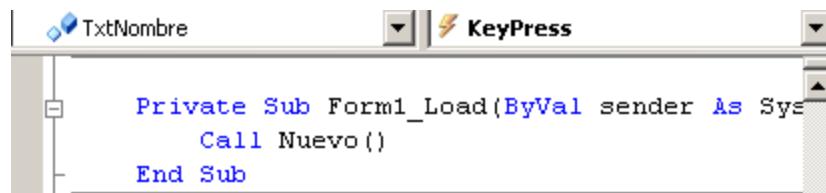
```
If TxtNombre.Text <> "" And IsNumeric(TxtN1.Text) And IsNumeric(TxtN2.Text) And
IsNumeric(TxtN3.Text) Then
Call Calcular()
Else
Call Limpiar()
EndIf
EndSub
```

Procedimiento Calcular: Este procedimiento calcula y muestra los puntos y promedio del alumno. También verifica que las notas estén entre 0 y 20.

```
Sub Calcular()
If Val(TxtN1.Text) < 0 Or Val(TxtN1.Text) > 20 Then
MsgBox("La Nota 1 debe estar entre 0 y 20", MsgBoxStyle.Critical, "Verifique")
TxtN1.Text = ""
    TxtN1.Focus()
ExitSub
EndIf
If Val(TxtN2.Text) < 0 Or Val(TxtN2.Text) > 20 Then
MsgBox("La Nota 2 debe estar entre 0 y 20", MsgBoxStyle.Critical, "Verifique")
TxtN2.Text = ""
    TxtN2.Focus()
ExitSub
EndIf
If Val(TxtN3.Text) < 0 Or Val(TxtN3.Text) > 20 Then
MsgBox("La Nota 3 debe estar entre 0 y 20", MsgBoxStyle.Critical, "Verifique")
TxtN3.Text = ""
    TxtN3.Focus()
ExitSub
EndIf
Dim promedio AsSingle
    LblPuntos.Text = Val(TxtN1.Text) + Val(TxtN2.Text) + Val(TxtN3.Text)
    promedio = (Val(TxtN1.Text) + Val(TxtN2.Text) + Val(TxtN3.Text)) / 3
If promedio >= 10.5 Then
    LblPromedio.ForeColor = Color.Blue
Else
    LblPromedio.ForeColor = Color.Red
EndIf
    LblPromedio.Text = promedio
EndSub
```

Instrucciones del evento Load del Formulario

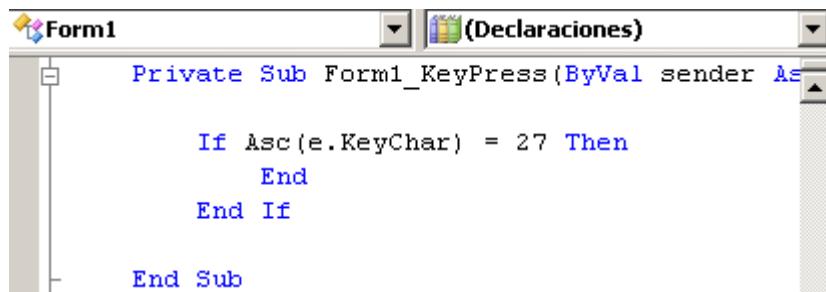
Estas instrucciones llaman al procedimiento Nuevo para limpiar el contenido de los controles.



```
TxtNombre KeyPress
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    Call Nuevo()
End Sub
```

Instrucciones del evento KeyPress del formulario

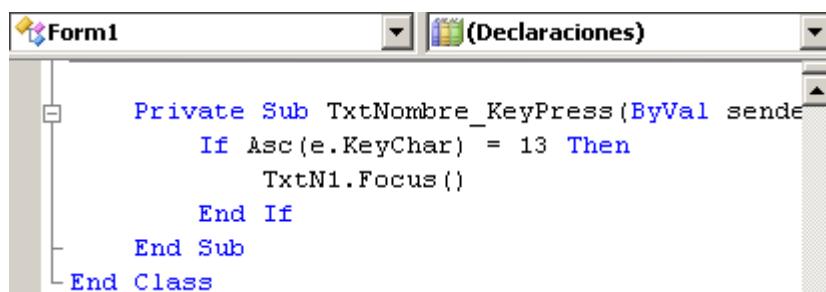
Estas instrucciones finalizan en programa cuando el usuario pulsa la tecla Esc.



```
Form1 (Declaraciones)
Private Sub Form1_KeyPress(ByVal sender As System.Object, ByVal e As System.Windows.Forms.KeyPressEventArgs)
    If Asc(e.KeyChar) = 27 Then
        End
    End If
End Sub
```

Instrucciones del evento KeyPress del control TxtNombre

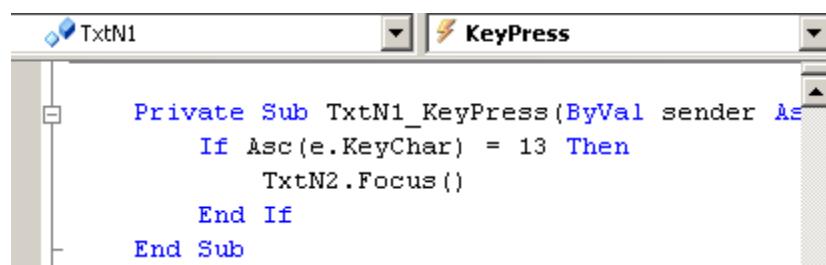
Estas instrucciones pasan el cursor al control TxtNota1 cuando el usuario pulsa la tecla Enter después de ingresar el nombre.



```
Form1 (Declaraciones)
Private Sub TxtNombre_KeyPress(ByVal sender As System.Object, ByVal e As System.Windows.Forms.KeyPressEventArgs)
    If Asc(e.KeyChar) = 13 Then
        TxtN1.Focus()
    End If
End Sub
End Class
```

Instrucciones del evento KeyPress del control TxtNota1

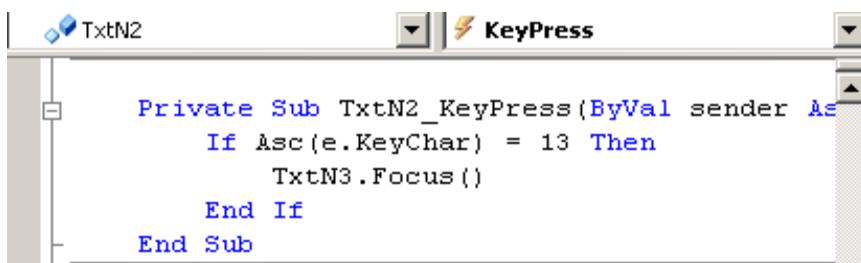
Estas instrucciones pasan el cursor al control TxtNota2 cuando el usuario pulsa la tecla Enter después de ingresar la nota Nº1.



```
TxtN1 KeyPress
Private Sub TxtN1_KeyPress(ByVal sender As System.Object, ByVal e As System.Windows.Forms.KeyPressEventArgs)
    If Asc(e.KeyChar) = 13 Then
        TxtN2.Focus()
    End If
End Sub
```

Instrucciones del evento KeyPress del control TxtNota2

Estas instrucciones pasan el cursor al control TxtNota2 cuando el usuario pulsa la tecla Enter después de ingresar la nota Nº1.



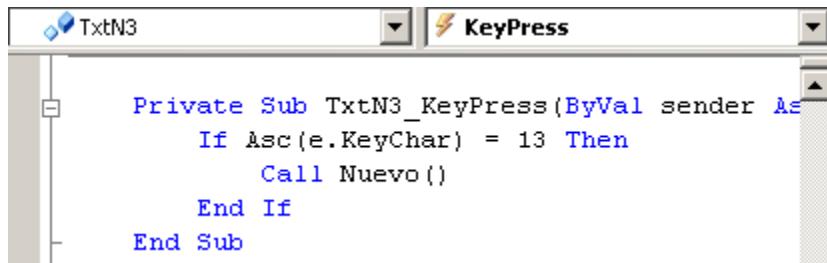
```

    Private Sub TxtN2_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs)
        If Asc(e.KeyChar) = 13 Then
            TxtN3.Focus()
        End If
    End Sub

```

Instrucción del evento KeyPress del control TxtNota3

Estas instrucciones limpian los datos ingresados cuando el usuario pulsa la tecla Enter



```

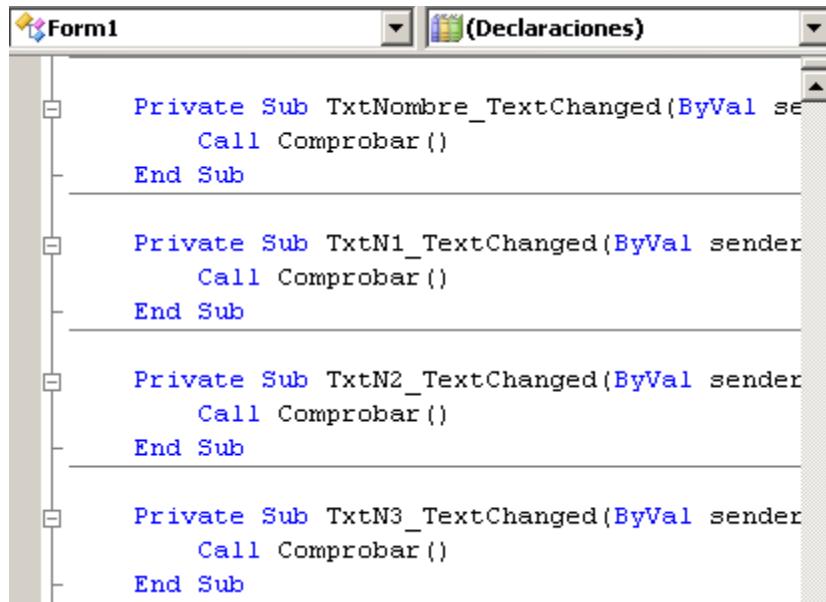
    Private Sub TxtN3_KeyPress(ByVal sender As Object, ByVal e As System.Windows.Forms.KeyPressEventArgs)
        If Asc(e.KeyChar) = 13 Then
            Call Nuevo()
        End If
    End Sub

```

después de ingresar la nota Nº3.

Instrucciones del evento TextChanged de los controles TextBox

Estas instrucciones llaman al procedimiento Comprobar que verifica si se han ingresado todos los datos para llamar al procedimiento calcular o al procedimiento Limpiar.



```

    Private Sub TxtNombre_TextChanged(ByVal sender As Object, ByVal e As System.EventArgs)
        Call Comprobar()
    End Sub

    Private Sub TxtN1_TextChanged(ByVal sender As Object, ByVal e As System.EventArgs)
        Call Comprobar()
    End Sub

    Private Sub TxtN2_TextChanged(ByVal sender As Object, ByVal e As System.EventArgs)
        Call Comprobar()
    End Sub

    Private Sub TxtN3_TextChanged(ByVal sender As Object, ByVal e As System.EventArgs)
        Call Comprobar()
    End Sub

```

Capítulo 3



FUNCIONES DEL VISUAL BASIC .NET

Contenido

En este capítulo, usted aprenderá a utilizar las funciones que ofrece el Visual Basic .Net.

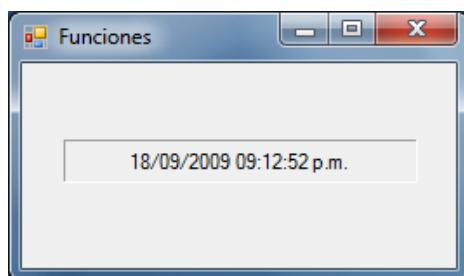
- ***Funciones tipo Fecha***
- ***Funciones tipo Cadena***
- ***Funciones Numéricas***
- ***Otras Funcione***
- ***Control de Excepciones***

FUNCIONES TIPO FECHA

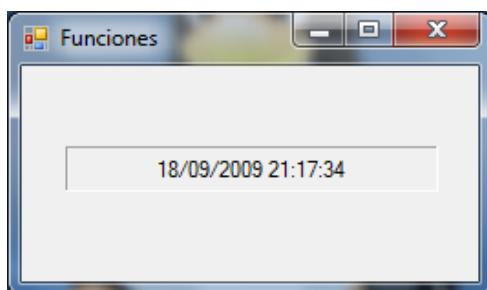
Now

Esta función devuelve la fecha y hora del sistema. La siguiente instrucción muestra la fecha y hora en el control Label 1. Se pueden escribir en el evento Load del Formulario:

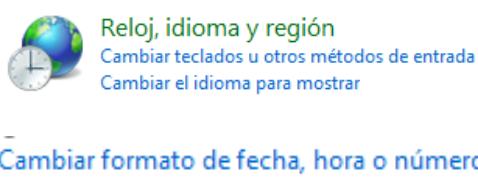
```
Label1.Text = Now
```



El formato en el que se visualiza la fecha y hora depende de la configuración que tiene la computadora:



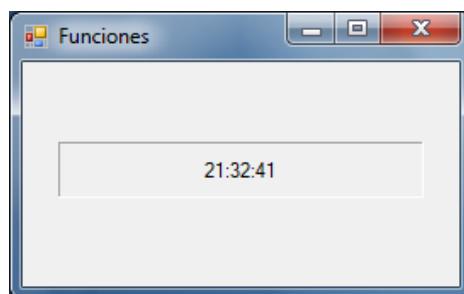
Para cambiar la configuración, ingrese al Panel de Control y elija la siguiente opción:



TimeString

Esta función devuelve sólo la hora del sistema como una cadena de caracteres. La siguiente instrucción muestra la hora en el control Label1.

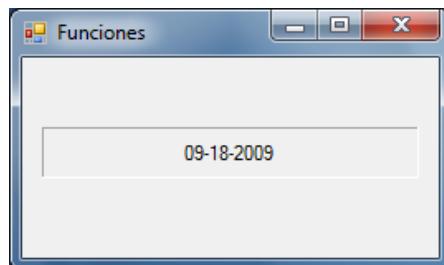
```
Label1.Text = TimeSpan
```



DateString

Esta función devuelve sólo la fecha del sistema como una cadena de caracteres. La siguiente instrucción muestra la fecha en el control Label1.

```
Label1.Text = DateString
```



ToDate

Esta función devuelve también la fecha del sistema.

Hour

Esta función devuelve sólo el número de la hora del sistema.

Minute

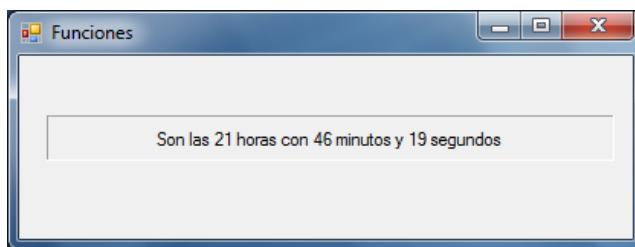
Esta función devuelve sólo los minutos de la hora del sistema.

Seconds

Esta función devuelve sólo los segundos de la hora del sistema.

La siguiente instrucción muestra en forma separada las horas, minutos y segundos de la hora del sistema:

```
Label1.Text = "Son las " & Hour(TimeString) & " horas con " & Minute(TimeString) & " minutos  
y " & Second(TimeString) & " segundos"
```



Day

Esta función devuelve sólo el número del día de la fecha del sistema.

Month

Esta función devuelve sólo el número del mes de la fecha del sistema.

MonthName

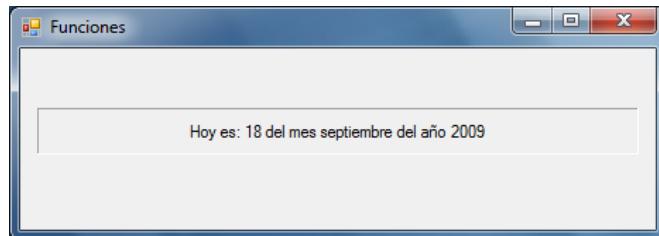
Esta función devuelve el nombre del número del mes de una fecha.

Year

Esta función devuelve sólo el año de la fecha del sistema.

Las siguientes instrucciones devuelven los datos anteriores separados:

```
Label1.Text = "Hoy es: " & Microsoft.VisualBasic.DateAndTime.Day(Now) & " del mes " &
MonthName(Month(Now)) & " del año " & Year(Now)
```



WeekDay

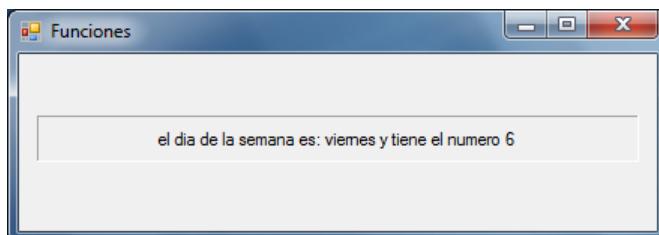
Esta función devuelve un número entre 1 y 7 que representa el número del día de la semana.

WeekDayName

Esta función devuelve el nombre del número del día de la semana.

Las siguientes instrucciones muestran en el control Label1 el número y nombre del día de la semana.

```
Label1.Text = "El día de la semana es: " & WeekdayName(Weekday(Now)) & " y tiene el
número " & Weekday(Now)
```



IsDate

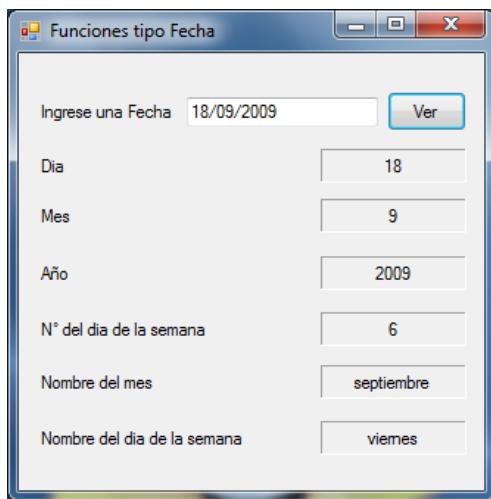
Esta función devuelve el valor True si un valor es de tipo Fecha.

Las siguientes instrucciones muestran un mensaje indicando si el contenido del control TextBox1 es de tipo fecha.

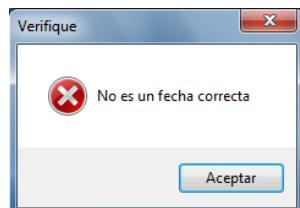
```
If IsDate(TextBox1.Text) Then
    MsgBox("Si es una fecha")
Else
    MsgBox("No es una fecha")
End If
```

Aplicación Desarrollada Nº III-01

Este programa permite ingresar una fecha y mostrar por separado toda su información. El programa verifica que se ingrese una fecha correcta.

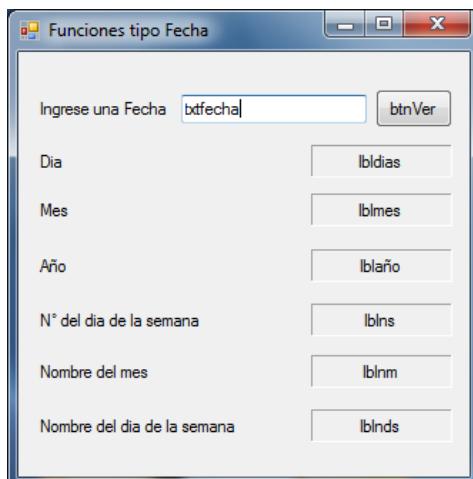


Si el usuario no ingresa correctamente una fecha en el control TextBox, se visualiza el siguiente mensaje de error:



Controles del formulario

Este formulario esta compuesto por una caja de textos llamada `TxtFecha`, un botón llamado `BtnVer` y por controles Label donde se muestran los resultados deseados.



Los controles Labels que tiene un nombre asignado y donde se visualizan los resultados tiene las siguientes propiedades:

AutoSize	False
----------	-------

BorderStyle	Fixed3D
TextAlign	MiddleCenter

Instrucciones del evento Load del formulario

```
'Muestra en forma predeterminada la fecha del sistema
TxtFecha.Text = Today()
```

Instrucciones del botón Ver

```
'Pregunta si es una fecha correcta
If IsDate(TxtFecha.Text) = True Then
    'Define una variable y almacena la fecha ingresada
    Dim Fecha As Date = Date.Parse(TxtFecha.Text)
    'Muestra la información de la fecha ingresada
    LblDía.Text = Microsoft.VisualBasic.Day(Fecha)
    LblMes.Text = Month(Fecha)
    LblAño.Text = Year(Fecha)
    LblNs.Text = Weekday(Fecha)
    LblNm.Text = MonthName(Month(Fecha))
    LblNd.Text = WeekdayName(Weekday(Fecha))
    Else
        'Muestra el mensaje si no es una fecha correcta
        MsgBox("No es una fecha correcta", MsgBoxStyle.Critical, "Verifique")
        'Limpia el contenido de la caja de textos
        TxtFecha.Clear()
        'Ubica el cursor en la caja de textos
        TxtFecha.Focus()
    End If
```

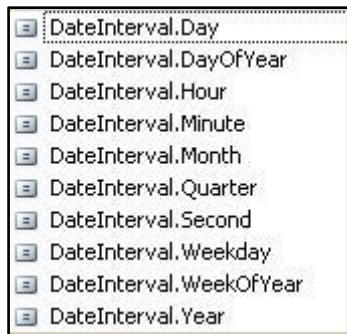
DateDiff

Esta función devuelve la diferencia entre dos fechas.

Su sintaxis es:

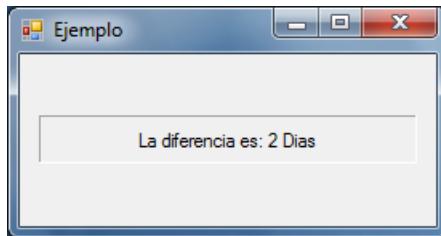
```
DateDiff(Intervalo, Fecha1, Fecha2)
```

La Fecha1 es la fecha menor y la Fecha2 es la fecha mayor. Si se intercambian las fechas el resultado es un número negativo. El intervalo es el tiempo en el cual se va a calcular la diferencia entre las dos fechas y puede tener cualquiera de los siguientes valores:



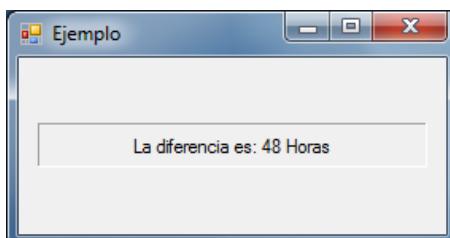
Las siguientes instrucciones muestran en un control llamado LblDías la diferencia en días entre el 01-05-2006 y el 03-05-2006.

```
Dim A,B As Date
A = Date.Parse("01-05-2006")
B = Date.Parse("03-05-2006")
LblDías.Text = "La diferencia es: " & DateDiff(DateInterval.Day, A, B) & " Dias"
```



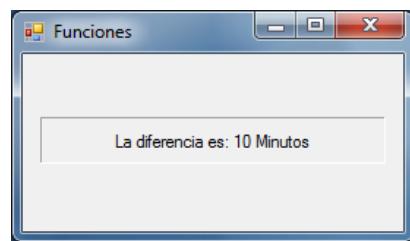
Las siguientes instrucciones muestran en un control llamado LblHoras la diferencia en horas entre el 01-05-2006.

```
Dim A, B As Date
A = Date.Parse("01-05-2006")
B = Date.Parse("02-05-2006")
LblHoras.Text = "La diferencia es: " & DateDiff(DateInterval.Hour, A, B) & " Horas"
```



Las siguientes instrucciones muestran en un control llamado LblMinutos la diferencia en minutos entre las 16:05 y las 16:20.

```
Dim A, B As Date
A = Date.Parse("16:05")
B = Date.Parse("16:20")
LblMinutos.Text = "La diferencia es: " & DateDiff(DateInterval.Minute, A, B) & " Minutos"
```



Aplicación Desarrollada Nº III-02

Este programa permite ingresar dos fechas y mostrar la diferencia de esas dos fechas en días, semanas, meses y años.

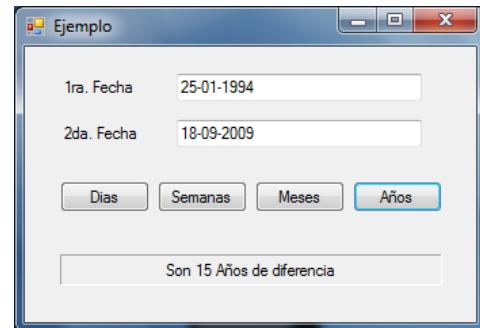
En esta ventana de ejemplo se muestra la diferencia en semanas.



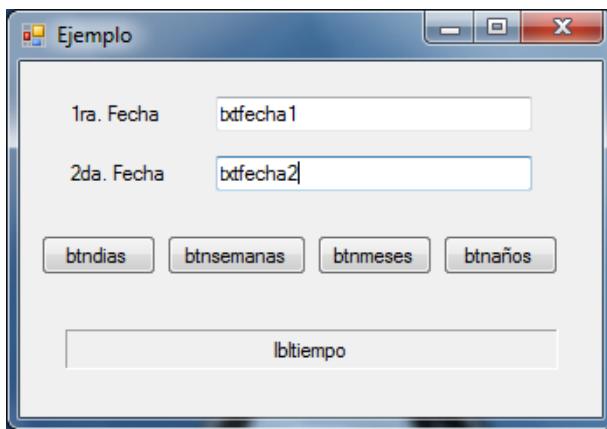
En esta ventana de ejemplo se muestra la diferencia en meses.



En esta ventana de ejemplo se muestra la diferencia en años.



Este programa está compuesto por los siguientes controles:



El control LblTiempo tiene las siguientes propiedades:

AutoSize	False
BorderStyle	Fixed3D
TextAlign	MiddleCenter

Instrucciones del botón BtnDías:

```
Dim Fecha1, Fecha2 As Date
Dim Tiempo As Integer
Fecha1 = Date.Parse(TxtFecha1.Text)
Fecha2 = Date.Parse(TxtFecha2.Text)
Tiempo = DateDiff(DateInterval.Day, Fecha1, Fecha2)
LblTiempo.Text = "Son " & Tiempo & " Días de diferencia"
```

Instrucciones del botón BtnSemanas:

```
Dim Fecha1, Fecha2 As Date
Dim Tiempo As Integer
Fecha1 = Date.Parse(TxtFecha1.Text)
Fecha2 = Date.Parse(TxtFecha2.Text)
Tiempo = DateDiff(DateInterval.Weekday, Fecha1, Fecha2)
LblTiempo.Text = "Son " & Tiempo & " Semanas de diferencia"
```

Instrucciones del botón BtnMeses:

```
Dim Fecha1, Fecha2 As Date
Dim Tiempo As Integer
Fecha1 = Date.Parse(TxtFecha1.Text)
Fecha2 = Date.Parse(TxtFecha2.Text)
Tiempo = DateDiff(DateInterval.Month, Fecha1, Fecha2)
LblTiempo.Text = "Son " & Tiempo & " Meses de diferencia"
```

Instrucciones del botón BtnAños:

```

Dim Fecha1, Fecha2 As Date
Dim Tiempo As Integer
Fecha1 = Date.Parse(TxtFecha1.Text)
Fecha2 = Date.Parse(TxtFecha2.Text)
Tiempo = DateDiff(DateInterval.Year, Fecha1, Fecha2)
LblTiempo.Text = "Son " & Tiempo & " Años de diferencia"

```

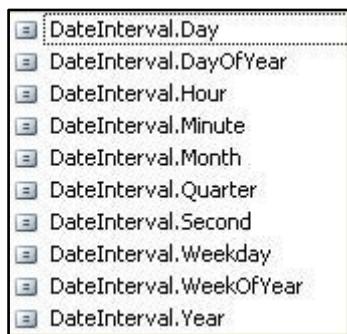
DateAdd

Esta función permite incrementar o disminuir un intervalo de tiempo a una fecha.

Su sintaxis es:

```
DateAdd(Intervalo,Numero,Fecha)
```

El número es el valor que se le va a incrementar o disminuir a la fecha. Si el valor es negativo se disminuye a la fecha. El intervalo puede ser cualquiera de los siguientes valores:

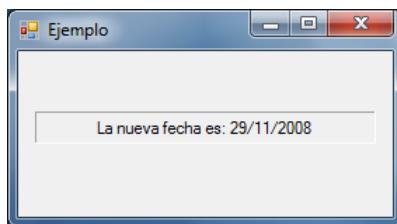


Las siguientes instrucciones incrementan 3 días a la fecha 24-08-2006:

```

Dim A As Date
A = Date.Parse("24-08-2006")
LblMinutos.Text = "La nueva fecha es: " & DateAdd(DateInterval.Day, 3, A)

```

**Aplicación Desarrollada Nº III-03**

Este programa permite ingresar una fecha y un número y muestra una nueva fecha incrementada o disminuida en días, semanas, meses y años.

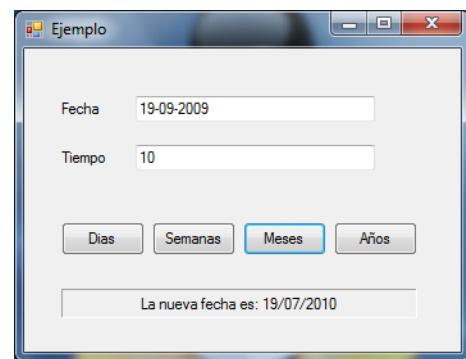
En esta ventana de ejemplo se muestra la incrementada en días.



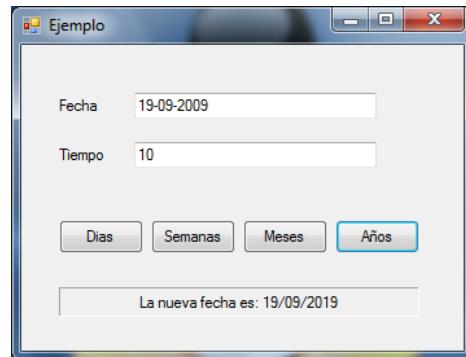
En esta ventana de ejemplo se muestra la incrementada en semanas.

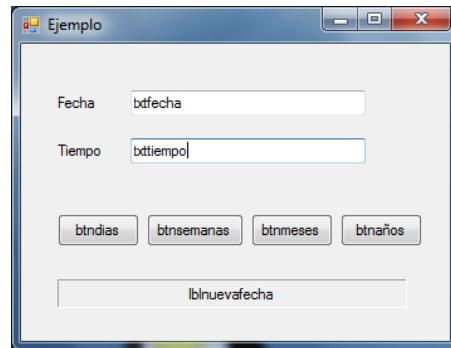


En esta ventana de ejemplo se muestra la incrementada en meses.



En esta ventana de ejemplo se muestra la incrementada en años.





Controles del formulario:

```

Dim Fecha, NuevaFecha As Date
Dim Tiempo As Integer
Fecha = Date.Parse(TxtFecha.Text)
Tiempo = Integer.Parse(TxtTiempo.Text)
NuevaFecha = DateAdd(DateInterval.Day, Tiempo, Fecha)
LblNuevaFecha.Text = "La nueva fecha es: " & NuevaFecha

```

Instrucciones del botón BtnDías:

```

Dim Fecha, NuevaFecha As Date
Dim Tiempo As Integer
Fecha = Date.Parse(TxtFecha.Text)
Tiempo = Integer.Parse(TxtTiempo.Text)
NuevaFecha = DateAdd(DateInterval.WeekOfYear, Tiempo, Fecha)
LblNuevaFecha.Text = "La nueva fecha es: " & NuevaFecha

```

Instrucciones del botón BtnMeses:

```

Dim Fecha, NuevaFecha As Date
Dim Tiempo As Integer
Fecha = Date.Parse(TxtFecha.Text)
Tiempo = Integer.Parse(TxtTiempo.Text)
NuevaFecha = DateAdd(DateInterval.Month, Tiempo, Fecha)
LblNuevaFecha.Text = "La nueva fecha es: " & NuevaFecha

```

Instrucciones del botón BtnAños:

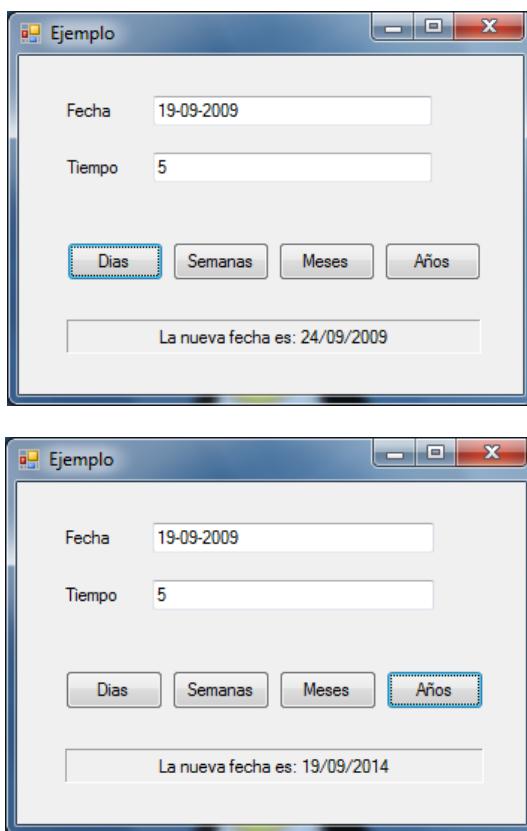
```

Dim Fecha, NuevaFecha As Date
Dim Tiempo As Integer
Fecha = Date.Parse(TxtFecha.Text)
Tiempo = Integer.Parse(TxtTiempo.Text)
NuevaFecha = DateAdd(DateInterval.Year, Tiempo, Fecha)
LblNuevaFecha.Text = "La nueva fecha es: " & NuevaFecha

```

Aplicación Desarrollada Nº III-03B

Este programa es similar al anterior, se diferencia porque está desarrollado con una función.



La función que utiliza este programa se llama NuevaFecha la cual sólo tiene un parámetro que indica un intervalo de tiempo que se debe incrementar a la fecha, y puede tener los siguientes valores:

- 1 Días
- 2 Semanas
- 3 Meses
- 4 Años

Las instrucciones de la función son las siguientes:

```
Function NuevaFecha(ByVal Intervalo As Byte) As String
```

```

Dim Nueva As Date
Dim Fecha As Date
Dim Tiempo As Integer
Fecha = Date.Parse(TxtFecha.Text)
Tiempo = Integer.Parse(TxtTiempo.Text)
Select Case Intervalo
Case 1
Nueva = DateAdd(DateInterval.Day, Tiempo, Fecha)
Case 2

```

```

Nueva = DateAdd(DateInterval.WeekOfYear, Tiempo, Fecha)
Case 3
Nueva = DateAdd(DateInterval.Month, Tiempo, Fecha)
Case 4
Nueva = DateAdd(DateInterval.Year, Tiempo, Fecha)
End Select
Return "La nueva fecha es: " & Nueva
End Function

```

Instrucciones del botón BtnDías:

```
LblNuevaFecha.Text = NuevaFecha(1)
```

Instrucciones del botón BtnSemanas:

```
LblNuevaFecha.Text = NuevaFecha(2)
```

Instrucciones del botón BtnMeses:

```
LblNuevaFecha.Text = NuevaFecha(3)
```

Instrucciones del botón BtnAños:

```
LblNuevaFecha.Text = NuevaFecha(4)
```

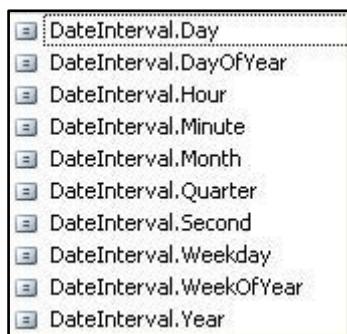
DatePart

Esta función permite obtener un dato de una fecha reemplazando a algunas funciones estudiadas anteriormente.

Su sintaxis es:

```
DatePart(Dato, Fecha)
```

El dato es cualquiera de las opciones que tienen las funciones tipo fecha:



DateInterval.Day

Obtiene el número del día del mes que puede ser entre 1 y 31 dependiendo del mes de la fecha.

Ejemplo, las siguientes instrucciones muestran el 13.

```
Dim Fecha As Date
Fecha = Date.Parse("13-07-2006")
LblResultado.Text = DatePart(DateInterval.Day, Fecha).ToString
```



DateInterval.Day

Obtiene el número del día del año que puede ser entre 1 y 365 dependiendo de la fecha.

Ejemplo: Las siguientes instrucciones muestran el número 32 porque el día 01-02-2006 es el día Nº 32 del año.

```
Dim Fecha As Date
Fecha = Date.Parse("01-02-2006")
LblResultado.Text = DatePart(DateInterval.DayOfYear, Fecha).ToString
```



DateInterval.Hour

Obtiene las horas cuando dentro de la fecha se encuentran las horas:

Ejemplo: Las siguientes instrucciones muestran el 15 porque la fecha contiene las 15 horas y 20 minutos.

```
Dim Fecha As Date
Fecha = Date.Parse("31-8-2006 15:20")
LblResultado.Text = DatePart(DateInterval.Hour, Fecha).ToString
```



DateInterval.Minute

Obtiene los minutos cuando dentro de la fecha se encuentran las horas:

Ejemplo: Las siguientes instrucciones del botón Ver muestran el 20 porque la fecha contiene las 15 horas y 20 minutos.

```
Dim Fecha As Date
Fecha = Date.Parse("31-8-2006 15:20")
LblResultado.Text = DatePart(DateInterval.Minute, Fecha).ToString
```



DateInterval.Month

Obtiene el número del mes de la fecha.

Ejemplo: Las siguientes instrucciones muestran el número 8, porque la fecha es del mes de Agosto.

```
Dim Fecha As Date
Fecha = Date.Parse("31-08-2006 15:20")
LblResultado.Text = DatePart(DateInterval.Month, Fecha).ToString
```



DateInterval.Quarter

Obtiene un número del 1 al 4 que representa el trimestre del año al que pertenece la fecha como se explica en el siguiente cuadro:

Trimestre	Rango de Fechas
1	Del 01-01 Al 31-03
2	Del 01-03 Al 30-06
3	Del 01-07 Al 30-09
4	Del 01-10 Al 31-12

Ejemplo: Las siguientes instrucciones muestran el número 3 porque la fecha 04-07-2006 pertenece al 3er. Trimestre.

```
Dim Fecha As Date
Fecha = Date.Parse("24-07-2006")
LblResultado.Text = DatePart(DateInterval.Quarter, Fecha).ToString
```



Ejemplo: Las siguientes instrucciones muestran el número 1:

```
Dim Fecha As Date
Fecha = Date.Parse("12-01-2006")
LblResultado.Text = DatePart(DateInterval.Quarter, Fecha).ToString
```



DateInterval.Second

Obtiene los segundos cuando dentro de la fecha se encuentran las horas:

Ejemplo: Las siguientes instrucciones del botón Ver muestran el 45 porque la fecha contiene las 20 horas, 13 minutos y 45 segundos.

```
Dim Fecha As Date
Fecha = Date.Parse("24-05-2006 20:13:45")
LblResultado.Text = DatePart(DateInterval.Second, Fecha).ToString
```



DateInterval.WeekDay

Obtiene el número del día de la semana que puede ser entre 1 y 7 dependiendo de la fecha. El día domingo es el Nº 1.

Ejemplo: Las siguientes instrucciones muestran el 1, porque la fecha utilizada en la función: 05-02-2006 es un día domingo.

```
Dim Fecha As Date
Fecha = Date.Parse("05-02-2006")
LblResultado.Text = DatePart(DateInterval.Weekday, Fecha).ToString
```



DateInterval.WeekOfYear

Obtiene el número de la semana del año que puede ser entre 1 y 53 dependiendo de la fecha.

Ejemplo: Las siguientes instrucciones muestran el 18, porque la fecha utilizada en la función: 01-05-2006 pertenece a la semana 18 del año.

```
Dim Fecha As Date
Fecha = Date.Parse("01-05-2006")
LblResultado.Text = DatePart(DateInterval.WeekOfYear, Fecha).ToString
```



DateInterval.Year

Obtiene el año de la fecha.

Ejemplo: Las siguientes instrucciones muestran el año 2006.

```
Dim Fecha As Date
Fecha = Date.Parse("01-05-2006")
LblResultado.Text = DatePart(DateInterval.Year, Fecha).ToString
```



FUNCIONES TIPO CADENA

Len

Esta función devuelve la cantidad de caracteres que tiene una cadena incluyendo los espacios en blanco.

Su sintaxis es:

```
Len(Cadena)
```

Ejemplo: las siguientes instrucciones muestra el numero 10.

```
Dim Texto As String
Dim R As Integer
Texto = "Hola Mundo"
R = Len(Texto)
LblResultado.Text = R
```



Las variables tipo String tienen una propiedad llamada Length que también devuelve la cantidad de caracteres que contiene.

Ejemplo: las siguientes instrucciones muestran el mismo resultado.

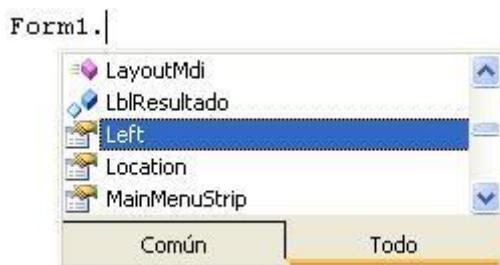
```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Texto.Length
```



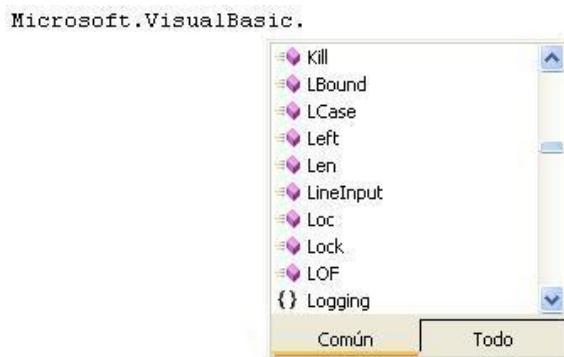
Left

Esta función devuelve una cantidad de caracteres de una cadena empezando por la izquierda.

Para evitar confusiones con propiedades Left de otros objetos, se debe utilizar su espacio de nombre: Microsoft.VisualBasic.Left. Por ejemplo, el formulario tiene una propiedad Left:



Microsoft.VisualBasic se puede utilizar para todas las funciones:



La sintaxis de la función Left es:

```
Microsoft.VisualBasic.Left(Cadena,N)
```

N es la cantidad de caracteres que se desea obtener de la cadena.

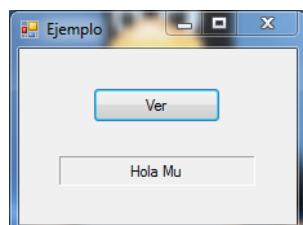
Ejemplo: las siguientes instrucciones muestra Hol.

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Microsoft.VisualBasic.Left(Texto,3)
```



Ejemplo: las siguientes instrucciones muestra Hola Mu.

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Microsoft.VisualBasic.Left(Texto,7)
```



Right

Esta función devuelve una cantidad de caracteres de una cadena empezando por la derecha.

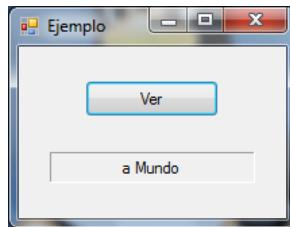
Su sintaxis es:

```
Microsoft.VisualBasic.Right(Cadena,N)
```

N es la cantidad de caracteres que se desea obtener de la cadena.

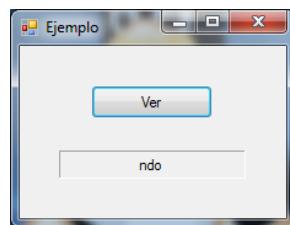
Ejemplo: las siguientes instrucciones muestran a Mundo.

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Microsoft.VisualBasic.Right(Texto,7)
```



Las siguientes instrucciones muestran ndo:

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Microsoft.VisualBasic.Right(Texto,3)
```



Mid

Esta función devuelve una cantidad de caracteres de una cadena a partir de una determinada posición.

Su sintaxis es:

`Microsoft.VisualBasic.Mid(Cadena,P,N)`

P es la posición de la cadena a partir de la cual se desean obtener los caracteres.

N es la cantidad de caracteres que se desea obtener de la cadena.

Ejemplo: las siguientes instrucciones muestran la M.



```

Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Microsoft.VisualBasic.Mid(Texto,3,4)

```

Las variables tipo String tienen un método llamado SubString que obtiene el mismo resultado de la función Mid, con la diferencia que el primer elemento es el cero (0). Se le debe dar como parámetros la posición inicial y la cantidad de caracteres. Ejemplo: las siguientes instrucciones muestran el mismo resultado anterior:

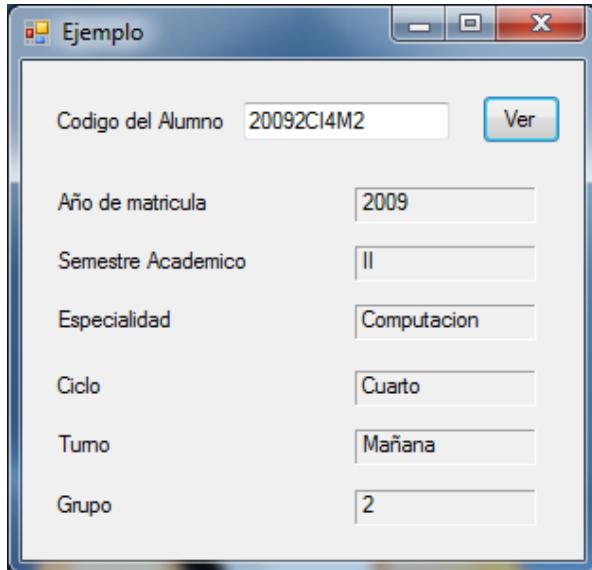
```

Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Texto.Substring(2,4)

```

Aplicación Desarrollada Nº III-04

Este programa permite ingresar el código de un alumno y mostrar los datos que representa.



El código del alumno está compuesto por 10 caracteres que representan:

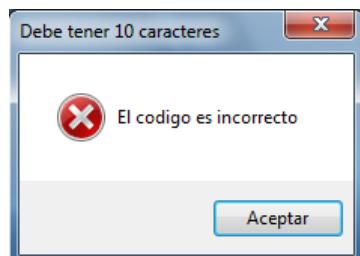
- 1. Los cuatro primeros caracteres representan el año de la matrícula.**
- 2. El quinto carácter representa el semestre y puede ser 1 ó 2.**
- 3. El sexto y séptimo carácter representa la especialidad del alumno y son las siguientes:**

CI	Computación
CO	Contabilidad
SE	Secretariado
ET	Enfermería

- 4. El octavo carácter representa el ciclo.**

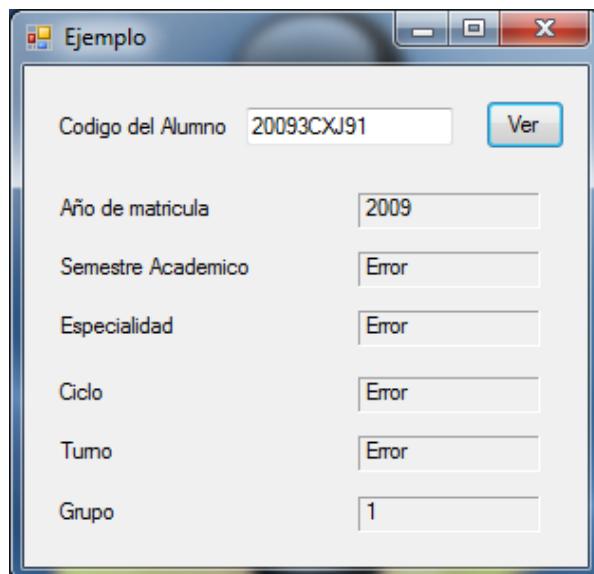
5. El noveno carácter representa el turno.**6. El decimo carácter representa el número del grupo.**

En caso de ingresar un código que no tenga 10 caracteres se visualiza el siguiente mensaje de error:



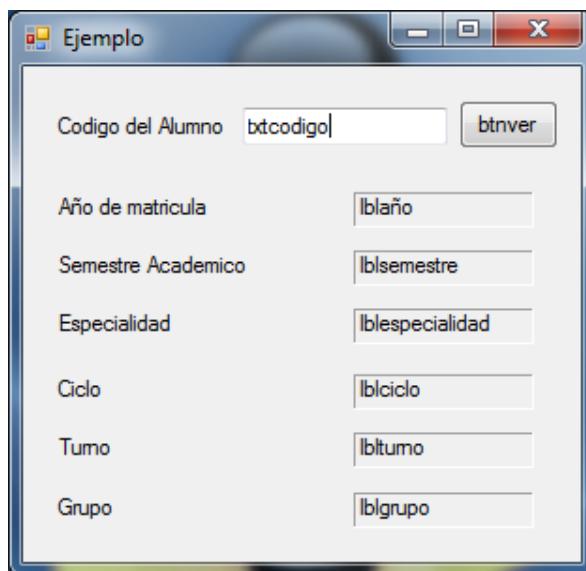
Si algún dato del código del alumno se ingresa en forma incorrecta se visualiza la palabra Error.

Ejemplo:



Codigo del Alumno	20093CXJ91	Ver
Año de matricula	2009	
Semestre Academico	Error	
Especialidad	Error	
Ciclo	Error	
Turno	Error	
Grupo	1	

Controles del formulario:



Los controles Labels que muestran los resultados tienen las siguientes propiedades:

AutoSize	<input type="checkbox"/> False
BorderStyle	<input checked="" type="checkbox"/> Fixed3D
TextAlign	<input checked="" type="checkbox"/> MiddleCenter

Instrucciones del botón Ver:

```

Dim Código, Año, Sem, Esp, Cic, Tur, Gru, Semestre, Especialidad, Ciclo, Turno As String
Código = TxtCódigo.Text
If Código.Length <> 10 Then
    MsgBox("El código es incorrecto", 16, "Debe tener 10 caracteres")
    TxtCódigo.Clear()
End If
'Lee los datos del código ingresado
Año = Microsoft.VisualBasic.Left(Código, 4)
Sem = Código.Substring(4, 1)
Esp = Código.Substring(5, 2)
Cic = Código.Substring(7, 1)
Tur = Código.Substring(8, 1)
Gru = Microsoft.VisualBasic.Right(Código, 1)
'Obtiene el Semestre Académico
Select Case Sem
    Case "1"
        Semestre = "I"
    Case "2"
        Semestre = "II"
End Select

```

```
Case Else
Semestre = "Error"
End Select
'Obtiene la especialidad
Select Case Esp
Case "CI"
Especialidad = "Computación"
Case "CO"
Especialidad = "Contabilidad"
Case "ET"
Especialidad = "Enfermería"
Case "SE"
Especialidad = "Secretariado"
Case Else
Especialidad = "Error"
End Select
'Obtiene el Ciclo
Select Case Cic
Case "1"
Ciclo = "Primero"
Case "2"
Ciclo = "Segundo"
Case "3"
Ciclo = "Tercero"
Case "4"
Ciclo = "Cuarto"
Case "5"
Ciclo = "Quinto"
Case "6"
Ciclo = "Sexto"
Case Else
Ciclo = "Error"
End Select
'Obtiene el Turno
Select Case Tur
Case "M"
Turno = "Mañana"
Case "T"
Turno = "Tarde"
Case "N"
Turno = "Noche"
Case Else
Turno = "Error"
End Select
'Muestra los resultados
```

```

LblAño.Text = Año
LblSemestre.Text = Semestre
LblEspecialidad.Text = Especialidad
LblCiclo.Text = Ciclo
LblTurno.Text = Turno
LblGrupo.Text = Gru

```

UCase

Esta función convierte un texto a mayúsculas.

Su sintaxis es:

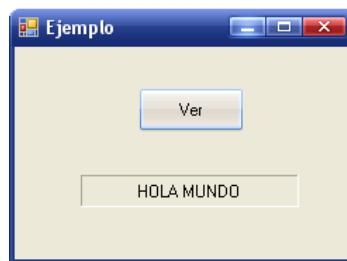
```
Microsoft.VisualBasic.UCase(Cadena)
```

Ejemplo: Las siguientes instrucciones muestran las palabras hola mundo en mayúsculas.

```

Dim Texto As String
Texto = "hola mundo"
LblResultado.Text = Microsoft.VisualBasic.UCase(Texto)

```



Si no existe ninguna propiedad o método que se llama UCase se puede escribir directamente el nombre de la función. Por ejemplo, las siguientes instrucciones muestran el mismo resultado anterior.

```

Dim Texto As String
Texto = "hola mundo"
LblResultado.Text = UCase(Texto)

```

También se puede utilizar el método ToUpper de las variables String:

```

Dim Texto As String
Texto = "hola mundo"
LblResultado.Text = Texto.ToUpper

```

LCase

Esta función convierte un texto a minúsculas.

Su sintaxis es:

```
Microsoft.VisualBasic.LCase(Cadena)
```

Ejemplo: Las siguientes instrucciones muestran las palabras HOLA MUNDO en minúsculas.

Johan Guerreros Montoya

```
Dim Texto As String
Texto = "HOLA MUNDO"
LblResultado.Text = Microsoft.VisualBasic.LCase(Texto)
```



También se puede utilizar el método ToLower de las variables String.



Las siguientes instrucciones devuelven el mismo resultado anterior.

```
Dim Texto As String
Texto = "HOLA MUNDO"
LblResultado.Text = Texto.ToLower
```

StrConv

Esta función convierte un texto a caracteres de otra región o minúsculas, mayúsculas o sólo la primera letra de cada palabra a Mayúsculas.

Su sintaxis es:

```
Microsoft.VisualBasic.StrConv(Cadena, Tipo)
```

El tipo representa la conversión que desea realizar al texto y se pueden utilizar las siguientes opciones:



Las opciones que se pueden utilizar para la configuración de nuestra región son:

VbStrCon.LowerCase

Convierte el texto a minúsculas.

VbStrCon.ProperCase

Convierte el texto a sólo la primera letra de cada palabra a mayúsculas.

VbStrCon.UpperCase

Convierte el texto a mayúsculas.

Aplicación Desarrollada Nº III-05

Este programa permite ingresar una frase y mostrarla en minúsculas, mayúsculas y sólo la primera letra de cada palabra en mayúsculas.





Controles del Formulario



Instrucciones del botón BtnMinusculas:

```
Dim Frase As String
Frase = TxtFrase.Text
LblResultado.Text = Frase.ToLower
```

Instrucciones del botón BtnMayusculas:

```
Dim Frase As String
Frase = TxtFrase.Text
LblResultado.Text = Frase.ToUpper
```

Instrucciones del botón BtnTitulos:

```
Dim Frase As String
Frase = TxtFrase.Text
LblResultado.Text = StrConv(Frase,VbStrConv.ProperCase)
```

Trim

Esta función quita los espacios en blanco que se encuentran a la izquierda o derecha de una cadena.

Su sintaxis es:

```
Microsoft.VisualBasic.Trim(Cadena)
```

Ejemplo: Las siguientes instrucciones muestran el valor 10, porque es la cantidad de caracteres que contiene. Al texto se le quita los espacios en blanco que existen a su izquierda y derecha.

```
Dim Texto As String
Texto = " Hola Mundo "
LblResultado.Text = Len(Microsoft.VisualBasic.Trim(Texto))
```



Las variables String tienen también método con el mismo nombre y cumple la misma función. Las siguientes instrucciones devuelven el mismo resultado anterior, pero, utilizando los métodos.

```
Dim Texto As String
Texto = " Hola Mundo "
LblResultado.Text = Texto.Trim.Length
```

Visual Basic .Net también tiene las funciones LTrim y RTrim que sólo quitan los espacios en blanco que se encuentran a la izquierda o derecha de una cadena.

GetChar

Esta función devuelve un carácter de una cadena.

Su sintaxis es:

```
Microsoft.VisualBasic.GetChar(Cadena, N)
```

Donde N es el número del carácter que se desea obtener. Las siguientes instrucciones de ejemplo devuelven la letra M:

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = GetChar(Texto, 6)
```



InStr

Esta función devuelve la posición inicial de una subcadena dentro de una cadena. La subcadena se empieza a buscar por la izquierda de la cadena.

Su sintaxis es:

```
Microsoft.VisualBasic.InsStr(Cadena, SubCadena)
```

Las siguientes instrucciones devuelven la posición Nº 3:

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = InStr(Texto, "la")
```



InStrRev

Esta función devuelve la posición inicial de una subcadena dentro de una cadena. La subcadena se empieza a buscar por la derecha de la cadena.

Su sintaxis es:

```
Microsoft.VisualBasic.InStrRev(Cadena, SubCadena,I)
```

Donde I es un valor opcional que indica la posición de la cadena a partir de la cual se desea realizar la búsqueda.

Las siguientes instrucciones de ejemplo muestra la posición número 10 porque la primera letra O está en la posición 10 empezando por la derecha.

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Microsoft.VisualBasic.InStrRev(Texto, "o", )
```



Las siguientes instrucciones de ejemplo muestra la posición número 2 porque la primera letra O está en la posición 2 empezando por la izquierda ya que se usa la función anterior InStr:

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Microsoft.VisualBasic.InStr(Texto, "o", )
```

En las dos funciones anteriores se puede indicar también en forma opcional el tipo de comparación que se desea realizar que puede ser: Binaria o Texto.

Replace

Esta función permite reemplazar una subcadena por otra dentro de una cadena.

Su sintaxis es:

```
Microsoft.VisualBasic.Replace(Cadena, SubCadena1, SubCadena2)
```

La SubCadena1 es la subcadena original y la SubCadena2 es la subcadena que contiene el texto que va a reemplazar al contenido de la SubCadena1.

Las siguientes instrucciones de ejemplo devuelven el texto: HXla MundX.

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Microsoft.VisualBasic.Replace(Texto, "o", "X")
```



Las siguientes instrucciones de ejemplo devuelven el texto: Hola_Mundo porque reemplazan los espacios en blanco por un guion bajo.

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Microsoft.VisualBasic.Replace(Texto, " ", "_")
```

Space

Esta función devuelve una cantidad de espacios en blanco.

Su sintaxis es:

`Microsoft.VisualBasic.Space(N)`

Donde N es la cantidad de espacios en blanco que se desea devolver.

Ejemplo: Las siguientes instrucciones muestran el texto Hola Mundo después de 4 espacios en blanco:

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Microsoft.VisualBasic.Space(4) & Texto
```



Las siguientes instrucciones de ejemplo muestran cada letra del texto Hola Mundo separadas por un espacio en blanco.



```

Dim Texto, N As String
Dim I As Integer
Texto = "Hola Mundo"
N = ""
For I = 1 To Texto.Length
    N = N & GetChar(Texto, I) & Space(1)
Next
LblResultado.Text = N.Trim

```

Str

Esta función convierte un valor a tipo String

Su sintaxis es:

```
Microsoft.VisualBasic.Str(Valor)
```

Las siguientes instrucciones convierten a tipo String dos números y los concatenan con el operador + devolviendo el número: 1025. También se utiliza el método Trim para quitar los espacios en blanco.

```

Dim A, B As Integer
A = 10
B = 25
LblResultado.Text = Str(A).Trim + Str(B).Trim

```



Si uno de los valores no se convierte a String, Visual Basic .Net realiza la operación aritmética de suma.

```

Dim A, B As Integer
A = 10
B = 25
LblResultado.Text = A + Str(B).Trim

```



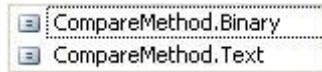
StrComp

Esta función convierte un valor a tipo String.

Su sintaxis es:

```
Microsoft.VisualBasic.StrComp(Cadena1,Cadena2,Tipo)
```

Tipo es opcional y se utiliza para indicar el tipo de comparación de las cadenas que puede ser Binary o Text.



El tipo de comparación Binary compara cada carácter por su valor binario que lo representa. La característica principal de este tipo de comparación es que las letras mayúsculas son diferentes a las minúsculas.

Esta función devuelve cualquiera de los siguientes valores como resultado de la comparación:

- 1 Si la cadena1 es menor que la cadena2.
- 1 Si la cadena1 es menor que la cadena2.
- 0 Si las dos cadenas son iguales.

Las siguientes instrucciones de ejemplo muestran el mensaje: Si son iguales:

```
Dim A, B As String
A = "Hola"
B = "hola"
If StrComp(A, B, CompareMethod.Text) = 0 Then
    LblResultado.Text = "Si son iguales"
Else
    LblResultado.Text = "No son iguales"
End If
```



StrReverse

Esta función invierte el orden de los caracteres de una cadena.

Su sintaxis es:

```
Microsoft.VisualBasic.StrReverse(Cadena)
```

Ejemplo: Las siguientes instrucciones muestran las palabras Hola Mundo en el orden inverso.

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Microsoft.VisualBasic.StrReverse(Texto)
```



Remove

Es un método de las variables String que permiten eliminar una subcadena de una cadena. Se le debe indicar la posición inicial y la cantidad de caracteres que se desea eliminar. La primera posición es cero (0).

Las siguientes instrucciones sólo muestran: Hola do.

```
Dim Texto As String
Texto = "Hola Mundo"
LblResultado.Text = Texto.Remove(5, 3)
```



Asc

Esta función devuelve un valor que representa a un carácter que se envía como parámetro.

Su sintaxis es:

```
Microsoft.VisualBasic.Asc(Caracter)
```

Por ejemplo, la siguiente instrucción muestra el número 209 que representa a la letra Ñ.

```
LblResultado.Text = Microsoft.VisualBasic.Asc("Ñ")
```



Chr

Esta función devuelve el carácter de un valor que se envía como parámetro.

Su sintaxis es:

```
Microsoft.VisualBasic.Chr(Caracter)
```

Por ejemplo, la siguiente instrucción muestra el carácter Ñ representado por el número 209.

```
LblResultado.Text = Microsoft.VisualBasic.Chr("209")
```



Aplicación Desarrollada Nº III-06

Este programa permite ingresar un carácter y mostrar el valor que lo representa.



Controles del formulario:



El control TextBox1 tiene en su propiedad MaxLength el valor 1 para que sólo se ingrese un carácter.

MaxLength	1
-----------	----------

Instrucciones del botón Ver:

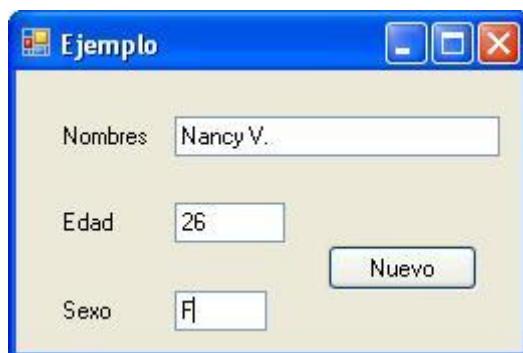
```

Dim C As Char
C = TextBox1.Text
LblResultado.Text = Microsoft.VisualBasic.Asc(C)

```

Aplicación Desarrollada Nº III-07

Este programa permite ingresar el nombre, edad y sexo de una persona realizando una consistencia de los datos que se ingresan.



La consistencia de los datos se caracteriza por lo siguiente:

- 1. En el nombre no se aceptan números.**
- 2. En la edad sólo se aceptan números.**
- 3. En el sexo sólo se aceptan las letras F o M en mayúsculas o minúsculas.**

Controles del formulario:



A los controles TextBox le debe asignar un valor en su propiedad MaxLength para limitar la cantidad de caracteres a ingresar.

TxtNombre	MaxLength	50
TxtEdad	MaxLength	2
TxtSexo	MaxLength	1

La consistencia de los datos se controla en el evento KeyPress de cada una de las cajas de texto.

Instrucciones del evento KeyPress del control TxtNombre

```
If Asc(e.KeyChar) >= 48 And Asc(e.KeyChar) <= 57 Then
    e.Handled = True
End If
```

Instrucciones del evento KeyPress del control Txt Edad

```
If (Asc(e.KeyChar) < 48 Or Asc(e.KeyChar) > 57) And Asc(e.KeyChar) <> 8 Then
    e.Handled = True
End If
```

Instrucciones del evento KeyPress del control Txt Sexo

```
If Asc(e.KeyChar) <> 102 And Asc(e.KeyChar) <> 109 And Asc(e.KeyChar) <> 70 And
    Asc(e.KeyChar) <> 77 And Asc(e.KeyChar) <> 8 Then
        e.Handled = True
    End If
```

Instrucciones del botón Nuevo

```
TxtNombre.Clear()
TxtEdad.Clear()
TxtSexo.Clear()
TxtNombre.Focus()
```

FUNCIONES NUMÉRICAS

Fix

Esta función devuelve sólo la parte entera de un número. Si el número es negativo, esta función devuelve el primer número entero negativo mayor o igual que el número.

Su sintaxis es:

```
Microsoft.VisualBasic.Fix(Numero)
```

Ejemplo: la siguiente instrucción muestra el número 12.

```
LblResultado.Text = (Microsoft.VisualBasic.Fix(12.6))
```



La siguiente instrucción muestra el número -15.

```
LblResultado.Text = (Microsoft.VisualBasic.Fix(-15.6))
```



Hex

Esta función convierte un número al sistema hexadecimal.

Su sintaxis es:

```
Microsoft.VisualBasic.Hex(Numero)
```

Ejemplo: la siguiente instrucción muestra la letra A.

```
LblResultado.Text = Microsoft.VisualBasic.Hex(10)
```



Int

Esta función devuelve sólo la parte entera de un número. Si el número es negativo, esta función devuelve el primer número entero negativo menor o igual que el número.

Su sintaxis es:

```
Microsoft.VisualBasic.Int(Numero)
```

Ejemplo: la siguiente instrucción muestra el número 23.

```
LblResultado.Text = Microsoft.VisualBasic.Int(23.5)
```



IsNumeric

Esta función permite saber si un dato es numérico. Si el dato es numérico devuelve True de lo contrario devuelve False.

Su sintaxis es:

```
Microsoft.VisualBasic.IsNumeric(Numero)
```

Las siguientes instrucciones de ejemplo muestran el mensaje: No es un número:

```
If Microsoft.VisualBasic.IsNumeric("20-89") = True Then
    LblResultado.Text = "Si es un número"
Else
    LblResultado.Text = "No es un número"
End If
```



Las siguientes instrucciones muestran el mensaje: Si es un número:

```
If Microsoft.VisualBasic.IsNumeric("2089") = True Then
    LblResultado.Text = "Si es un número"
Else
    LblResultado.Text = "No es un número"
End If
```



Oct

Esta función convierte un número al sistema octal.

Su sintaxis es:

`Microsoft.VisualBasic.Oct(Numero)`

Ejemplo: la siguiente instrucción muestra el número 10.

`LblResultado.Text = Microsoft.VisualBasic.Oct(8)`



Val

Esta función convierte un valor a tipo numérico.

Johan Guerreros Montoya

Su sintaxis es:

```
Microsoft.VisualBasic.Val(Valor)
```

Las siguientes instrucciones devuelven el número 84.

```
Dim A, B As String
A = "15"
B = "69"
LblResultado.Text = Microsoft.VisualBasic.Val(A) + Microsoft.VisualBasic.Val(B)
```



Otras funciones numéricas se encuentran en el espacio de nombre: System.Math o Math



Ejemplo:

Abs

Esta función devuelve el valor absoluto de un número.

Su sintaxis es:

```
System.Math.Abs(Numero)
```

Ejemplo: la siguiente instrucción muestra el número 12.

```
LblResultado.Text = System.Math.Abs(-12)
```



Ceil

Esta función permite redondear un número decimal al entero mayor más próximo.

Su sintaxis es:

`Math.Ceil(Numero)`

Ejemplo: la siguiente instrucción muestra el número 3.

`LblResultado.Text = Math.Ceil(2.1)`



Floor

Esta función permite redondear un número decimal al entero menor más próximo.

Su sintaxis es:

`Math.Floor(Numero)`

Ejemplo: la siguiente instrucción muestra el número 2.

`LblResultado.Text = Math.Floor(2.1)`



Max

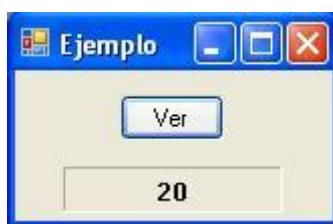
Esta función devuelve el número mayor de dos números que se pasan como parámetros.

Su sintaxis es:

`Math.Max(Numero1, Numero2)`

Ejemplo: la siguiente instrucción muestra el número 20.

`LblResultado.Text = Math.Max(15, 20)`



Min

Esta función devuelve el número menor de dos números que se pasan como parámetros.

Su sintaxis es:

`Math.Min(Numero1, Numero2)`

Ejemplo: la siguiente instrucción muestra el número 15.

`LblResultado.Text = Math.Min(15, 20)`



Pow

esta función devuelve la potencia de un número.

Su sintaxis es:

`Math.Pow(Numero, Potencia)`

Ejemplo: la siguiente instrucción muestra el número 8.

`LblResultado.Text = Math.Pow(2, 3)`



Round

Esta función permite redondear un número. Si el valor decimal es mayor a 0.5 se redondea al entero mayor más próximo.

Su sintaxis es:

`Math.Round(Numero)`

Ejemplo: la siguiente instrucción devuelve el Nº 3 y la segunda el Nº 4.

`LblResultado.Text = Math.Round(3.2)`

`LblResultado.Text = Math.Round(3.6)`



Sign

Esta función permite saber si un número es cero, positivo o negativo. Si el número es cero devuelve 0, si el número es positivo devuelve 1 y si el número es negativo devuelve -1.

Su sintaxis es:

```
Math.Sign(Numero)
```

Ejemplo: la siguiente instrucción muestra el número 1.

```
LblResultado.Text = Math.Sign(23)
```



Sqrt

Esta función permite obtener la raíz cuadrada de un número.

Su sintaxis es:

```
Math.Sqrt(Numero)
```

Ejemplo: la siguiente instrucción muestra el número 5.

```
LblResultado.Text = Math.Sqrt(25)
```



Aplicación Desarrollada Nº III-07B

Este programa permite ingresar un número y mostrar su raíz cuadrada.



Controles del formulario:



Instrucciones del botón BtnRaiz

```
Dim Raiz, Numero As Double
Numero = Double.Parse(TxtNumero.Text)
Raiz = Math.Sqrt(Numero)
LblResultado.Text = Raiz
```

Aplicación Desarrollada Nº III-08

Este programa permite ingresar dos números y muestra cual es el número mayor y cuál es el número menor.



Este programa sólo permite ingresar números en cada una de las cajas de texto.

Si los números ingresados son iguales se muestra el mensaje en una ventana como se muestra a continuación:



Si alguno de los números no se ingresa, se visualiza un mensaje de advertencia.



Controles del formulario

Los controles Labels que tiene un nombre asignado y donde se visualizan los resultados, tiene las siguientes propiedades:

AutoSize	<input type="text" value="False"/>
BorderStyle	<input type="text" value="Fixed3D"/>
TextAlign	<input type="text" value="MiddleCenter"/>

Instrucciones del evento KeyPress de TxtValor1 y TxtValor2

Estas instrucciones sólo permiten que se ingresen números en los dos controles.

```
'Sólo aceptan números
If (Asc(e.KeyChar) < 48 Or Asc(e.KeyChar) > 57) And Asc(e.KeyChar) <> 8 Then
    e.Handled = True
End If
```

Instrucciones del botón BtnResultados

```
'Pregunta si no se ha ingresado el primer valor
If TxtValor1.Text = String.Empty Then
    MsgBox("Ingrese el primer número", MsgBoxStyle.Critical, "Para mostrar los resultados")
    TxtValor1.Focus()
    Exit Sub
End If
```

```

'Pregunta si no se ha ingresado el segundo valor
If TxtValor2.Text = String.Empty Then
    MsgBox("Ingrese el segundo número", MsgBoxStyle.Critical, "Para mostrar los
resultados")
    TxtValor2.Focus()
End Sub
Exit If
'Almacena los números ingresados
Dim A, B As Integer
A = Integer.Parse(TxtValor1.Text)
B = Integer.Parse(TxtValor2.Text)
'Pregunta si los números son iguales
If A = B Then
    LblMayor.Text = ""
    LblMenor.Text = ""
    MsgBox("Los números son iguales", MsgBoxStyle.Information, "Resultados")
Else
    'Muestra el número mayor
    LblMayor.Text = Math.Max(A, B)
    'Muestra el número menor
    LblMenor.Text = Math.Min(A, B)
End If

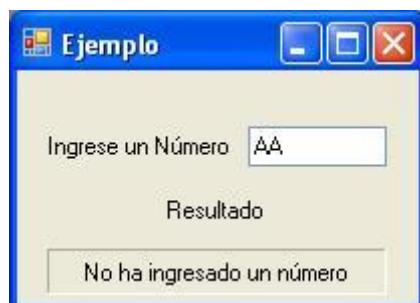
```

Aplicación Desarrollada Nº III-09

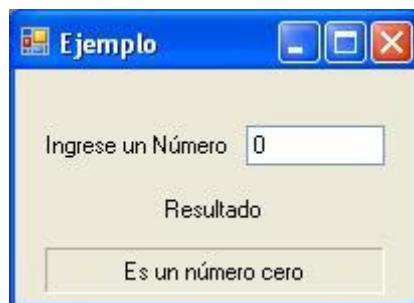
Este programa permite ingresar un número y muestra un mensaje si el número es positivo, negativo o cero.



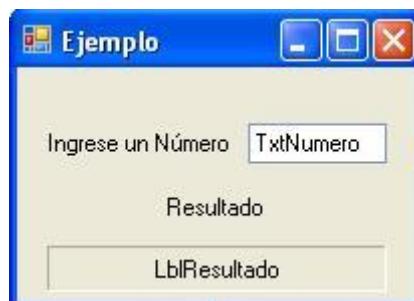
Si no se ingresa un número, se muestra un mensaje de advertencia:



El mensaje se visualiza en cuanto se digita el número en la caja de textos para ello se programa en su evento TextChanged.



Controles del formulario



Instrucciones del evento TextChanged del control TxtNumero.

```
'Pregunta si se ha ingresado un número
If IsNumeric(TxtNumero.Text) Then
    Dim Numero, Signo As Integer
    'Almacena el número ingresado
    Numero = Integer.Parse(TxtNumero.Text)
    'Obtiene el signo del número
    Signo = Math.Sign(Numero)
    'Muestra el mensaje
    Select Case Signo
        Case -1
            LblResultado.Text = "El número " & Numero & " es Negativo"
        Case 0
            LblResultado.Text = "Es un número 0"
        Case 1
            LblResultado.Text = "El número " & Numero & " es Positivo"
    End Select
    Else
        LblResultado.Text = "No ha ingresado un número"
    End If
```

Aplicación Desarrollada Nº III-09B

Este programa permite ingresar dos números y muestra el resultado del primer número elevado a la potencia del segundo:



Controles del formulario



Instrucciones del botón BtnPotencia

```

Dim Valor1, Valor2, Potencia As Integer
Valor1 = Integer.Parse(TxtValor1.Text)
Valor2 = Integer.Parse(TxtValor2.Text)
Potencia = Math.Pow(Valor1, Valor2)
'Muestra el resultado como una cadena para que se ejecute más rápido
LblResultado.Text = Potencia.ToString

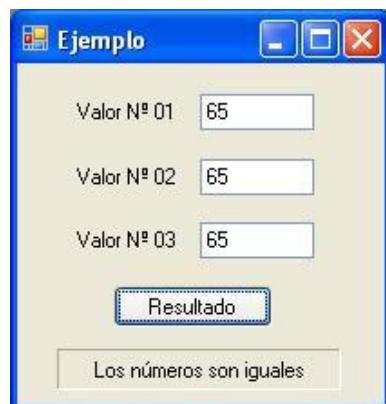
```

Aplicación Desarrollada Nº III-10

Este programa permite ingresar tres números y muestra el número mayor.



Si los tres números son iguales, se muestra el respectivo mensaje:



Controles del formulario



Instrucciones del botón BtnResultado:

```

Dim A, B, C, Mayor1, Mayor2 As Integer
'Almacena los números ingresados
A = Convert.ToInt32(TxtValor1.Text)
B = Convert.ToInt32(TxtValor2.Text)
C = Convert.ToInt32(TxtValor3.Text)
'Calcula el número mayor
Mayor1 = Math.Max(A, B)
Mayor2 = Math.Max(Mayor1, C)
'Pregunta si los tres números son iguales
If A = B And B = C Then
    LblResultado.Text = "Los números son iguales"
Else
    LblResultado.Text = "El Número mayor es: " & Mayor2
End If

```

Aplicación Desarrollada Nº III-11

Este programa permite ingresar cuatro números y muestra el número menor.

Johan Guerreros Montoya



Si los números son iguales, también se muestra el respectivo mensaje como en el programa anterior.



Controles del formulario



Instrucciones del botón BtnResultado:

```
'Verifica que se hayan ingresado correctamente los 4 números
If Not IsNumeric(TxtValor1.Text) Then
```

```
MsgBox("No ha ingresado correctamente el 1er. número", MsgBoxStyle.Critical,  
"Verifique")  
TxtValor1.Clear()  
TxtValor1.Focus()  
Exit Sub  
End If  
If Not IsNumeric(TxtValor2.Text) Then  
MsgBox("No ha ingresado correctamente el 2do. número", MsgBoxStyle.Critical,  
"Verifique")  
TxtValor2.Clear()  
TxtValor2.Focus()  
Exit Sub  
End If  
  
If Not IsNumeric(TxtValor3.Text) Then  
MsgBox("No ha ingresado correctamente el 3er. número", MsgBoxStyle.Critical,  
"Verifique")  
TxtValor3.Clear()  
TxtValor3.Focus()  
Exit Sub  
End If  
If Not IsNumeric(TxtValor4.Text) Then  
MsgBox("No ha ingresado correctamente el 4to. número", MsgBoxStyle.Critical,  
"Verifique")  
TxtValor4.Clear()  
TxtValor4.Focus()  
Exit Sub  
End If  
'Almacena los números ingresados utilizando Convert  
Dim A, B, C, D, Menor1, Menor2, Menor3 As Integer  
A = Convert.ToInt32(TxtValor1.Text)  
B = Convert.ToInt32(TxtValor2.Text)  
C = Convert.ToInt32(TxtValor3.Text)  
D = Convert.ToInt32(TxtValor4.Text)  
'Calcula el número menor  
Menor1 = Math.Min(A, B)  
Menor2 = Math.Min(Menor1, C)  
Menor3 = Math.Min(Menor2, D)  
'Pregunta si los cuatro números son iguales  
If A = B And B = C And C = D Then  
LblResultado.Text = "Los números son iguales"  
Else  
LblResultado.Text = "El Número menor es: " & Menor3  
End If
```

OTRAS FUNCIONES

Rnd

Esta función permite obtener un número al azar entre 0 y 1.

Su sintaxis es: Microsoft.VisualBasic.Rnd ()

La siguiente instrucción de ejemplo muestra un número al azar entre 0 y 1 en el control LblResultado:

```
LblResultado.text= Microsoft.VisualBasic.Rnd ()
```



Para obtener un numero al azar entre un rango de números se debe utilizar la siguiente fórmula: $\text{int}((\text{máximo}-\text{minimo}+1)*\text{Rnd})$.

Donde máximo es el numero mayor y minino es el número menor.

Ejemplo: la siguiente instrucción muestra en el control Lblresultado un numero al azar entre 15 y 20.

```
LblResultado.text= Int((15 - 20+1)*Rnd()+15)
```



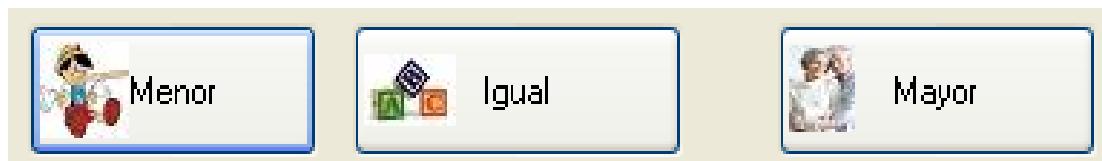
La función Randomize evita que la secuencia de números al azar se repita.

Aplicación desarrollada Nro. III-12

Este programa consiste en un juego con números que se generan al azar utilizando la función Rnd



Al hacer un clic en el botón jugar un número al azar entre 10 y 20, el cual será el número jugado y el jugador debe hacer clic en cualquiera de los tres botones:menor, igual, mayor.



Estos 3 botones muestran un número al azar entre 5 y 25. El jugador debe acertar con el número que se genera y con el botón donde hace clic.

Si el jugador hace clic en el botón menor y el número que se genera es menor que el número jugado, entonces gana la jugada. Lo mismo con el resto de botones

Ejemplo: en la siguiente jugada el jugador ha hecho clic en el botón mayor y a ganado la jugada por que el numero que se genero es 23 y este es mayor que el jugado (15).



Por cada jugada con el botón menor o mayor, el jugador recibe un punto y por cada jugada con el botón igual, el jugador obtiene tres puntos. Cada una de las jugadas que se realizan son contabilizadas.

En la parte inferior se visualiza la estadística del juego, es decir, la cantidad de jugadas realizadas y la cantidad de puntos obtenidos por el jugador.

Por ejemplo, en la siguiente ventana se indica que el jugador va realizando 12 jugadas y que ha obtenido 9 puntos:



Si desea reinicializar la estadística del juego, es decir que el número de jugadas y el numero de puntos obtenidos sea 0, puedo hacer clic en el botón que se encuentra en la parte inferior derecha:



Al hacer clic en el botón reiniciar, el juego muestra la siguiente ventana.



En esta ventana se ha reiniciado la estadística de juego, se han limpiado los mensajes y se han desactivado los botones para jugar. Solo se encuentra activo el botón que genera y muestra el número a jugar.

Al hacer clic en el botón jugar, este se desactiva y se activan los botones para realizar la jugada.



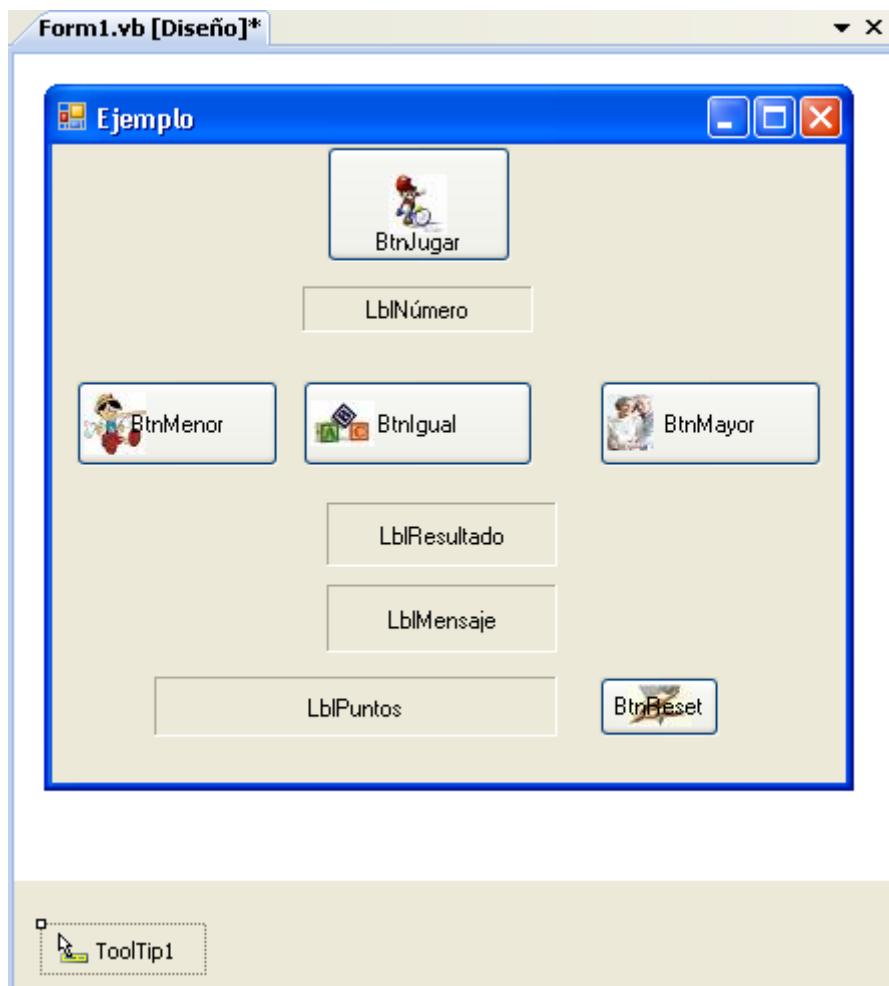
Cada uno de los botones tiene asignado un texto explicativo en su propiedad tooltip.
Ejemplo:



Todos los botones actúan como inteligentes como por ejemplo, cuando se hace clic en cualquiera de los botones menor, igual o mayor, estos se desactivan para que el jugador ya no vuelva a jugar hasta generar un nuevo número a jugar.



Controles del formulario:

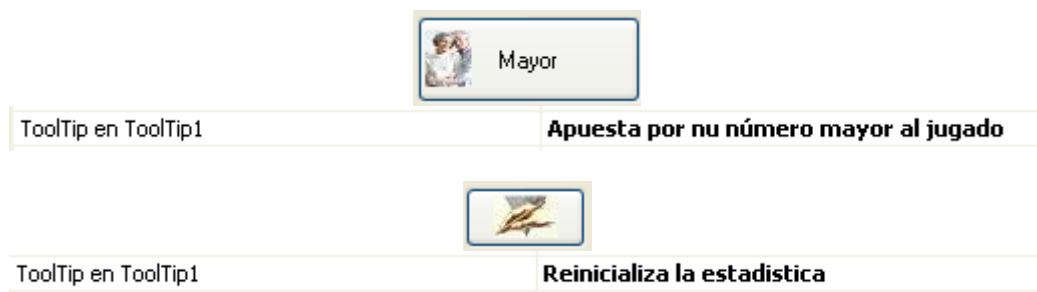


Cada uno de los botones tiene su propiedad image, un grafico asignada con una aliniacion determinada, para lo cual se utiliza su propiedad imageAlign.

ImageAlign MiddleCenter

El control tooltip1 se agrega al aplicación para asignar a cada uno de los botones un texto explicativo en su propiedad tooltip. Ejemplos:





Los controles Labels que tiene un nombre asignado donde se visualizan los resultados, tiene un tamaño de letra asignado en su propiedad Font y las siguientes propiedades:

AutoSize	False
BorderStyle	FixedSingle
TextAlign	MiddleRight

Antes de escribir las instrucciones debe de declarar de tipo Integer y a nivel de formulario las variables puntos y jugadas, las cuales se utilizan para llevar la estadística del juego.



```
Randomize()
Lblnumero.Text = String.Empty
Lblrgebnado.Text = String.Empty
Lbldatos.Text = String.Empty
lblpuntos.Text = "tiene 0 puntos en 0 jugadas"
Btnmayor.Enabled = False
Btngual.Enabled = False
Btnmenor.Enabled = False
```

```
Dim numero AsByte
numero = Int((20 - 10 + 1) * Rnd() + 10)
Lblnumero.Text = numero
Lblrgebnado.Text = String.Empty
Lbldatos.Text = String.Empty
Btnmayor.Enabled = True
Btngual.Enabled = True
Btnmenor.Enabled = True
Btnjugar.Enabled = False
```

```
Dim numero, juego AsByte
juego += 1
numero = Int((25 - 5 + 1) * Rnd() + 5)
Lblrgebnado.Text = numero
juego = Byte.Parse(Lblnumero.Text)
```

```

If numero < juego Then
    Lblmensaje.Text = "gano"
    Lblmensaje.ForeColor = Color.Red
    puntos += 1
Else
    Lblmensaje.Text = "perdio"
    Lblmensaje.ForeColor = Color.Blue
EndIf
lblpuntos.Text = "tiene"& puntos &"puntos en"& jugadas &"jugadas"
Btnmayor.Enabled = False
Btningual.Enabled = False
Btnmenor.Enabled = False
Btnjugar.Enabled = True

Dim numero, juego AsByte
jugadas += 1
numero = Int((25 - 5 + 1) * Rnd() + 5)
Lblresultado.Text = numero
juego = Byte.Parse(Lblnumero.Text)
If numero = juego Then
    Lblmensaje.Text = "gano"
    Lblmensaje.ForeColor = Color.Red
    puntos += 3
Else
    Lblmensaje.Text = "perdio"
    Lblmensaje.ForeColor = Color.Blue
EndIf
lblpuntos.Text = "tiene"& puntos &"puntos en"& jugadas &"jugadas"
Btnmayor.Enabled = False
Btningual.Enabled = False
Btnmenor.Enabled = False
Btnjugar.Enabled = True

Dim numero, juego AsByte
jugadas += 1
numero = Int((25 - 5 + 1) * Rnd() + 5)
Lblresultado.Text = numero
juego = Byte.Parse(Lblnumero.Text)
If numero > juego Then
    Lblmensaje.Text = "gano"
    Lblmensaje.ForeColor = Color.Red
Else
    Lblmensaje.Text = "perdio"
    Lblmensaje.ForeColor = Color.Blue
EndIf
lblpuntos.Text = "tiene"& puntos &"puntos en"& jugadas &"jugadas"
Btnmayor.Enabled = False
Btningual.Enabled = False
Btnmenor.Enabled = False
Btnjugar.Enabled = True

puntos = 0

```

```

jugadas = 0
Lblnumero.Text = String.Empty
Lblrgebnado.Text = String.Empty
Lbldatos.Text = String.Empty
lblpuntos.Text = "tiene" & puntos & "puntos en" & jugadas & "jugadas"
Btnmayor.Enabled = False
Btngual.Enabled = False
Btnmenor.Enabled = False
Btnjugar.Enabled = True

```

Aplicación desarrollada Nro. III-13

Este programa consiste en un juego con numeros que se generan al azar utilizando la función Rnd.



En este juego, el jugador debe sacar 4 números al azar entre 5 y 95 y para que sea ganador, los cuatro números que genera al azar deben estar ordenados en forma ascendente.

Si antes de sacar los cuatro números al azar se saca un número que no está ordenado en forma ascendente, se muestra un mensaje de error y se reinicia el juego:

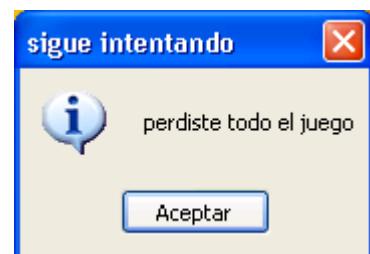


Por cada número correcto que se genera al azar clic en el botón jugar se va mostrando un mensaje:



Si uno de los cuatro numeros no sale ordenado

Alfabeticamente se muestra el mensaje explicativo anteriormente y se limpia todo el juego para iniciiar uno nuevo.



La siguiente ventana se visualiza cuando se inicia o pierde una jugada.



Controles del formulario.



Los controles Labels tienes un tamaño de la letra asignado en su propiedad Font y las siguientes propiedades:

AutoSize	False
BorderStyle	FixedSingle
.TextAlign	MiddleRight

Instrucciones del programa

Antes de escribir las instrucciones se debe crear el siguiente procedimiento llamado Limpia.

Este procedimiento limpia el contenido de todos los controles Labels y es llamado cuando se inicia el juego o cuando el jugador pierde.

Procedimiento Limpia

```
Sub limpia()
'Limpia todos los labels
Llbl1.Text = String.Empty
Llbl2.Text = String.Empty
Llbl3.Text = String.Empty
Llbl4.Text = String.Empty
EndSub
```

Instrucciones del evento Load del formulario:

'Limpia al procedimiento que limpia todas los labels

call limpia()

'Evita que la secuencia de numeros al azar se repita en cada jugada

Randomize()

Estas instrucciones se ejecutan cuando se inicia el juego y llaman al procedimiento limpia y ejecuta la función Randomize para la secuencia de números al azar que se generan al hacer clic en el botón Jugar no se repite.

Instrucciones del botón BtnJugar:

Johan Guerreros Montoya

```

Static jugadas AsByte
Dim a, b, numero AsInteger
    jugadas += 1
numero = Int((95 - 5 + 1) * Rnd() + 5)
SelectCase jugadas
Case 1
    Lblv1.Text = numero.ToString

    Llblv2.Text = String.Empty
    Llblv3.Text = String.Empty
    Llblv4.Text = String.Empty

    lblmensaje.Text = "gracias por jugar"
Case 2
    Llblv2.Text = numero.ToString
    a = Integer.Parse(Lblv1.Text)
    b = Integer.Parse(Llblv2.Text)

    If Math.Max(a, b) Then
        lblmensaje.Text = "muy bien"
    Else
        lblmensaje.Text = ""
        MsgBox("perdiste", MsgBoxStyle.Information, "sigue intentando")
        Call limpia()
        jugadas = 0
    EndIf
Case 3
    Llblv3.Text = numero.ToString
    a = Integer.Parse(Llblv2.Text)
    b = Integer.Parse(Llblv3.Text)

    If Math.Max(a, b) = b Then
        lblmensaje.Text = "muy bien, solo te falta un numero"
    Else
        lblmensaje.Text = ""
        MsgBox("perdiste", MsgBoxStyle.Information, "sigue intentando")
        Call limpia()
        jugadas = 0
    EndIf
Case 4
    Llblv4.Text = numero.ToString
    a = Integer.Parse(Llblv3.Text)
    b = Integer.Parse(Llblv4.Text)

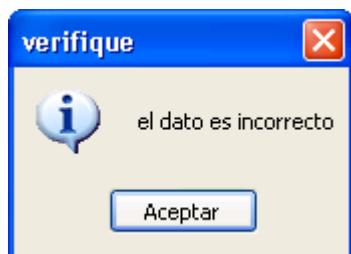
    If Math.Max(a, b) = b Then
        lblmensaje.Text = "ganaste, felicitacionees"
    Else
        lblmensaje.Text = ""
        MsgBox("perdiste todo el juego", MsgBoxStyle.Information, "sigue intentando")
        Call limpia()
        jugadas = 0
    EndIf
EndSelect

```

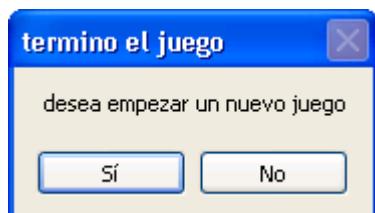
MsgBox

Esta función pertenece también a versiones anteriores de Visual Basic.Net y permite mostrar una ventana de mensaje para el usuario y en forma opcional esperar una respuesta.

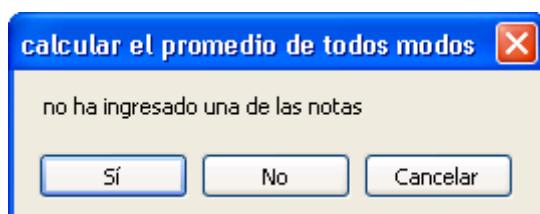
Por ejemplo, en la siguiente ventana solo se muestra un mensaje para el usuario.



En la siguiente ventana se muestra un mensaje para el usuario y se separa una respuesta:



En la siguiente ventana también se muestra un mensaje para el usuario y se espera una respuesta:



Cuando solo desea enviar un mensaje al usuario, se utiliza la siguiente sintaxis:

```
MsgBox ("mensaje", valor, "titulo")
```

El mensaje es el texto que se muestra en el centro de la ventana, se puede utilizar Ch(13) y Ch(10) para que ocupe varias líneas, el valor es un numero o constante que indica los botones e iconos que van a acompañar al mensaje, y el titulo es el texto que se muestra en la parte superior de la ventana.

En el siguiente ejemplo el mensaje de error critico y el titulo es Verifique.

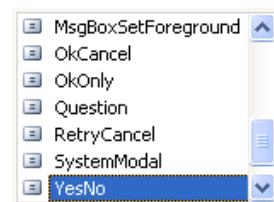


Los iconos y botones se pueden obtener escribiendo la palabra Msgboxstyle seguida de un punto se muestra a continuación.

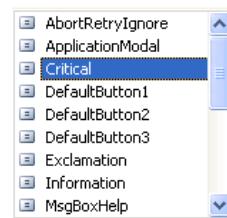
Msgstyle.

Cuando selecciona una de las opciones se visualiza un mensaje normalmente de fondo amarillo que muestra el valor de esa constante.

En el siguiente ejemplo se ha seleccionado Yes, No, que muestra los botones Si y No y el mensaje indica que sea palabra tiene valor Nro. 4. msgboxstyle.



En la siguiente muestra se ha seleccionado Critical que muestra el icono de error critico y el mensaje indica que sea palabra tiene el valor Nro. 16. Msgstyle.



Ejemplo, la siguiente instrucción:

```
MsgBox("El dato es incorrecto", MsgBoxStyle.Critical, "Verifique")
```

Muestra la siguiente ventana:



El botón aceptar tiene valor cero (0), por lo que se visualiza sin especificarlo.

La siguiente instrucción también muestra la ventana anterior, porque el icono de error critico tiene el valor 16.

```
MsgBox("El dato es incorrecto", 16, "Verifique")
```

ICONOS PARA LA FUNCION MSGBOX



Icono de información



Icono de error crítico



Icono de exclamación



Icono de interrogación

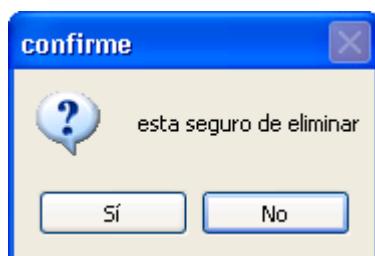
Después del parámetro iconos se puede indicar con una constante que botón debe mostrarse seleccionado. Para ello se utiliza cualquiera de las siguientes constantes:

- MsgBoxStyle.DefaultButton1
- MsgBoxStyle.DefaultButton2
- MsgBoxStyle.DefaultButton3

La siguiente instrucción muestra los botones si y no con el botón No seleccionado en forma predeterminada.

Dim N AsInteger

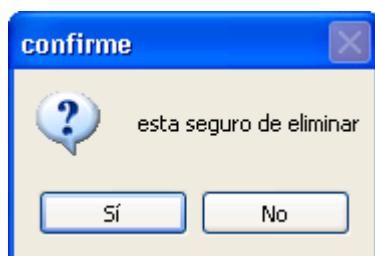
```
N = MessageBox.Show("esta seguro de eliminar", "confirme", _ MessageBoxButtons.YesNo,
MessageBoxIcon.Question, _ MessageBoxDefaultButton.Button2)
```



Si no se especifica este parámetro, se visualiza el mensaje con el primer botón seleccionado en forma predeterminada.

Dim N AsInteger

```
N = MessageBox.Show("esta seguro de eliminar", "confirme", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)
```



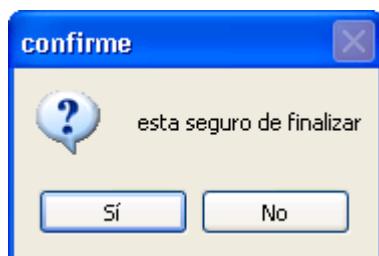
Cuando se espera una respuesta del usuario, la función messagebox devuelve cualquiera de las siguientes constantes o valores:

- Windows.Forms.DialogResult.Abort
- Windows.Forms.DialogResult.Cancel
- Windows.Forms.DialogResult.Ignore
- Windows.Forms.DialogResult.No
- Windows.Forms.DialogResult.None
- Windows.Forms.DialogResult.OK
- Windows.Forms.DialogResult.Retry
- Windows.Forms.DialogResult.Yes

Cada una de estas constantes tiene un valor similar a la función msgbox, por ejemplo, la constante yes (si) tiene el valor 6.

Las siguientes instrucciones de ejemplo preguntan si el usuario desea finalizar el programa. Si el usuario responde si, el programa finaliza:

```
Dim N AsInteger
N = MessageBox.Show("esta seguro de finalizar", "confirme", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)
If N = 6 Then
Close()
EndIf
```

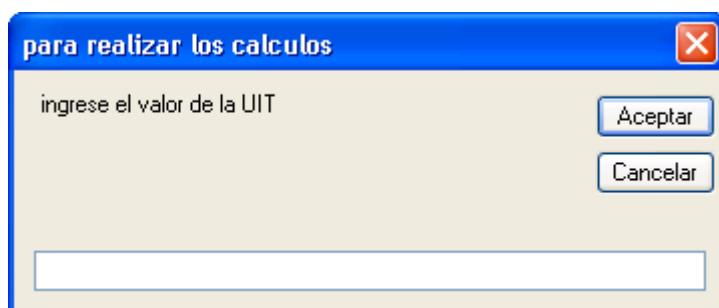


Las siguientes instrucciones cumplen la misma función anterior:

```
If MessageBox.Show("esta seguro de finalizar", "confirme", MessageBoxButtons.YesNo,
MessageBoxIcon.Question) = Windows.Forms.DialogResult.Yes Then
Close()
EndIf
```

InptBox

Esta función permite mostrar una ventana de mensaje para que el usuario ingrese un dato. ejemplo



Su sintaxis es la siguiente: InputBox ("Mensaje", "titulo", valor predeterminado, x, y)

El mensaje es el texto que se muestra en el centro de la ventana, se puede utilizar chr (13) y chr(10) para que ocupe varias líneas. El título es el texto que se muestra en la parte superior de la ventana.

El valor predeterminado es opcional y como su nombre lo indica, se utiliza para establecer un valor que debe mostrar en la caja de ingreso.

X es también un valor opcional que indica la posición horizontal de la ventana dentro de la pantalla.

Y es también un valor opcional que indica la posición vertical de la ventana dentro de la pantalla.

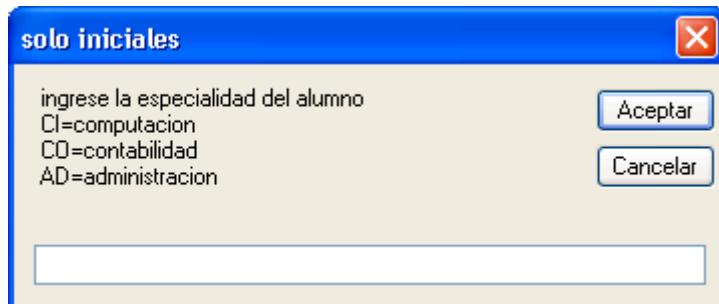
Ejemplo: con la siguiente instrucción se muestra la ventana anterior.

`InputBox("ingrese el valor de la UIT", "para realizar los calculos")`

El valor que se ingresa en la ventana se considera de tipo String.

La siguiente instrucción de ejemplo muestra una ventana para ingresar la especialidad del alumno .Utiliza en el mensaje varias lineas de texto.

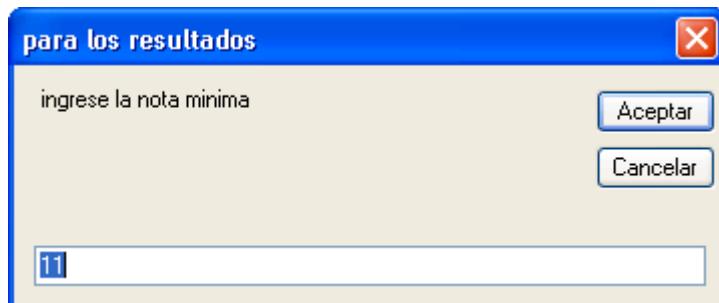
```
InputBox("ingrese la especialidad del alumno"
& Chr(13) & Chr(10) &"CI=computacion"
& Chr(13) & Chr(10) &"CO=contabilidad"
& Chr(13) & Chr(10) &"AD=administracion", "solo iniciales")
```



La siguiente instrucción de ejemplo muestra una ventana para que el usuario ingrese la nota minima. Tiene como valor predeterminados la nota 11 y una posicion horizontal y vertical.

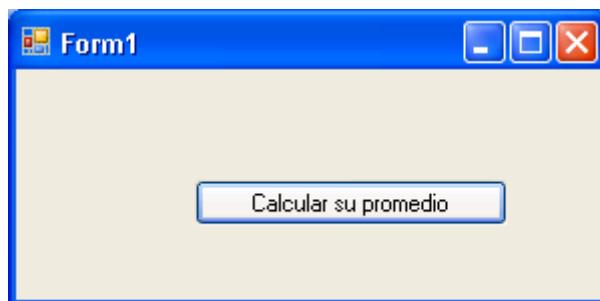
`Dim nota AsString`

```
nota = InputBox("ingrese la nota minima", "para los resultados", 11, 400, 300)
```

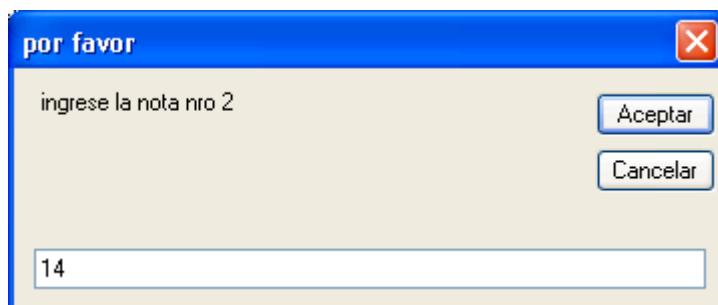
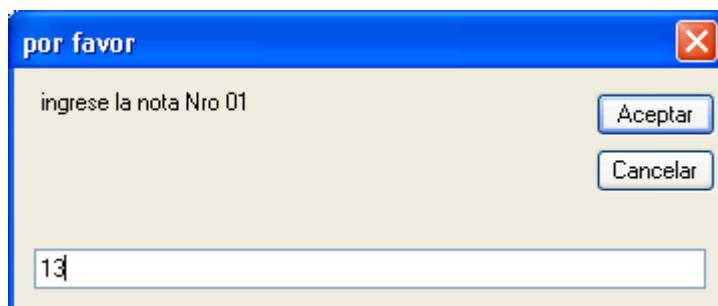


Aplicación desarrollada Nro. III-14

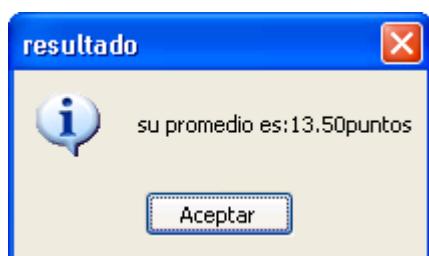
Este programa consiste en calcular al promedio de dos notas que se ingresan utilizando la función `inputbox`.el resultado se visualiza utilizando la función `msgbox`.



Al hacer clic en el botón se piden las dos notas y se calcula el promedio como se muestra a continuación:



Despues de ingresar las dos notas se visualiza el promedio del alumno:

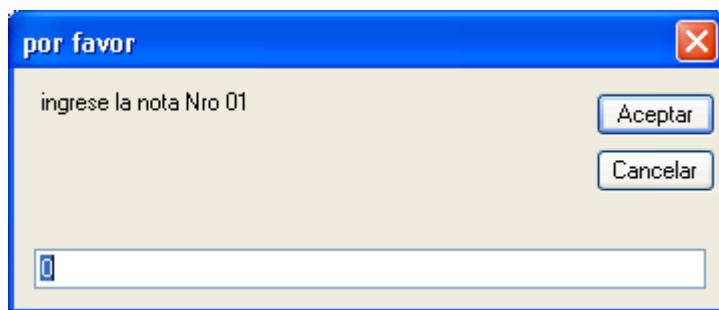


Instrucciones del boton Calcular su promedio

```
Dim dato AsString
Dim nota1, nota2, promedio AsDouble
dato = InputBox("ingrese la nota Nro 01", "por favor")
nota1 = Double.Parse(dato)
dato = InputBox("ingrese la nota nro 2", "por favor")
nota2 = Double.Parse(dato)
promedio = (nota1 + nota2) / 2
MsgBox("su promedio es:" & promedio.ToString("#0.00") &"puntos", MsgBoxStyle.Information,
"resultado")
dato = InputBox("ingrese la nota Nro 01", "por favor", 0)
```

para evitar algun error de ingreso se puede asignar como valor predeterminado de cada nota el valor 0, por ejemplo:

```
dato = InputBox("ingrese la nota Nro 01", "por favor", 0)
```



En el siguiente ejemplo se ha modificado las instrucciones del botón calcular su promedio para que si no se ingresa algú8na de las notas o se hace clic en el botón cancelar, se le asigne el valor 0.

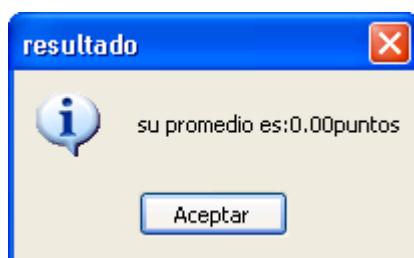
Calcular su promedio

```

Dim dato AsString
Dim nota1, nota2, promedio AsDouble
    dato = InputBox("ingrese la nota nro 01", "por favor", 0)
IfString.IsNullOrEmpty(dato) = FalseThen
nota1 = Double.Parse(dato)
Else
    nota1 = 0
EndIf
    dato = InputBox("ingrese la nota nro 02", "por favor", 0)
IfString.IsNullOrEmpty(dato) = FalseThen
nota2 = Double.Parse(dato)
Else
    nota2 = 0
EndIf
    promedio = (nota1 + nota2) / 2
    MsgBox("su promedio es:" & promedio.ToString("#0.00") & "puntos", MsgBoxStyle.Information,
"resultado")

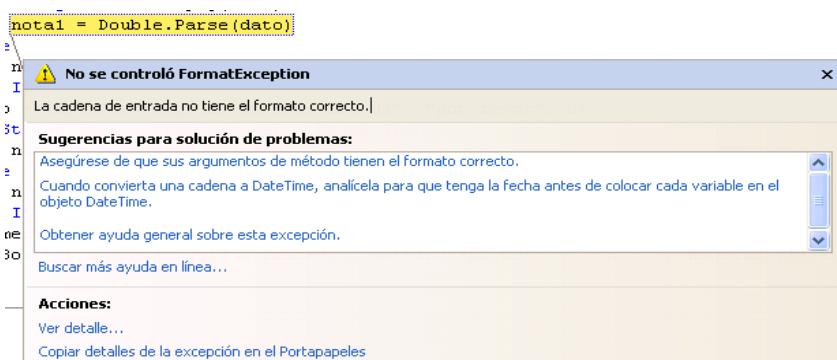
```

Si no se ingresan las notas o se hace clic en el botón cancelar se visualiza el siguiente resultado.



A pesar de los cambios realizados a las instrucciones iniciales del botón calcular su promedio, el programa se detendrá y se producirá en error, por ejemplo, si en lugar de ingresar un numero, el usuario ingresa una letra.

En el siguiente ejemplo se visualiza el programa detenido porque el usuario ha ingresado una letra en lugar de una nota:



Cuando el programa se detiene en forma inesperada por algún error producido se le llama excepción. A continuación se explica cómo controlarla:

CONTROL DE EXCEPCIONES

Una excepción es un error inesperado que puede ocurrir durante la ejecución de un programa, lo que genera que la ejecución se detenga o funciones en forma incorrecta.

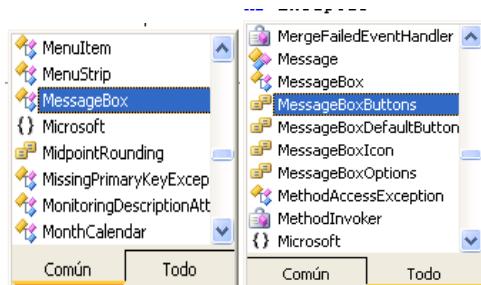
El control de excepciones en toda aplicación es muy importante, porque permite brindar información adecuada al usuario de la aplicación sobre el problema detectado, sin necesidad que se interrumpa su ejecución y permitiendo que se corrija el problema.

En Visual basic .Net existe la clase exception que permite controlar las excepciones dentro de una aplicación.

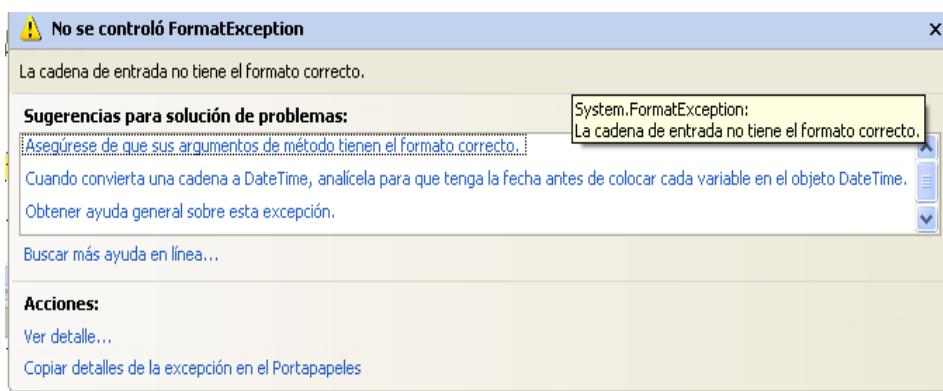
Para empezar a controlar las excepciones debe definir una variable con esta clase. Por ejemplo:

```
Dim Ex As Exception
```

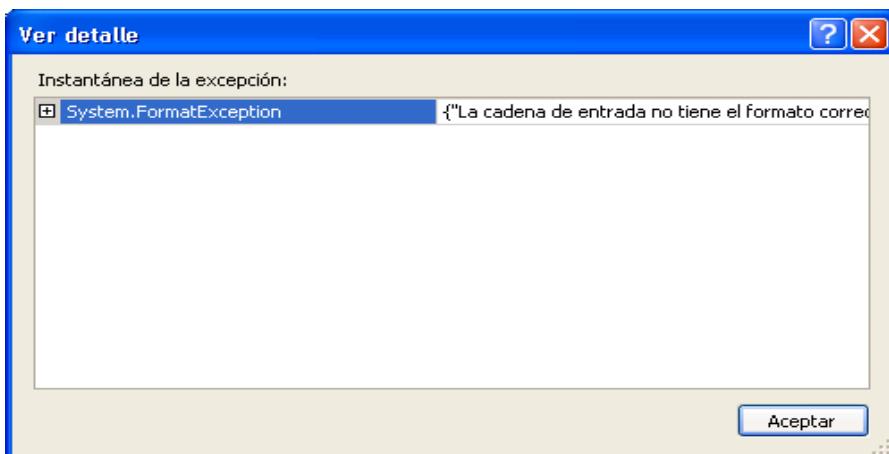
La variable que se define con la clase exception tendrá toda la información de la excepción o error que se produce. Por ejemplo: ex.



Cuando ocurre una excepción dentro de un programa y no se controla la ejecución del programa, se detiene y se visualiza una ventana similar a la siguiente, dependiendo del tipo de excepción:



Si hace clic en ver detalle, se visualiza el detalle de la excepción:



Try catch

Esta instrucción permite controlar las excepciones o errores inesperados que pueden ocurrir en visual basic .net

Su sintaxis es la siguiente:

```
try
```

Bloque de instrucciones Nro. 01

Catch variable as excepción Nro. 02

Finally

Bloque de instrucciones nro. 03

End try

Bloque de Instrucciones Nro. 01

Es el bloque de instrucciones que deseamos controlar, es decir, el bloque de instrucciones de nuestra aplicación donde puede ocurrir la excepción o error. Por ejemplo, en este bloque de instrucciones puede ir el método open, porque ocurrir un error al abrir la conexión a SQL Server, porque no existe el servidor, el nombre del usuario o su contraseña son incorrectos, hemos asignado un valor incorrecto en la cadena de conexión, etc.

Bloque de Instrucciones Nro. 02

Johan Guerreros Montoya

Es el bloque de instrucciones que se debe ejecutar cuando ocurra una excepción durante la ejecución de la aplicación.

En este bloque de instrucciones debemos reconocer las excepción o el error producido para enviar un mensaje adecuado al usuario y evitar que nuestra aplicación deje de funcionar.

En la parte inicial de este bloque de instrucciones y después de la palabra catch, se debe definir una variable de tipo excepción producida. La variable también se puede definir al inicio del programa.

Bloque de Instrucciones Nro. 03

Bloque de Instrucciones es opcional y si se escriben, se ejecutan siempre, es decir, si se produce o no la excepción.

Si en una parte del bloque try catch desea salir de él, puede utilizar exit try.

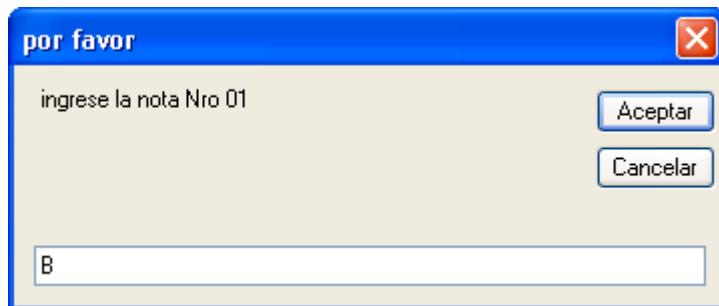
Exit try es una palabra clave que permite salir del bloque try catch y ejecutar la instrucción o instrucciones que se encuentran después de end try.

La palabra clave exit no se puede utilizar del bloque de instrucciones Nro. 03, es decir, después de la palabra finally.

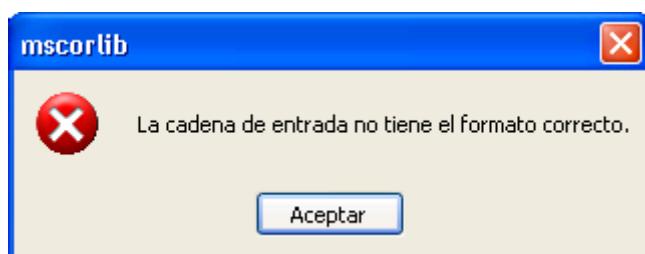
Las siguientes instrucciones son las mismas del programa anterior, pero, controlando excepciones.

```
Dim dato AsString
Dim nota1, nota2, promedio AsDouble
Try
    'ingresa la primera nota
    dato = InputBox("ingrese la nota nro 01", "por favor", 0)
    'Pregunta si esta nula o vacia
    IfString.IsNullOrEmpty(dato) = FalseThen
        nota1 = Double.Parse(dato)
    Else
        'Si esta vacia se le asigna el 0
        nota1 = 0
    EndIf
    'ingresa primera nota
    dato = InputBox("ingrese la nota nro 02", "por favor", 0)
    'Pregunta si esta nula o vacia
    IfString.IsNullOrEmpty(dato) = FalseThen
        nota2 = Double.Parse(dato)
    Else
        nota2 = 0
    EndIf
    'Calcula el promedio
    promedio = (nota1 + nota2) / 2
    'muestra el promedio
    MsgBox("su promedio es:" & promedio.ToString("#0.00") &"puntos",
    MsgBoxStyle.Information, "resultado")
    Catch ex As Exception
        MsgBox(ex.Message, MsgBoxStyle.Critical, ex.Source)
    EndTry
EndSub
```

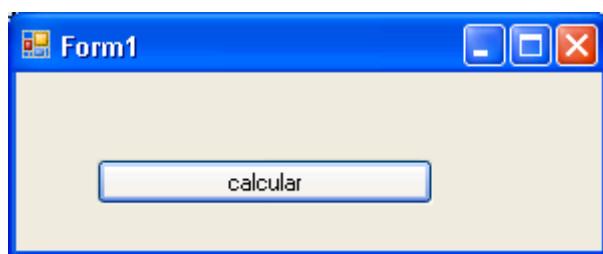
En estas instrucciones de ejemplo no se utiliza finally y se muestra el mensaje del error producido (message) y su origen (source).por ejemplo, con estas instrucciones, si el usuario escribe letras en lugar de una nota:



Ya no se interrumpe la ejecución del programa, solo se muestra una ventana de mensaje:



Al hacer clic en el botón Aceptar el programa permanece activo:



Las siguientes instrucciones son las mismas del programa anterior. Aquí se define primero la variable ex de tipo excepción y se usa el bloque Finally.

```
Dim ex As excepcion
```

```
Dim dato AsString
```

```
Dim nota1, nota2, promedio AsDouble
```

```
Try
```

```
'ingresa la primera nota
```

```
    dato = InputBox("ingrese la nota nro 01", "por favor", 0)
```

```
'Pregunta si esta nula o vacia
```

```
IfString.IsNullOrEmpty(dato) = FalseThen
```

```
    nota1 = Double.Parse(dato)
```

```
Else
```

```
    'Si esta vacia se le asigna el 0
```

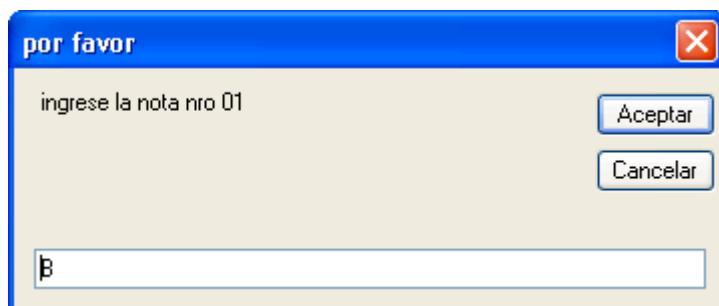
Johan Guerreros Montoya

```

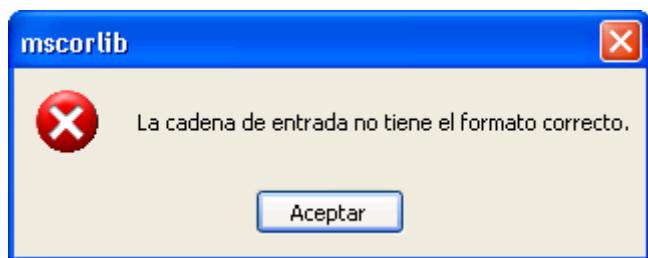
nota1 = 0
EndIf
'ingresa primera nota
dato = InputBox("ingrese la nota nro 02", "por favor", 0)
'Pregunta si esta nula o vacia
IfString.IsNullOrEmpty(dato) = FalseThen
nota2 = Double.Parse(dato)
Else
    nota2 = 0
EndIf
'Calcula el promedio
promedio = (nota1 + nota2) / 2
'muestra el promedio
MsgBox("su promedio es:" & promedio.ToString("#0.00") &"puntos",
MsgBoxStyle.Information, "resultado")
Catch ex As Exception
    MsgBox(ex.Message, MsgBoxStyle.Critical, ex.Source)
Finally
    'muestra el promedio
    Msgbox("su promedioes:" &
Promedio.ToString("#0.00") &"Puntos",msgbostyle.Information,"resultado")
EndTry
EndSub

```

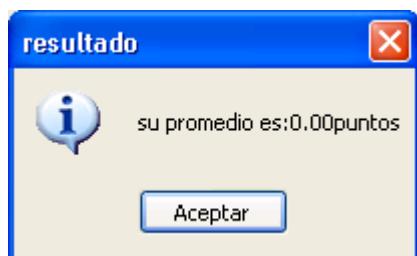
Con las instrucciones anteriores si el usuario escribe letras en lugar de una nota:



Tampoco se interrumpe la ejecución del programa, solo se muestra una ventana de mensaje.



Al hacer clic en el botón se muestra el promedio del alumno con 0.00 puntos por las instrucciones del bloque Finally.

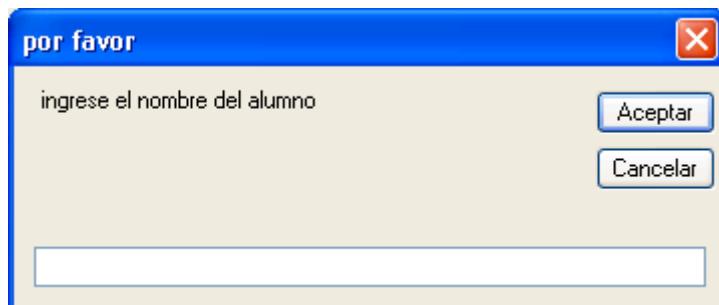


Aplicación Desarrollada Nro. III-14-B

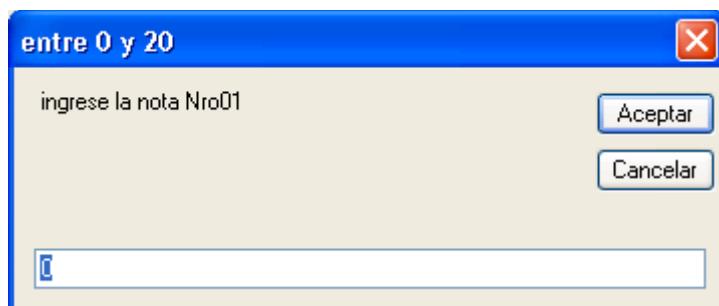
Este programa permite ingresar el nombre y tres notas de un alumno mediante la función inputbox y mostrar su promedio.



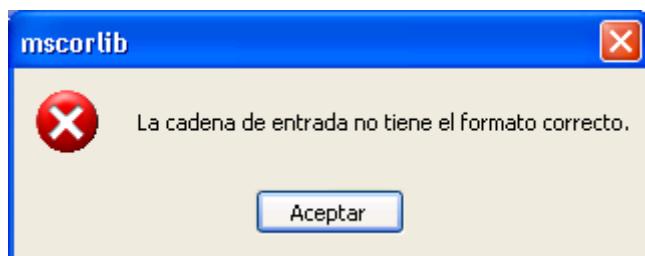
El nombre del alumno es obligatorio ingresar y el programa mostrara la siguiente ventana hasta que se ingrese.



Las notas deben estar entre 0 y 20, y el programa mostrara la siguiente ventana hasta que se ingrese la nota correcta.



En este programa también se controlan las excepciones, por ejemplo:



Instrucciones del botón ingresar:

```

Dim nombre, nota AsString
Dim I AsByte
Dim promedio AsDouble
Static puntos AsDouble
Try
'Pide el nombre hasta que nno sea una cadena vacia
Do
    nombre = InputBox("ingrese el nombre del alumno", "por favor")
LoopUntil nombre <>String.Empty
For I = 1 To 3
'Pide la nota hasta que sea mayor o igual a 0 y menor a 20
Do
    nota = InputBox("ingrese la nota Nro"& I.ToString("00"), "entre 0 y 20", 0)
LoopUntilDouble.Parse(nota) >= 0 AndDouble.Parse(nota) <= 20

puntos = puntos + Double.Parse(nota)
Next I
    promedio = puntos / 3

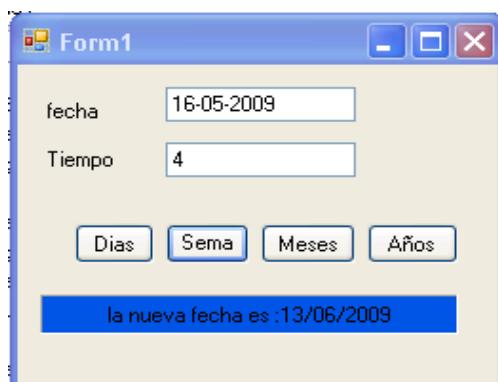
lblresultado.Text = "el promedio de:"& nombre &"es:"& promedio.ToString("#0.00")

Catch ex As Exception
    MessageBox.Show(ex.Message,ex.Source,MessageBoxButtons.OK, MessageBoxIcon.Error)
Finally
'Reiniciando los puntos del alumno
    puntos = 0
EndTry

```

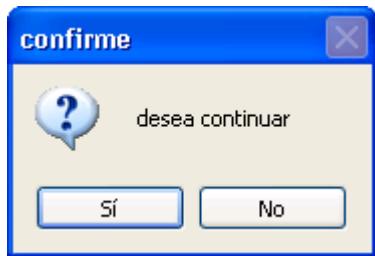
Aplicación Desarrollada Nro. III-15

Este programa es similar al programa III-03, pero, aquí se realiza el control de excepciones y otras características mas.



Si ocurre algun error inesperado,el programa no se detiene, solo muestra un mensaje que indica error.Ejemplo:

Despues de controlar la excepcion o de mostrar los resultados correctos, el programa pregunta si deseas continuar:



Instrucciones del boton dias:

```

Dim fecha, nuevafecha AsDate
Dim tiempo AsInteger
Try
    fecha = Date.Parse(Txtfecha.Text)
    tiempo = Integer.Parse(txttiempo.text)
    nuevafecha = DateAdd(DateInterval.Day, tiempo, fecha)
    Lblnuevafecha.Text = "la nueva fecha es :" & nuevafecha
Catch ex As Exception
    MessageBox.Show(ex.InnerException, ex.Source, MessageBoxButtons.OK,
    MessageBoxIcon.Error)
Finally
    If MessageBox.Show("desea continuar", "confirme", MessageBoxButtons.YesNo,
    MessageBoxIcon.Question) = Windows.Forms.DialogResult.Yes Then
        Txtfecha.Clear()
        txttiempo.clear()
        Lblnuevafecha.text = String.Empty
        Txtfecha.Focus()
    Else
        Close()
    EndIf
EndTry

```

Instrucciones del boton Semanas:

```

Dim fecha, nuevafecha AsDate
Dim tiempo AsInteger
Try
    fecha = Date.Parse(Txtfecha.Text)
    tiempo = Integer.Parse(Txttiempo.Text)
    nuevafecha = DateAdd(DateInterval.WeekOfYear, tiempo, fecha)
    Lblnuevafecha.Text = "la nueva fecha es :" & nuevafecha
Catch ex As Exception
    MessageBox.Show(ex.InnerException, ex.Source, MessageBoxButtons.OK,
    MessageBoxIcon.Error)
Finally
    If MessageBox.Show("desea continuar", "confirme", MessageBoxButtons.YesNo,
    MessageBoxIcon.Question) = Windows.Forms.DialogResult.Yes Then
        Txtfecha.Clear()

```

```

Txttiempo.Clear()
Lblnuevafecha.Text = String.Empty
Txtfecha.Focus()
Else
    Close()
EndIf
EndTry

```

Instrucciones del boton Meses:

```

Dim fecha, nuevafecha AsDate
Dim tiempo AsInteger
Try
    fecha = Date.Parse(Txtfecha.Text)
    tiempo = Integer.Parse(Txttiempo.Text)
    nuevafecha = DateAdd(DateInterval.Month, tiempo, fecha)
    Lblnuevafecha.Text = "la nueva fecha es :" & nuevafecha
Catch ex As Exception
    MessageBox.Show(ex.InnerException, ex.Source, MessageBoxButtons.OK,
    MessageBoxIcon.Error)
Finally
    If MessageBox.Show("desea continuar", "confirme", MessageBoxButtons.YesNo,
    MessageBoxIcon.Question) = Windows.Forms.DialogResult.Yes Then
        Txtfecha.Clear()
        Txttiempo.Clear()
        Lblnuevafecha.Text = String.Empty
        Txtfecha.Focus()
    Else
        Close()
    EndIf
EndTry

```

Instrucciones del boton Año:

```

Dim fecha, nuevafecha AsDate
Dim tiempo AsInteger
Try
    fecha = Date.Parse(Txtfecha.Text)
    tiempo = Integer.Parse(Txttiempo.Text)
    nuevafecha = DateAdd(DateInterval.Year, tiempo, fecha)
    Lblnuevafecha.Text = "la nueva fecha es :" & nuevafecha
Catch ex As Exception
    MessageBox.Show(ex.InnerException, ex.Source, MessageBoxButtons.OK,
    MessageBoxIcon.Error)
Finally
    If MessageBox.Show("desea continuar", "confirme", MessageBoxButtons.YesNo,
    MessageBoxIcon.Question) = Windows.Forms.DialogResult.Yes Then
        Txtfecha.Clear()
        Txttiempo.Clear()
        Lblnuevafecha.Text = String.Empty
        Txtfecha.Focus()
    Else
        Close()
    EndIf
EndTry

```

CAPITULO 4

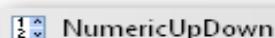
OTROS CONTROLES DE VISUAL BASIC .NET 2008

CONTENIDO

En este capítulo, usted aprenderá a utilizar otro controles que ofrece Visual Basic .NET.

- ***El Control NumericUpDown***
- ***El Control DateTimePicker***
- ***El Control MonthCalendar***
- ***El Control Timer***
- ***El Control ComboBox***
- ***El Control ListBox***
- ***El Control CheckBox***
- ***El Control RadioButton***
- ***Y Mucho Más***

El Control NumericUpDown

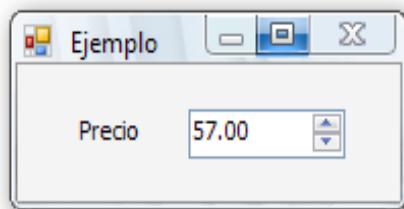


Este control permite ingresar datos numéricos en una aplicación. El ingreso puede ser digitando el numero o pulsando la flecha hacia arriba o hacia abajo para incrementar o disminuir hasta encontrar el numero deseado.

Sus principales propiedades son:

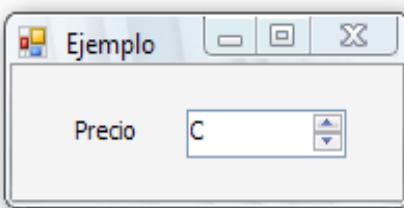
DecimalPlaces

Esta propiedad permite indicar la cantidad de decimales que debe aceptar el valor numérico que se ingresa o selecciona en el control. En la siguiente ventana de ejemplo, el control se ha configurado para 2 decimales:



Hexadecimal

Esta propiedad permite indicar si el control debe mostrar el valor en Hexadecimal. En la siguiente ventana de ejemplo esta propiedad tiene valor True y muestra el número 12 en hexadecimal.



Increment

Esta propiedad permite indicar el valor que se debe incrementar o disminuir cada vez que se pulse la flecha hacia arriba o hacia abajo. En forma predeterminada es 1. El incremento también puede ser en decimales, por ejemplo 0.5.

Maximun

Esta propiedad se utiliza para indicar el valor máximo al que se puede incrementar este control o el valor máximo que se pueda ingresar. Si se ingresa un valor mayor que el máximo establecido, el control mostrará el valor máximo al ubicar el cursor en otro control.

Mínimum

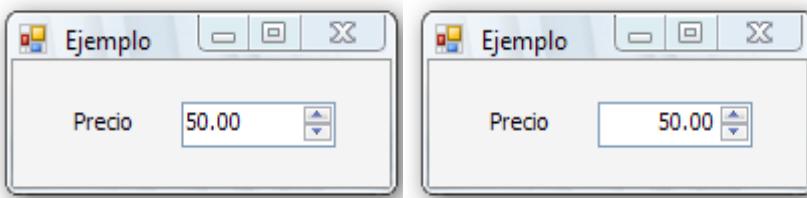
Esta propiedad se utiliza para indicar el valor mínimo al que se puede disminuir este control o el valor mínimo que se pueda ingresar. Si se ingresa un valor mínimo que el máximo establecido, el control mostrará el valor mínimo al ubicar el cursor en otro control.

ReadOnly

Esta propiedad se utiliza para indicar el usuario puede indicar el valor en el control. Si esta propiedad tiene valor True, el usuario solo podrá hacer clic en la flecha hacia arriba o hacia abajo para incrementar o disminuir el valor.

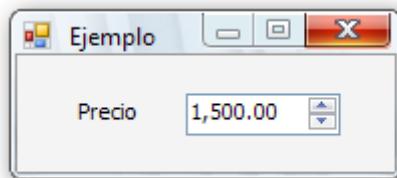
TextAlignm

Esta propiedad permite alinear el valor dentro del control y puede ser hacia la izquierda, derecha o centro.



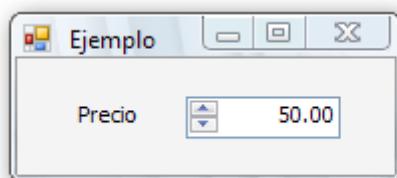
ThousandsSeparator

Esta propiedad permite indicar si el valor de este control se debe mostrar con los separadores de miles. En el siguiente ejemplo esta propiedad tiene valor True:



UpDownAlign

Esta propiedad permite indicar la ubicación de la flecha del control que puede ser a la derecha o izquierda como se muestra a continuación.

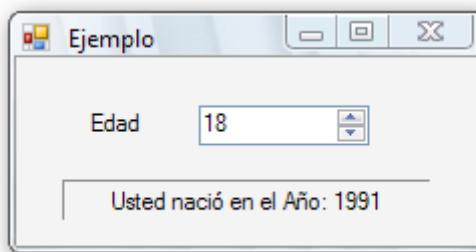


Value

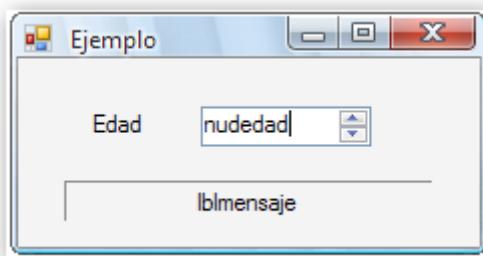
Esta propiedad almacena el valor que se digita o se selecciona en el control.

Aplicación Desarrollada N^a IV-01

Este programa permite ingresar o seleccionar en un control NumericUpDown la edad de una persona y mostrar el año que nació.



Para desarrollar esta aplicación debe dibujar un control NumericUpDown llamado NudEdad y con un control Label llamado LblMensaje.



La edad solo se permite entre 18 y 99 años, para lo cual el control NumericUpDown se le debe asignar las siguientes propiedades.

(Name)	nudedad	DecimalPlaces	0
Hexadecimal	False	Increment	1
Maximum	99	Minimum	18

El control LblMensaje debe tener las siguientes propiedades:

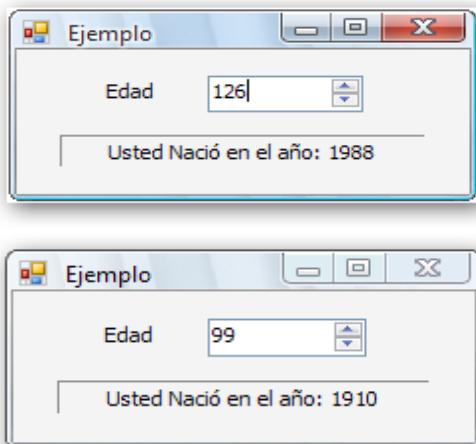
(Name)	lblmensaje
BorderStyle	Fixed3D
TextAlign	MiddleCenter

Instrucciones del evento ValueChanged del control NumericUpDown

Estas instrucciones muestran el año de nacimiento cuando el usuario escribe la edad de la persona hace clic en la flecha.

```
'Declaramos la variable edadde tipo numero
Dim edad AsByte
Declaramos las variable año, de tipo entero
Dim año AsInteger
'Digitamos la edad
edad = nudedad.Value
'Resta el año del sistema menos la edad
año = Year(Today()) - edad
'Muestra el año obtenido en el control LabelMensaje
lblmensaje.Text = "Usted Nacio en el Año: "& año
```

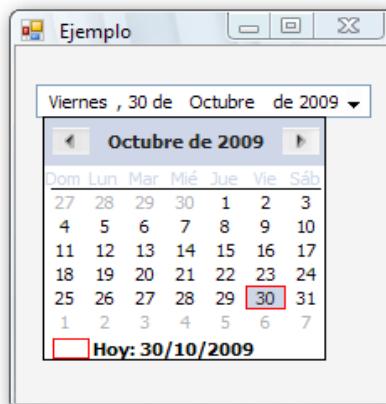
Si el usuario digita un numero mayor que el máximo (99), este valor máximo semuestra el control, al pulsar la tecla Enter o enfocar a otro control. Ejemplo:



El control DateTimePicker



Este control permite ingresar fecha en una aplicación. El ingreso se realiza mediante un calendario que muestra el control.



Sus principales propiedades son:

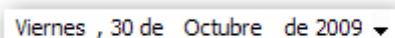
Checked

Esta propiedad trabaja junto con la propiedad ShowSelectBox y permite indicar si el usuario ha seleccionado la fecha o no.

ShowSelectBox

Esta propiedad permite indicar si el control debe tener una casilla de verificación.

La siguiente figura muestra el control con el valor True en las dos propiedades anteriores:



CustomFormat

Esta propiedad permite establecer el formato en el cual se debe visualizar la fecha y/u hora en el control cuando se le ha asignado el valor Custom en la propiedad Format.

Format

Esta propiedad permite seleccionar el formato en el cual se debe visualizar la fecha y /u hora en el control. Si se selecciona Custom la fecha se mostrara en el formato establecido en la propiedad CustomFormat.

Los formatos son:

Long

Short

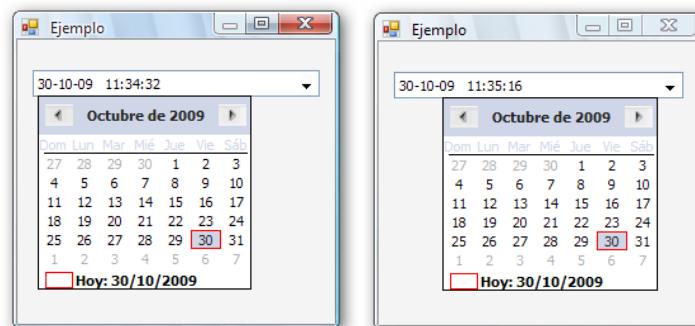
Time

Un ejemplo de formato que puede establecer en la propiedad CustomFormat es (las letras M están en mayúsculas):

Si se selecciona el formato Custom en la propiedad Format, el resultado será el siguiente:

DropDownAlign

Esta propiedad permite indicar la posición izquierda o derecha del calendario cuando se muestra en el formulario.



MaxDate

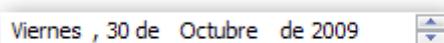
Esta propiedad permite establecer la fecha máxima que se puede seleccionar en el calendario. En forma predeterminada la fecha máxima es: 31/12/9998.

MixDate

Esta propiedad permite establecer la fecha mínima que se puede seleccionar en el calendario. En forma predeterminada la fecha mínima es: 01/01/1753.

ShowUpDown

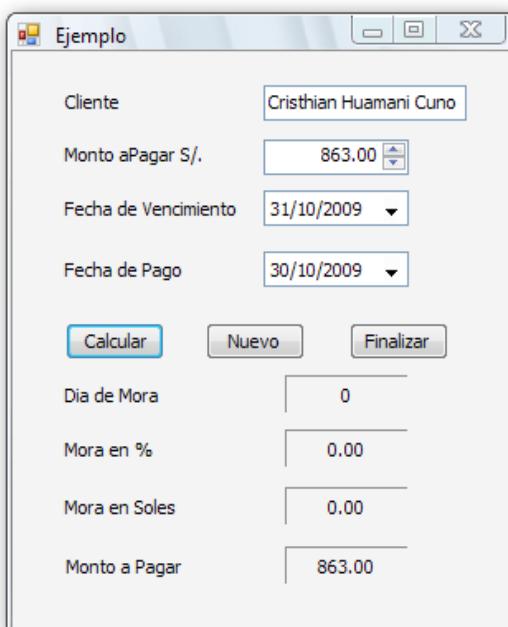
Esta propiedad permite indicar si debe mostrar las flechas hacia arriba y hacia abajo en el control. Ejemplo:

**Value**

Esta propiedad almacena la fecha y/u hora seleccionada en el control.

Aplicación Desarrollada N^a IV-02

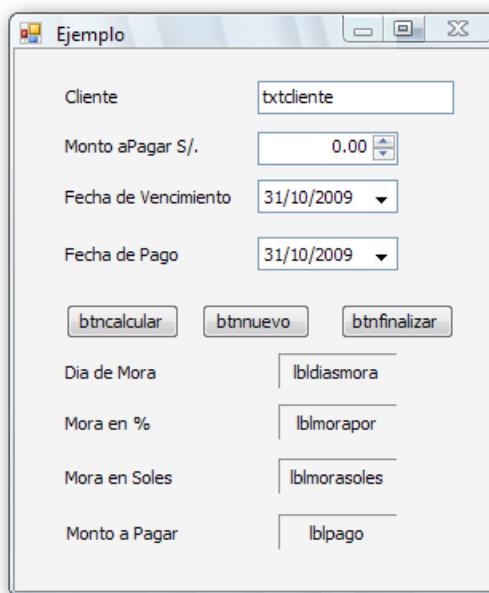
Este programa permite ingresar el nombre de un cliente, el monto de su deuda, la fecha de vencimiento y la fecha de pago.



El programa debe calcular y mostrar los días de mora, la mora en porcentaje, la mora en soles y el monto total que debe pagar el cliente. La tasa en porcentaje es 0.5% diario.

Como se debe observar en el formulario de ejemplo, si el cliente paga antes o el mismo dia de la fecha de vencimiento, la mora es cero.

Controles del formulario.



El control TxtCliente se le debe asignar el valor 50 en su propiedad MaxLength para que solo acepte hasta esa cantidad de caracteres en el nombre del cliente.

MaxLength 50

El control NudMonto debe tener las siguientes propiedades:

DecimalPlaces 2

Maximum 1000000

TextAlign Right

Los dos controles DateTimePicker deben tener en su propiedad Format la opción Short

Format Short

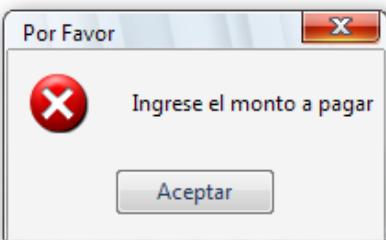
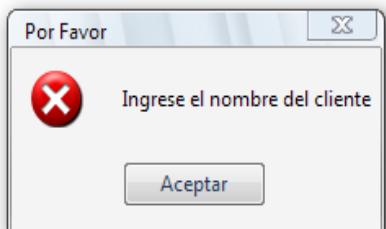
Los controles Label deben tener las siguientes propiedades:

AutoSize False

BorderStyle Fixed3D

[TextAlign](#) [MiddleCenter](#)

Si no se ingresa el nombre del cliente o el monto a pagar, visualiza un mensaje de aviso:



Instrucciones del evento Load del formulario

Estas instrucciones muestran la fecha del sistema en la fecha de vencimiento y en la fecha de pago.

```
'Asigna la fecha actual del sistema
DtpVencimiento.Value = Today()
'Asigna la fecha actual del sistema
DtpPago.Value = Today()
```

```
DtpPago.Value = Today()
```

Instrucciones del botón Nuevo

Estas instrucciones limpian los datos ingresados y los resultados obtenidos para ingresar la información de otro cliente, además muestra la fecha del sistema en la fecha de vencimiento y en la fecha de pago

```
'Limpia los controles
TxtCliente.Clear()
NudMonto.Value = 0
DtpVencimiento.Value = Today()
DtpPago.Value = Today()
LblDMora.Text = ""
LblPMora.Text = ""
LblSMora.Text = ""
```

Johan Guerreros Montoya

```
LblMPagar.Text = ""
'Ubica el cursor en la caja de texto Clientes
TxtCliente.Focus()
```

Instrucciones del botón finalizar

```
'Finaliza la aplicacion
Close()
```

Instrucciones del botón Calcular:

```
'Declaramos las variables de tipo Numero
Dim monto, moraporcentaje, morasoles, pago AsSingle
'Declaramos las variables del tipo Entero
Dim diasmora AsInteger
'Declaramos la variables de tipo Fecha
Dim fvencimiento, fpago AsDate
'Verifica que se haya ingresado el nombre del cliente
If txtcliente.Text.Trim = "" Then
    'Muestra un mensaje pidiendo que se ingrese el Nombre del Cliente
    MsgBox("Ingrese el nombre del cliente", MsgBoxStyle.Critical, "Por Favor")
    'Posiciona el cursor en la caja de texto Cliente
    txtcliente.Focus()
    'Finaliza la subrutina
ExitSub
    'Finaliza la condicion
EndIf

'Verifica que se haya ingresado el monto apagar
If nudmonto.Value <= 0 Then
    'Muestra un mensaje pidiendo que ingrese el mosto a pagar
    MsgBox("Ingrese el monto a pagar", MsgBoxStyle.Critical, "Por Favor")
    'Posiciona el cursor en la caja de texto monto
    nudmonto.Focus()
    'Finaliza la subrutina
ExitSub
    'Finaliza la condicion
EndIf
'Almacena el valor del monto
Monto = Nudmonto.Value
'Almacena la fecha
Fvencimiento = Dtpvencimiento.Value
Fpago = dtppago.Value
'Calcula la diferencia de dias entre el las fechas
    diasmora = DateDiff(DateInterval.Day, fvencimiento, fpago)
'Pregunta si hay dias de mora, entonces
If diasmora < 0 Then
    'Asigna el valor cero a DiasMora
        Diasmora = 0
    'Finaliza la condicion
EndIf

'Calcula la MoraPorcentaje
Moraporcentaje = diasmora * 0.5
```

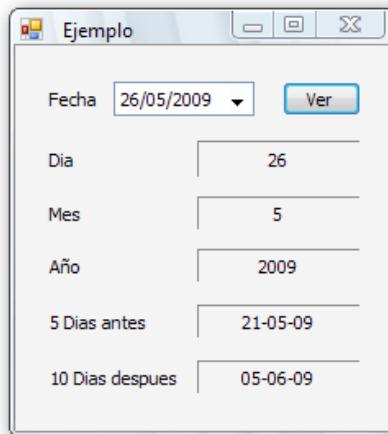
```

'Calcula la MoraSoles
Morasoles = Monto * Moraporcentaje / 100
'Calcula la Pago
Pago = Monto + Morasoles
'Muestra los dias mora
    lbldiasmora.Text = diasmora
'Calcula el Promedio Mora
    lblmorapor.Text = moraporcentaje.ToString("##0.00")
'Calcula los soles de mora
    lblmorasoles.Text = morasoles.ToString("###,##0.00")
'Calcula el monto a pagar
    lblpago.Text = pago.ToString("###,##0.00")

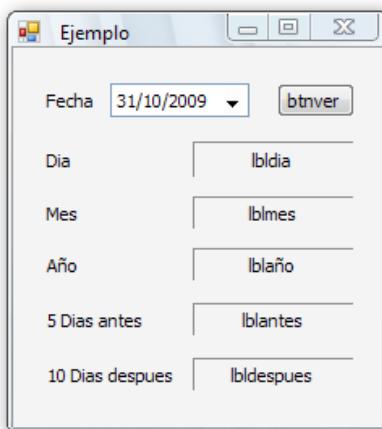
```

Aplicación Desarrollada N^aIV – 02B

Este programa permite ingresar una fecha y mostrar por separado el día, mes y año de la fecha, así como la fecha 5 días antes y 10 días después.



Los controles del formulario son:



Los controles Label, donde se muestran los resultados deben tener las siguientes propiedades:

AutoSize	False
BorderStyle	Fixed3D
TextAlign	MiddleCenter

Instrucciones del evento Load del formulario:

Instrucciones del botón BtnVer:

```

'Declaramos las variables de tipo Fecha
Dim fecha, antes, despues AsDate
'Declaramos las variables de tipo entero
Dim dia, mes, año AsInteger
'Almacena la fecha
    fecha = dtpfecha.Value
'Obtiene el dia de la fecha
    dia = fecha.Day
'Obtiene el mes de la fecha
    mes = fecha.Month
'Obtiene el año de la fecha
    año = fecha.Year
'Calcula el intervalo que hay 5 dias antes
    antes = DateAdd(DateInterval.Day, -5, fecha)
'Calcula el intervalo que hay 10 dias despues
    despues = DateAdd(DateInterval.Day, 10, fecha)

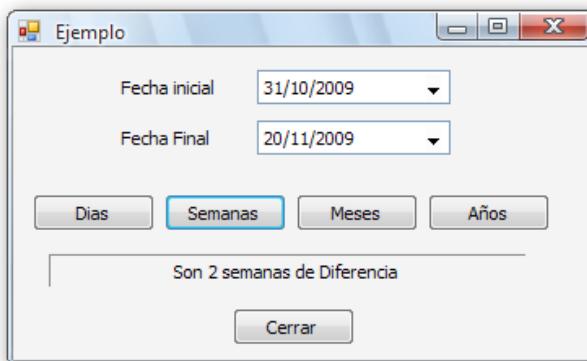
'Muestra el resultado LblDia de tipo cadena
lbldia.Text = dia.ToString
'Muestra el resultado LblMes de tipo cadena
    lblmes.Text = mes.ToString
'Muestra el resultado LblAño de tipo cadena
    lblaño.Text = año.ToString

```

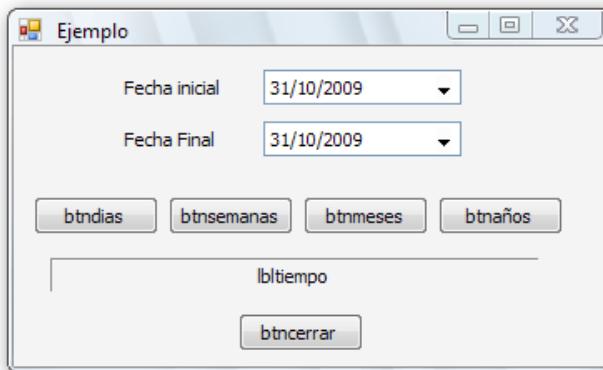
```
'Muestra el resultado LblAntes de tipo cadena
lblantes.Text = antes.ToString("dd-MM-yy")
'Muestra el resultado LblDespues de tipo cadena
lbldespues.Text = despues.ToString("dd-MM-yy")
```

Aplicación Desarrollada N^a IV – 02C

Este programa permite seleccionar dos fechas y mostrar su diferencia en días, semanas, meses y años.



Controles del formulario



Instrucciones del evento Load del formulario

Estas instrucciones asignan la fecha actual a los dos controles DateTimePicker.

```
'Asigna la fecha actual
DtpFInicial.Value = Today()
'Asigna la fecha actual
DtpFFinal.Value = Today()
```

Instrucciones del botón BtnDias:

'Declaramos las variables de tipo fecha

Johan Guerreros Montoya

```

Dim fecha1, fecha2 AsDate
'Declaramos las variables de tipo entero
Dim tiempo AsInteger
'Almacena el valor de la Fecha1
fecha1 = dtpinicial.Value
'Almacena el valor de la Fecha2
fecha2 = dtpfinal.Value
'Calcula la diferencia de dias entre el las fechas
tiempo = DateDiff(DateInterval.Day, fecha1, fecha2)
'Muestra el resultado obtenido de la diferencia
lbltiempo.Text= "Son" & tiempo & " dias de Diferencia"

```

Instrucciones del botón BtnSemanas:

```

'Declaramos las variables de tipo fecha
Dim fecha1, fecha2 AsDate
'Declaramos las variables de tipo entero
Dim tiempo AsInteger
'Almacena el valor de la Fecha1
fecha1 = dtpinicial.Value
'Almacena el valor de la Fecha2
fecha2 = dtpfinal.Value
'Calcula la diferencia de dias entre el las fechas
tiempo = DateDiff(DateInterval.Weekday, fecha1, fecha2)
'Muestra el resultado obtenido de la diferencia
lbltiempo.Text= "Son" & tiempo & " semanas de Diferencia"

```

Instrucciones del botón BtnMeses:

```

'Declaramos las variables de tipo fecha
Dim fecha1, fecha2 AsDate
'Declaramos las variables de tipo entero
Dim tiempo AsInteger
'Almacena el valor de la Fecha1
fecha1 = dtpinicial.Value
'Almacena el valor de la Fecha2
fecha2 = dtpfinal.Value
'Calcula la diferencia de dias entre el las fechas
tiempo = DateDiff(DateInterval.Month, fecha1, fecha2)
'Muestra el resultado obtenido de la diferencia
lbltiempo.Text = "Son " & tiempo & " meses de Diferencia"

```

Instrucciones del botón BtnAños:

```

'Declaramos las variables de tipo fecha
Dim fecha1, fecha2 AsDate
'Declaramos las variables de tipo entero
Dim tiempo AsInteger
'Almacena el valor de la Fecha1
fecha1 = dtpinicial.Value
'Almacena el valor de la Fecha2
fecha2 = dtpfinal.Value
'Calcula la diferencia de dias entre el las fechas

```

```

tiempo = DateDiff(DateInterval.Year, fecha1, fecha2)
'Muestra el resultado obtenido de la diferencia
lbltiempo.Text = "Son "& tiempo &" años de Diferencia"

```

El control MonthCalendar

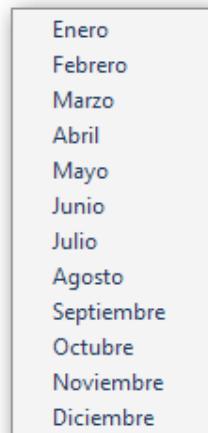


Este control permite mostrar en un formulario un calendario, pero, sólo son algunos meses determinados. También se puede utilizar para seleccionar e ingresar una fecha a una aplicación.

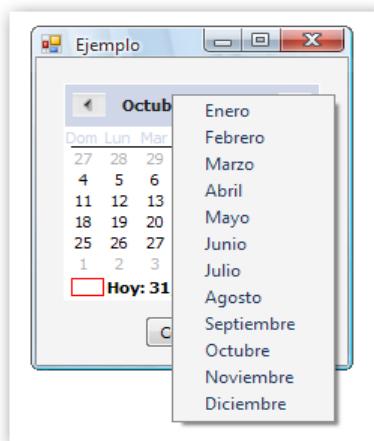


Para seleccionar un nuevo mes, puede hacer clic en la flecha hacia la derecha o izquierda que se encuentra en le título del calendario. También se puede hacer clic en el nombre del mes, con lo cual se visualizan todos los meses del año.

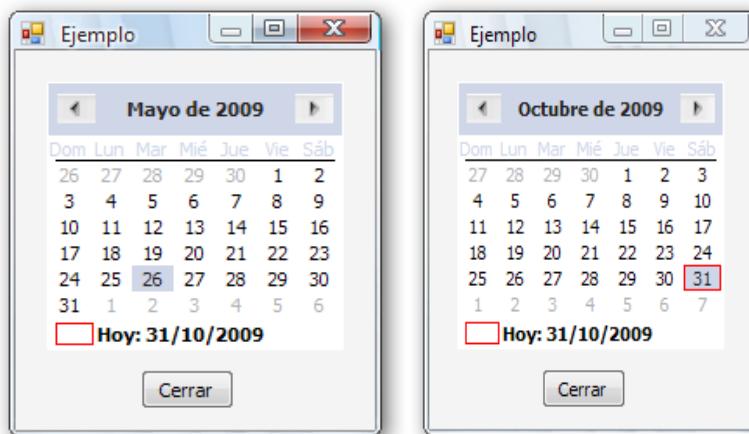
Para seleccionar un nuevo año, puede hacer clic en el año que se encuentra en el título del calendario con lo cual se muestra un control UpDown junto al año para cambiarlo.



En la siguiente ventana de ejemplo se ha hecho clic en el nombre del mes:



Si se encuentra en cualquier fecha y desea ir a la fecha actual, puede hacer clic en la casilla que dice Hoy:

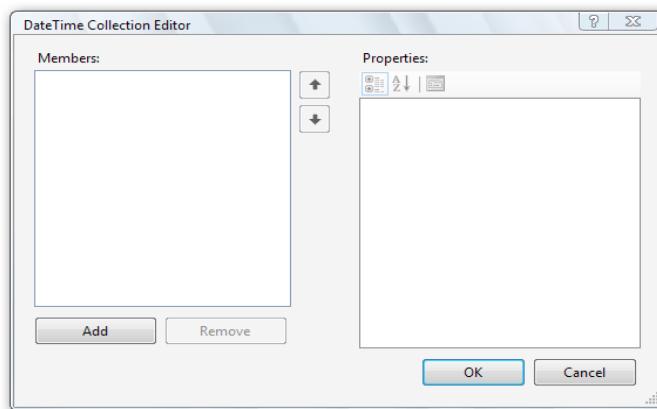


Sus principales actividades son:

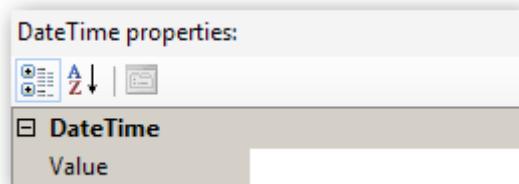
AnnuallyBoldedDates

AnnuallyBoldedDates `DateTime[] Array`

Esta propiedad se utiliza para indicar las fechas de cualquier año que desea visualizar en Negrita cuando visualice el calendario.

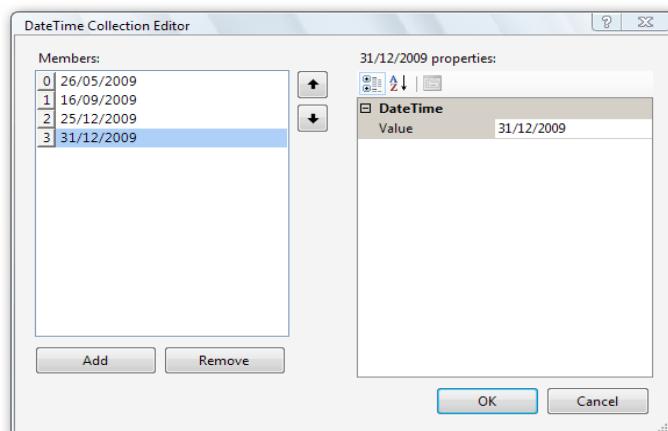


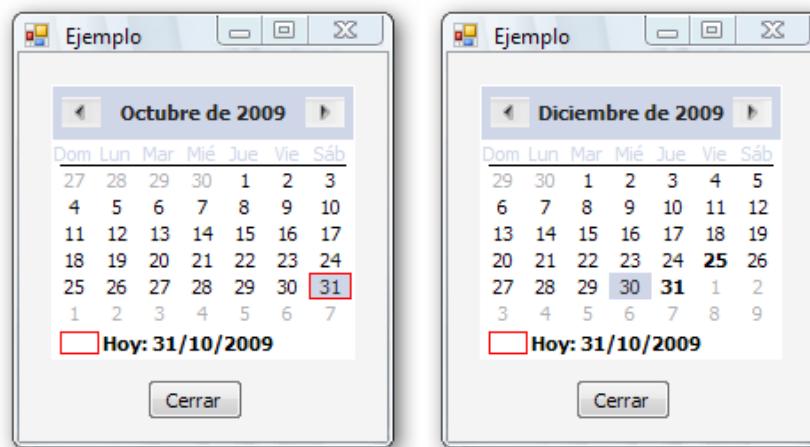
El ingresar a esta propiedad, se visualiza la siguiente ventana:



En esta ventana haga clic en el botón Agregar y escriba o selecciones en la propiedad Value la fecha que desea visualizar en negrita:

En la siguiente ventana de ejemplo se ha agregado 4 fechas del año 2009 para que se visualicen en negrita:



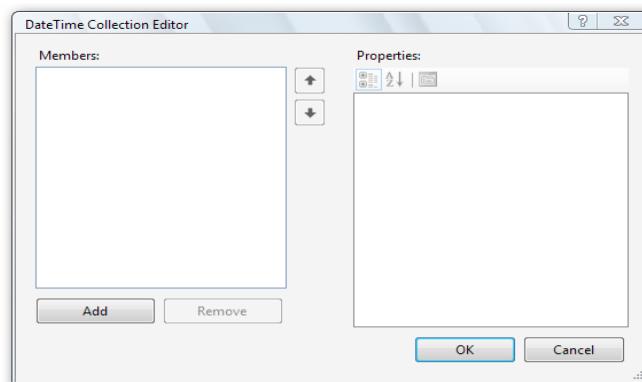


BoldedDates

Esta propiedad es similar a la propiedad anterior, se diferencian porque las fechas que aquí se seleccionan solo se muestran en negrita en el año seleccionado, para que otros años se visualicen en color normal.

BoldedDates **DateTime[] Array**

Al ingresar a esta propiedad se visualiza la misma ventana de la propiedad anterior donde se agregan las fechas que desean visualizar en negrita, pero solo en el año seleccionado.



CalendarDimensions

Esta propiedad se utiliza para indicar la cantidad de meses que desea mostrar en el formulario. Esta propiedad acepta dos valores separados por un punto y coma. El primer valor indica cantidad de columnas y el segundo, cantidad de filas. En la siguiente ventana de ejemplo se ha indicado dos columnas y una fila:

CalendarDimen: 3, 1



FirstDayOfWeek

Esta propiedad se utiliza para establecer el primer día de la semana. En forma predeterminada es Sunday (Domingo).



En la siguiente ventana de ejemplo se ha establecido como primer día de la semana el día Monday (lunes):



MaxDate

Esta propiedad permite establecer la fecha máxima que se pueda seleccionar en el calendario. En forma predeterminada la fecha máxima es: 31/12/9998.

MaxSelectionCount

Esta propiedad permite establecer la cantidad de días que puedan seleccionar en el calendario. Los días se pueden seleccionar utilizando las teclas Shift y las flechas o con el puntero del mouse. En forma predeterminada solo se pueden seleccionar 7 días.

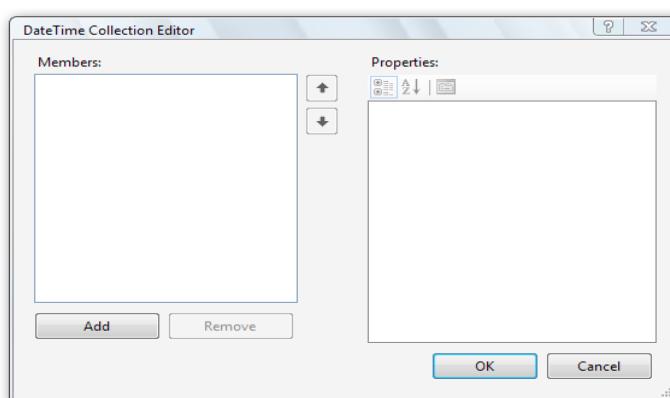
MixDate

Esta propiedad permite establecer la fecha mínima que se puede seleccionar en el calendario. En forma predeterminada la fecha mínima es: 01/01/1753.

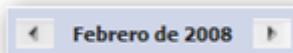
MonthlyBoldedDates

Esta propiedad permite establecer los días del mes que deben salir. Por ejemplo, si usted agrega el día 20 de febrero, entonces el día 20 de todos los meses se visualizan en negrita.

Al ingresar a esta propiedad se visualiza la misma ventana de la propiedad BoldedDates donde debe agregar los días que desean visualizar en negrita en todos los meses del año.

**ScrollChange**

Esta propiedad permite establecer la cantidad de meses que se desean saltar cuando el usuario hace clic en la fecha hacia la derecha o izquierda del calendario.

**SelectionRange**

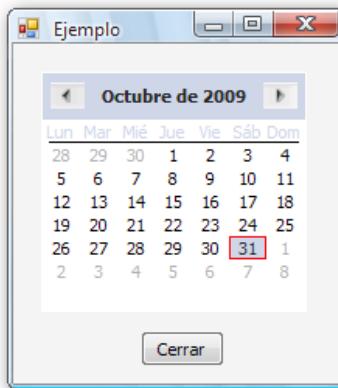
Esta propiedad almacena el rango de fechas seleccionando con el control. Las fechas se separan por un punto y coma.

En el siguiente ejemplo se ha seleccionado las fechas entre el 20-02-2006y el 26-02-2006. La fecha inicial se almacena en Start y la fecha final se almacena en End. Para visualizar esta información haga clic en el signo + de esta propiedad.

SelectionRange	31/10/2009, 31/10/2009
Start	31/10/2009
End	31/10/2009

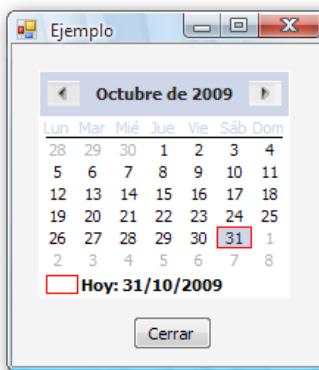
ShowToday

Esta propiedad se utiliza para indicar si en la parte inferior del calendario se debe visualizar la fecha actual. En forma predeterminada tiene el valor True. En la siguiente ventana se le ha asignado el valor False:



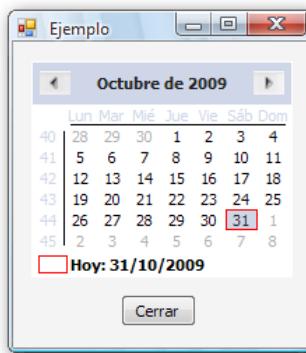
ShowTodayCircle

Esta propiedad se utiliza para indicar si en la parte inferior del calendario se debe visualizar en la fecha actual (Hoy) un rectángulo de color rojo.



ShowWeekNumbers

Esta propiedad se utiliza para indicar si las semanas del calendario se deben visualizar enumeradas. En la ventana de ejemplo se le ha asignado True.



TitleBackColor

Esta propiedad se utiliza para establecer el color de fondo del título del calendario.

TitleBackColor **ActiveCaption**

TitleForeColor

Esta propiedad se utiliza para establecer el color de las letras del título del calendario.



TrailingForeColor

Esta propiedad se utiliza para establecer el color de los numero de los días que pertenecen al mes anteriores y siguiente de los meses que se esta visualizando en el calendario.

TodayDate

Esta propiedad se utiliza para establecer o almacenar la fecha actual.



En modo de diseño el usuario puede seleccionar el dia desde la propiedad.

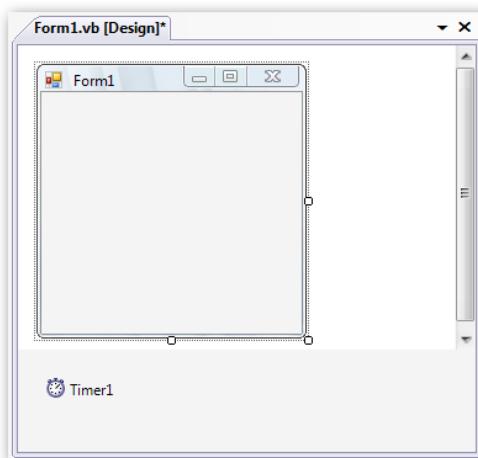


El Control Timer



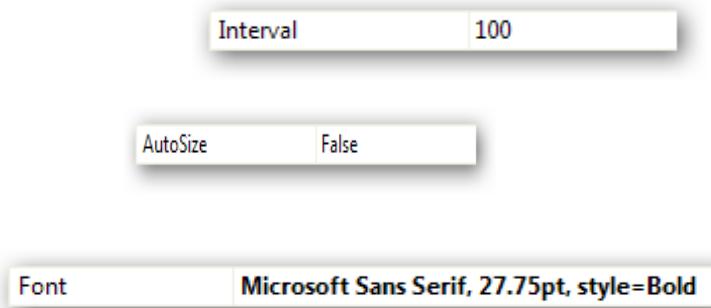
Este control permite ejecutar instrucciones cada intervalo de tiempo en el cual se deben ejecutar las instrucciones se le asigna en milisegundo, esto quiere decir, que el valor 1000 representa un segundo.

Cuando este control se dibuja en el formulario, se ubica en la parte inferior, como se muestra en la siguiente ventana de ejemplo:



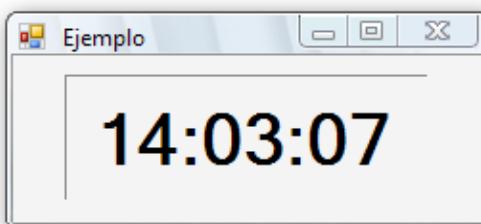
El control Timer tiene la propiedad interval, donde se indica el intervalo de tiempo en el cual se deben ejecutar las instrucciones.

La propiedad Enabled permite que se ejecuten o no las instrucciones. El True permite que se ejecuten las instrucciones.



Aplicación Desarrollada N^a IV – 02D

Este programa muestra la hora en el formulario. Funciona como un reloj digital, porque la hora va cambiando cada segundo.



Controles del formulario



El control LblReloj tiene las siguientes propiedades:

TextAlign MiddleCenter BorderStyle Fixed3D

El control Timer1 tiene las siguientes propiedades:

Enabled True Interval 1000

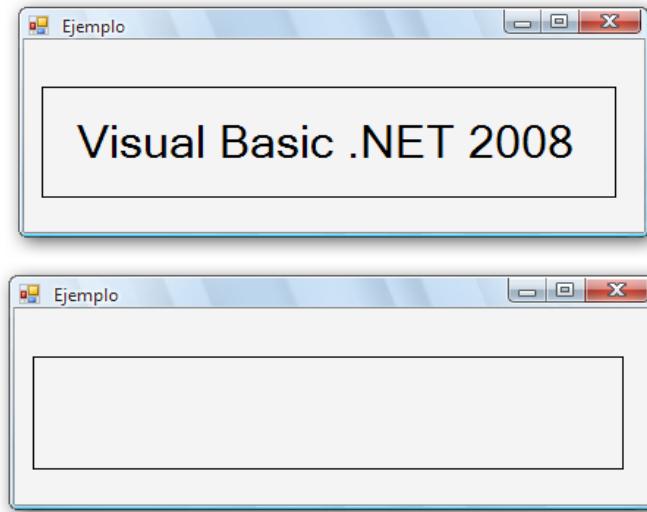
Instrucciones del evento Tick del control Timer1.

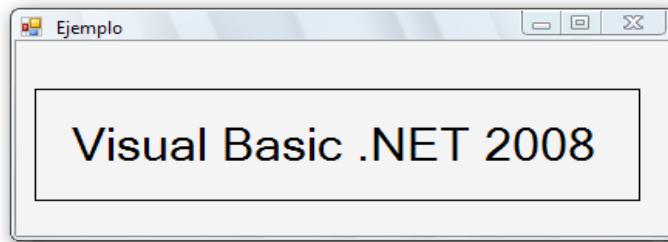
Estas instrucciones se ejecutan cada 1 segundo, porque en la propiedad interval de este control hemos asignado el valor 1000 y las instrucciones se ejecutan en forma automática por el valor True que tiene la propiedad Enabled.

LblReloj.Text = TmeString

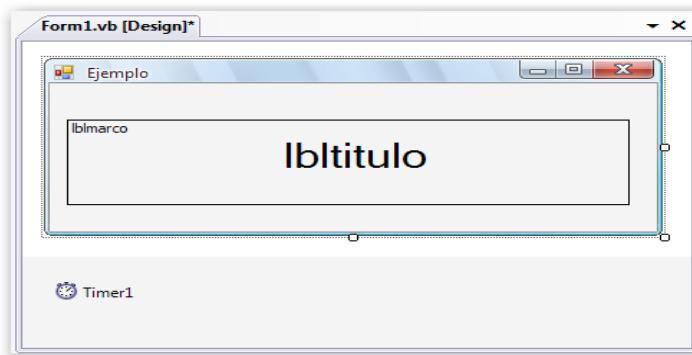
Aplicación Desarrollada N^a IV -02E

Este programa permite mostrar el mensaje: Visual Basic.Net 2008 parpadeando, es decir, ocultándose y visualizándose, cada 2 segundos.





Controles del formulario



El control LblMarco tiene las siguientes propiedades:

AutoSize | **False**

BorderStyle | **FixedSingle**

TextAlign | **MiddleCenter**

El control LblTitulo dibújelo sobre el control LblMarco y asígnele las siguientes propiedades:

AutoSize | **True**

Font | **Microsoft Sans Serif, 24pt**

Text | **Visual Basic .NET 2008**

TextAlign | **MiddleCenter**

Los valores de las propiedad Font también pueden visualizar y asignar haciendo clic en su signo +.

Font	Microsoft Sans Serif, 24pt
Name	Microsoft Sans Serif
Size	24
Unit	Point
Bold	False
GdiCharSet	0
GdiVerticalFont	False
Italic	False
Strikeout	False
Underline	False

El control Time1 tiene las siguientes propiedades:

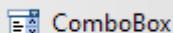
Enabled	True
Interval	2000

Instrucciones del evento Tick del control Timer1.

Estas instrucciones hacen que el control LblTitulo se visualice u se oculte cada 2 segundos por el valor 2000 que tiene el control Timer1 en su propiedad Interval.

```
LblTitulo.Visible = Not LblTitulo.Visible
```

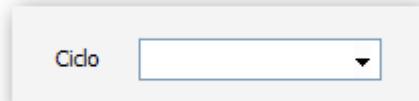
El Control ComboBox



Este control permite ingresar datos seleccionándolo desde una lista de elementos. Ejemplo:



Los elementos a seleccionar se muestran cuando el usuario hace clic en la flecha hacia abajo o pulsa la tecla F4.



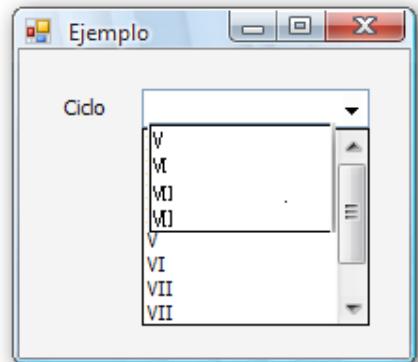
Sus propiedades son:

Name

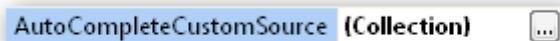
Esta propiedad se utiliza para asignar un nombre al control. Se recomienda que los nombres empiecen con las letras Cbo. Por ejemplo: CboCiclo.

AutoCompleteCustomSource

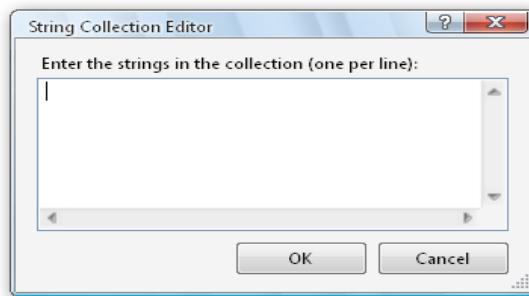
Esta propiedad se utiliza cuando deseamos que el control ComboBox auto complete alguna palabra que podemos escribir en ese control. En esta propiedad se escriben las palabras que el control ComboBox debe auto completar. Por ejemplo, si tenemos los ciclos de estudios, al escribir V se auto completa con los ciclos que empiezan con esa inicial o iniciales.



Al activar esta propiedad se visualiza la palabra Colección y un botón.



Al hacer clic en el botón con tres puntos se visualiza la siguiente ventana donde debe escribir las palabras que desea que se auto completen:



AutoCompleteMode

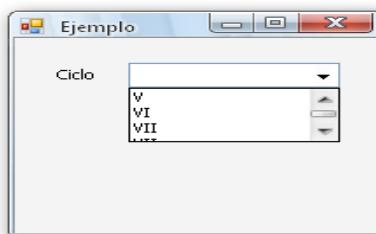
Esta propiedad se utiliza para indicar la forma como se deben auto completar las palabras en el control ComboBox y son las siguientes:

AutoCompleteSource

Esta propiedad se utiliza para indicar el origen de las palabras que se deben auto completar en el control ComboBox. Elija CustomSource para que se utilicen las palabras que ha escrito en la propiedad AutoCompleteCustomSource y elija ListItems para que se utilicen las palabras escritas en la propiedad Items.

DropDownHeight

Esta propiedad se utiliza para establecer el tamaño de la lista que contiene los elementos a seleccionar en el control ComboBox. El valor pre-determinado es 106 pixeles. En el siguiente ejemplo la propiedad tiene el valor 45.



DropDownStyle

Esta propiedad se utiliza para establecer el comportamiento del ComboBox para seleccionar los elementos.



DropDownWidth

Esta propiedad se utiliza para establecer el ancho de la lista que contiene los elementos a seleccionar en el control ComboBox. El valor predeterminado es 121 pixeles.

Items

Esta propiedad permite ingresar los elementos que el control debe mostrar. Al ingresar a esta propiedad se visualiza la siguiente ventana:



Sorted

Esta propiedad se utiliza para indicar los elementos que muestra el control deben ordenarse.

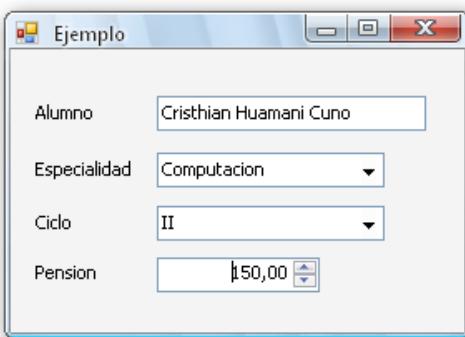
Text

Esta propiedad almacena el texto seleccionado en el control.

Este control también tiene las propiedades DataSource, DisplayMember y ValueMember que se utilizan para adelante con bases de datos.

Aplicación Desarrollada N^a IV – 03

Este programa permite Ingresar el nombre del alumno, su especialidad, ciclo y pensión. Utiliza para el ingreso un control TextBox, dos controles ComboBox y un control NumericUpDown.



La pensión mínima es 50 y la máxima es 500 y cada vez que hace clic en la flecha hacia abajo o hacia arriba el incremento es de 5 nuevos soles.

Si desea limpiar los datos para ingresar un nuevo alumno, se debe hacer doble clic en cualquier espacio del formulario y para finalizar el programa debe pulsar la tecla ESC.

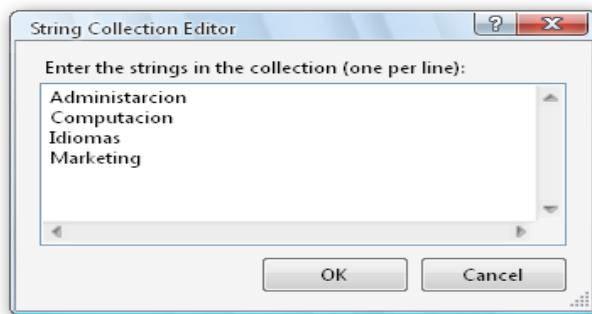
Los controles que se deben dibujar en el formulario son:



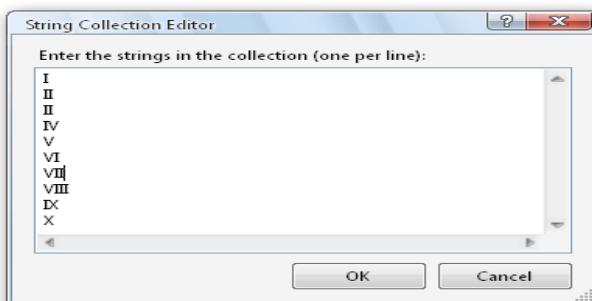
Al control TxtAlumno asígnele el valor 50 en su propiedad Maxlength para controlar la cantidad de caracteres que se deben ingresar.

Al control CboEspecialidad asígnele valor True en su propiedad Sorted para que las especialidades se muestren ordenadas alfabéticamente.

Elementos de la propiedad Item del Control CboEspecialidad



Elementos de la propiedad Item del Control CboCiclo



Al control NudPension le debe asignar las siguientes propiedades:

DecimalPlaces	2
Increment	5
Maximum	500
Minimum	50
TextAlign	Right
Value	100

El formulario debe tener valor True en su propiedad KeyPreview para detectar cuando el usuario pulse la tecla ESC para finalizar el programa.

KeyPreview	True
------------	------

Instrucciones del evento KeyPress del formulario.

Estas instrucciones preguntan si se ha pulsado la tecla ESC .si la respuesta es verdad finaliza el programa.

```
If Asc (e.KeyChar) = 27 Then Close ()
```

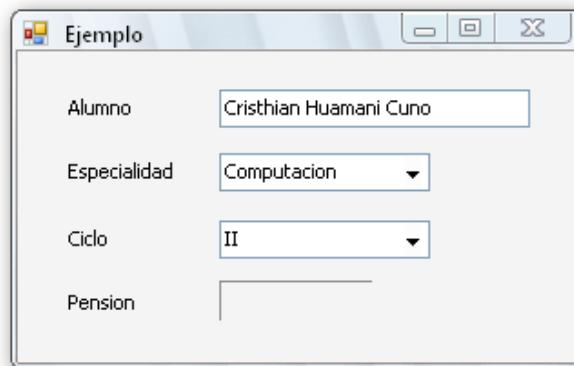
Instrucciones del evento DoubleClick del formulario.

Estas instrucciones limpian los datos ingresados para ingresar nuevos datos y asigna la pension predeterminada de 100 soles.

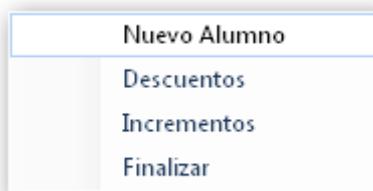
```
txtalumno.text=""
cbociclo.tezt=""
cboespecialidad=""
Nudpension.value=100
Txtalumno.Focus()
```

Aplicación Desarrollada N° IV – 04

Este programa permite ingresar el nombre de un alumno, su especialidad, ciclo y según los datos ingresados y configurados muestra su pension en forma automática. Utiliza para el ingreso un control TextBox, dos controles ComboBox y un control Label.



Este programa de ejemplo también tiene un menú contextual con las siguientes opciones.



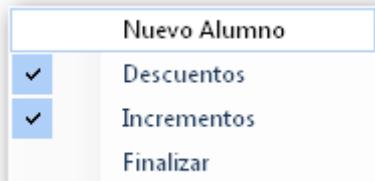
La opción Nuevo Alumno limpia los datos ingresados y la pensión del alumno para ingresar los datos de un nuevo alumno.

La opción Descuentos permite configurar el programa para que se le aplique o no descuento a la pensión de todos los alumnos que están en el I ciclo.

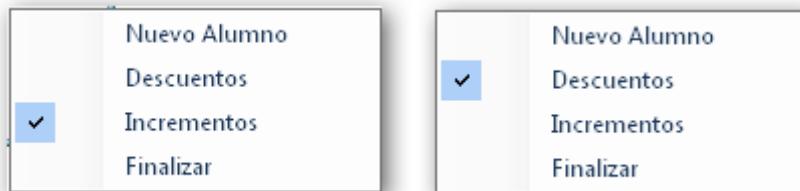
La opción Incrementos permite configurar el programa para que se le aplique o no un incremento a la pensión de todos los alumnos que se encuentran a partir del III ciclo.

El porcentaje que se incrementa a la pensión de los alumnos es del 10 % por cada ciclo a partir del III ciclo. Esto quiere decir, que un alumno del V ciclo tendrá un incremento del 30 % de la pensión normal.

Si se activan las dos opciones, solo los alumnos del II ciclo pagan la pensión exacta.

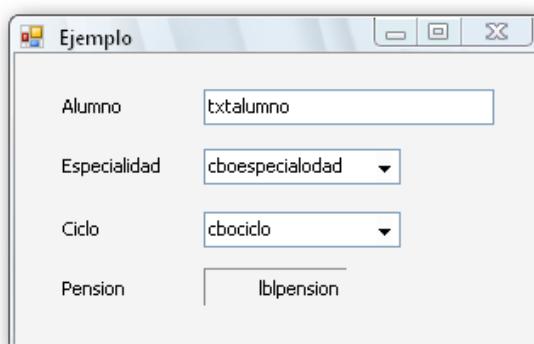


En la ejecución del programa también se puede activar cualquiera de las dos opciones.

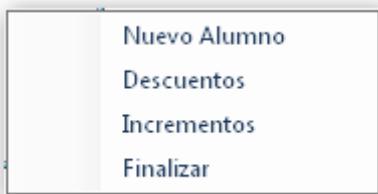


Las pensiones del alumno dependen de la especialidad y son las siguientes:

Controles del formulario



El menú contextual tiene las siguientes opciones:



El menú contextual debe tener en su propiedad ShowCheckMargin el valor True para que muestre la barra donde se indica una opción está activada.

ShowCheckMargin	True
-----------------	-------------

La opción Descuentos e Incrementos deben tener el valor True en su propiedad CheckOnClick para que muestren su casilla de verificación para saber si el usuario las ha activado o no.

CheckOnClick	True
--------------	-------------

Al control TxtAlumno asígnele el valor 50 en su propiedad MaxLength para controlar su cantidad de caracteres que se deben ingresar.

Al control CboEspecialidad asígnele el valor True en su propiedad Sorted para que las especialidades se muestren ordenadas alfabéticamente.

Al control LblPension debe tener las siguientes propiedades:

AutoSize	False	BorderStyle	Fixed3D
.TextAlign	MiddleRight		

El formulario debe tener en su propiedad ContextMenuStrip el nombre del menú contextual para que se muestre al hacer clic derecho en cualquier parte libre de:

ContextMenuStrip	ContextMenuStrip1
------------------	--------------------------

Antes de escribir las instrucciones de los controles, debe crear el siguiente procedimiento:

Procedimiento Resultados

Johan Guerreros Montoya

Este procedimiento muestra la pension del alumno según la especialidad y ciclo, y según la configuración de las opciones, es decir, si tiene descuento en el primer ciclo o incremento a partir del tercer ciclo:

```

Sub resultados()
    'verifica que se haya seleccionado la especialidad
    If cbociclo.SelectedIndex = -1 Then
        lblpencion.Text = String.Empty
        ExitSub
    EndIf
    'verifica q se haya seleccionado el ciclo
    If cboespecialidad.SelectedIndex = -1 Then
        lblpencion.Text = String.Empty
        ExitSub
    EndIf

    Dim ciclo AsByte
    Dim especialidad AsString
    Dim pension AsSingle
    'almacena la especialidad de alumnos
    especialidad = cbociclo.Text
    'almacena el ciclo del alumno se le suma 1 a la propiedad selectindex
    ciclo = cboespecialidad.SelectedIndex + 1
    'establece la pension del alumno segun su especialidad
    SelectCase especialidad
        Case "computacion"
            pension = 150
        Case "administracion"
            pension = 140
        Case "idiomas"
            pension = 100
        Case "marketing"
            pension = 120
        CaseElse
            pension = 0
    EndSelect
EndSub

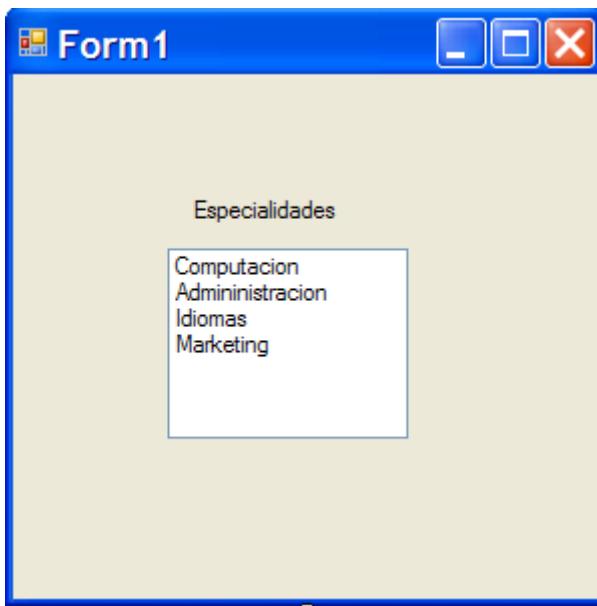
```

El Control ListBox



Este control permite ingresar datos seleccionandolos desde una lista de elementos similar al ComboBox con la diferencia que no es necesario hacer clic en la flecha hacia abajo ni pulsar F4 para mostrar los elementos.

Ejemplo:



Sus principales propiedades son:

Name

Esta propiedad se utiliza para asignarle un nombre particular al control. Se recomienda que empieze con las iniciales Lst o Lb.

ColumnWidth

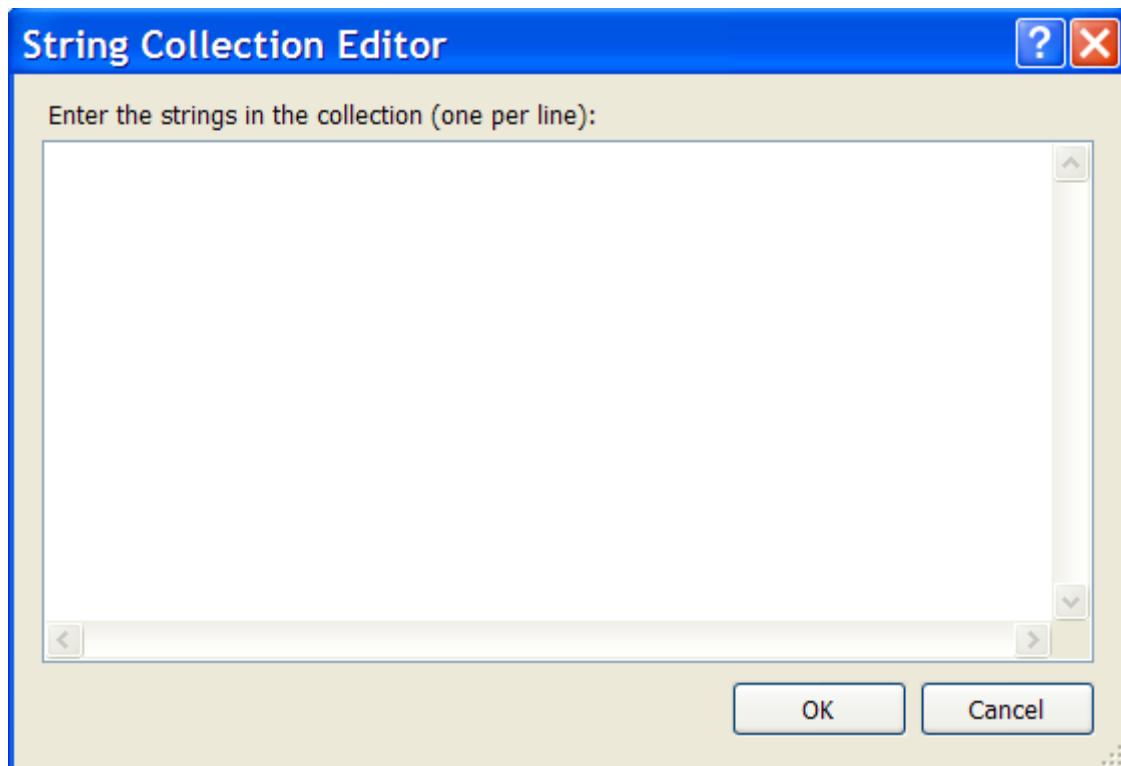
Esta propiedad se utiliza cuando el control ListBox esta configurado para mostrar los elementos en varias columnas y permite indicar el ancho de cada columna.

HorizontalScrollBar

Esta propiedad se utiliza para indicar si el control ListBox debe mostrar un abarra de desplazamiento horizontal cuando no se pueda visualizar el texto o los elementos que se encuentran a la derecha del control.

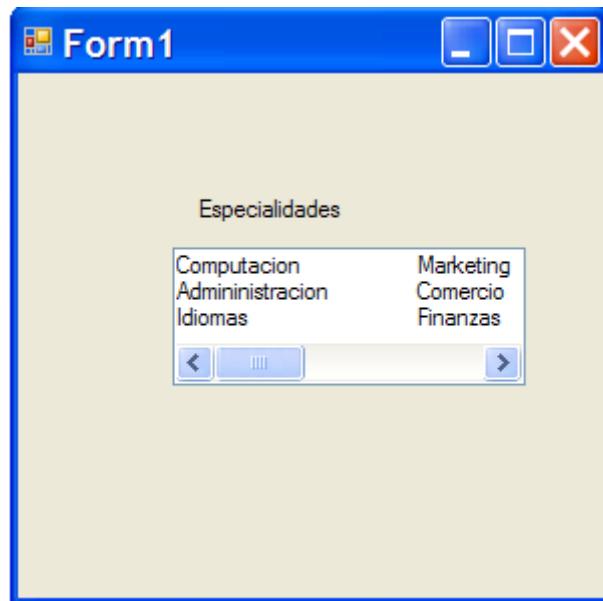
Items

Esta propiedad se utiliza para escribir los elementos que el control ListBox debe mostrar. Al ingresar a esta propiedad se muestra la siguiente ventana donde debe escribir los elementos y para finalizar haga clic en Aceptar.



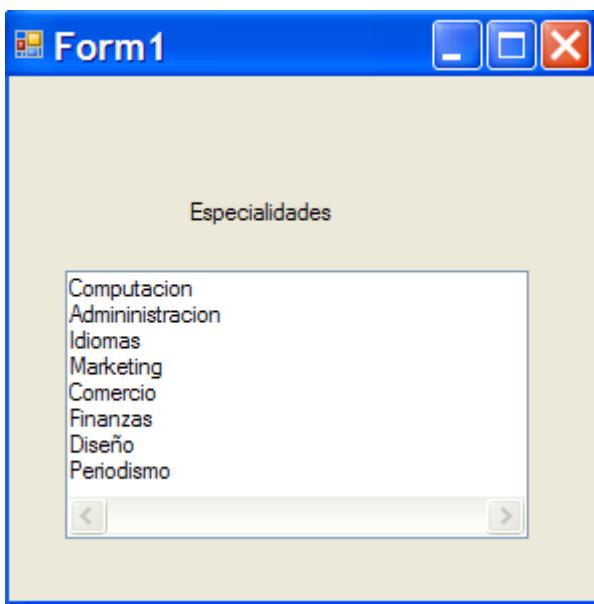
MultiColumn

Esta propiedad se utiliza para indicar si los elementos que muestra el control ListBox debe mostrarse ocupando mas de una columna.



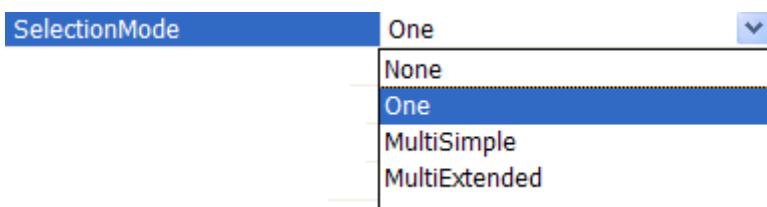
ScrollAlwaysVisible

Esta propiedad permite indicar si el control ListBox debe mostrar siempre una barra de desplazamiento horizontal.



SelectionMode

Esta propiedad se utiliza para indicar como se pueden seleccionar los elementos que muestra el control ListBox . Las opciones que tiene esta propiedad son:



La opción None no permite seleccionar ningún elemento del control ListBox.

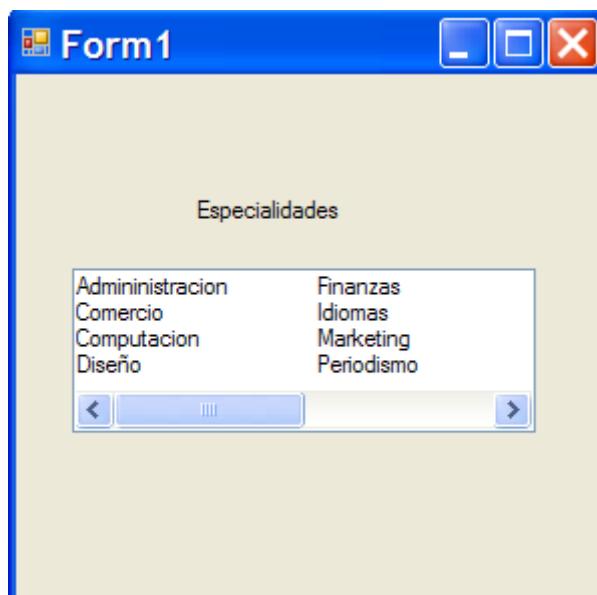
La opción One solo permite seleccionar un elemento del control ListBox.

La opción MultiSelect permite seleccionar varios elementos del control ListBox haciendo clic en cada uno de ellos.

La opción MultiExtend permite seleccionar varios elementos del control ListBox utilizando la tecla Ctrl. O Shift.

Sorted

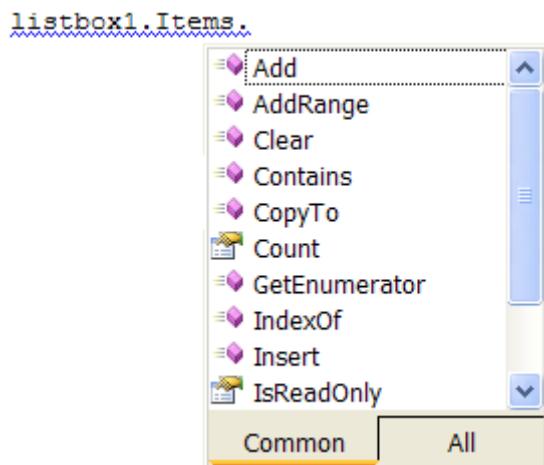
Esta propiedad permite indicar si el control ListBox debe mostrar los elementos ordenados.



Items

Esta propiedad también se puede utilizar mediante código, es decir, mediante instrucciones donde representa a todos los elementos que contiene el control. Esta propiedad también la tiene el ComboBox.

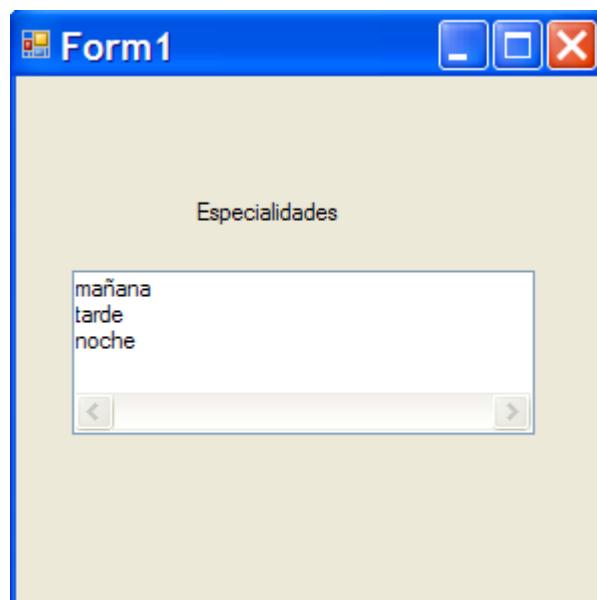
La propiedad Items tiene varias funciones que permiten administrar los elementos que contiene el control. Ejemplo:



Add: Permite agregar elementos al control ListBox y ComboBox. Ejemplo: las siguientes instrucciones agregan tres elementos al control ListBox.

```
listbox1.Items.Add("mañana")
```

```
listbox1.Items.Add("tarde")
```



```
listbox1.Items.Add("noche")
```

Clear: Elimina todos los elementos que contiene el control ListBox o ComboBox. Ejemplo: la siguiente instrucción alimina todos los elementos del control ListBox1.

```
ListBox1.Items.Clear()
```

Count: Devuelve la cantidad de elementos que contiene el control ListBox o ComboBox. Ejemplo : las siguientes instrucciones muestran en el control LblCantidad la cantidad de elementos que tiene el control ListBox1.

Dim n AsInteger

'declara la variable n tipo entero

```
n = listbox1.Items.Count
```

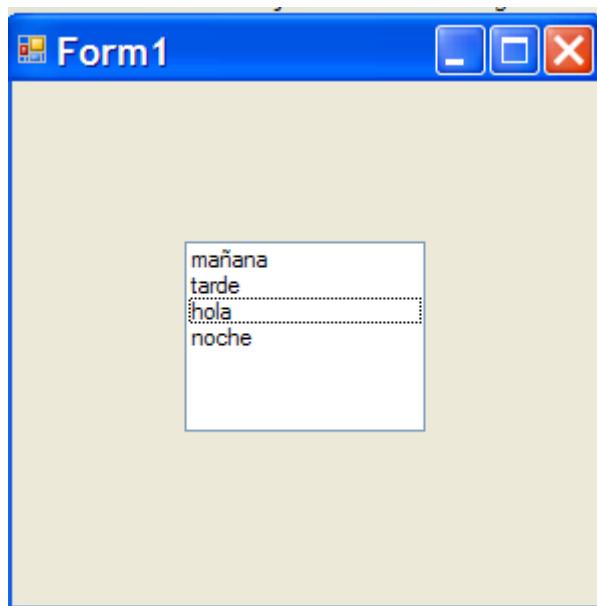
'cuenta los elementos que hay en lb

```
lblcantidad.Text = "son:" & n & "elementos"
```



Insert: Inserta un nuevo elemento en el control ListBox o ComboBox. Se debe indicar la posición y el elemento a insertar. La primera posición es 0. Ejemplo: la siguiente instrucción inserta la palabra HOLA después de la palabra tarde.

```
ListBox1.Items.Insert(2, "hola")
```

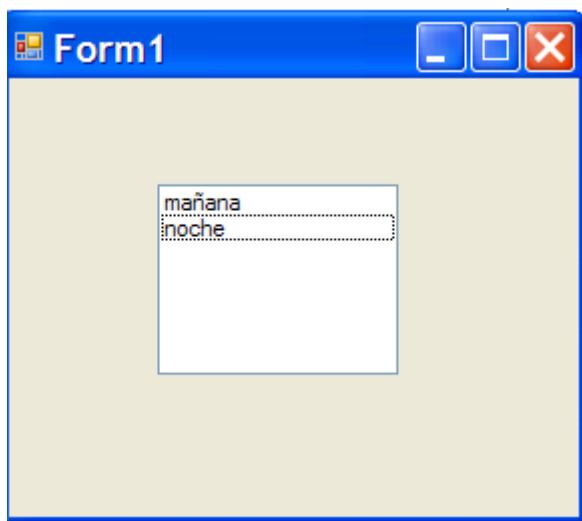


Remove: Permite eliminar un elemento del control . Se le debe enviar como parámetro el texto del elemento a eliminar. Por ejemplo, la siguiente instrucción elimina el elemento Tarde .

```
ListBox1.Items.Remove("tarde")
```

Remove At: Permite eliminar un elemento del control. Se le debe enviar como parametro el numero de elemento a eliminar. El primer elemento tiene el valor cero(0). Por ejemplo, la siguiente instrucción elimina el elemnto Tarde.

```
ListBox1.Items.RemoveAt(1)
```



SelectedIndex

Esta propiedad devuelve el numero del elemnto seleccionado. El primer elemnto es mcero(0). Ejemplo: las sigueitnes instrucciones muestran en el control LblCantidad el numero del elemnto seleccionado del control ListBox1.

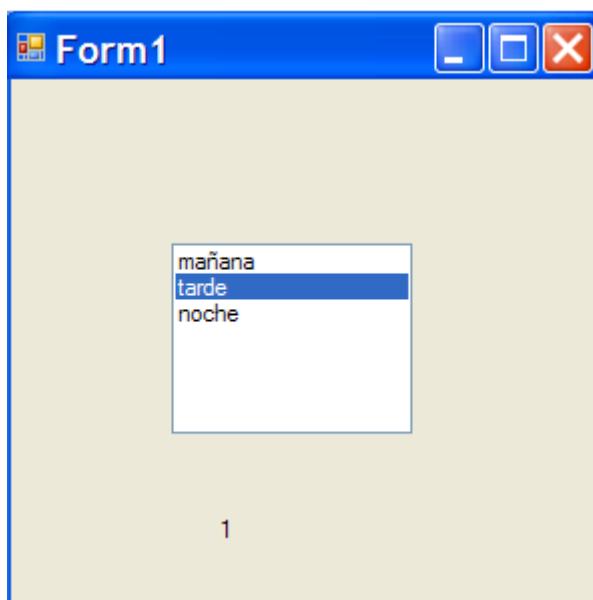
Dim n AsInteger

'declara la variable n tipo entero

n = ListBox1.SelectedIndex

'asignandole un apropiedad que de valor con el zsslectedindex

Iblcantidad.text = n.ToString



SelectedItem

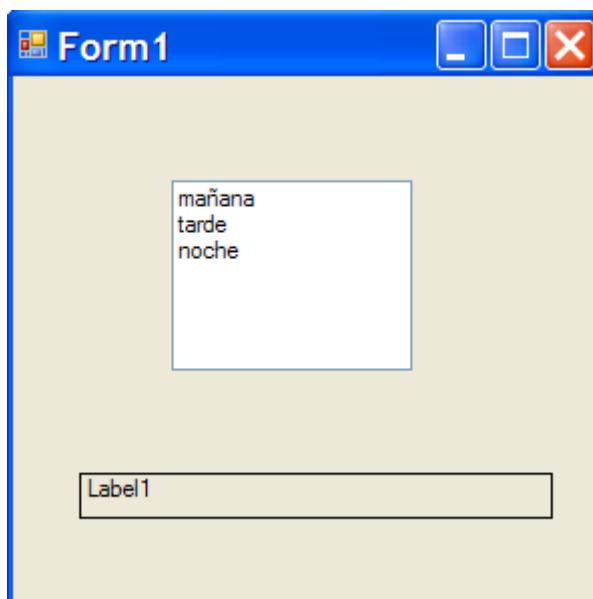
Esta propiedad devuelve el numero del elemento seleccionado. El primer elemento es cero(0). Ejemplo: las siguientes instrucciones muestran en el control LblCantidad el texto del elemento seleccionado del control ListBox1.

Dim l AsString

```
l = ListBox1.SelectedItem
```

'devuelve el numero del emlemento seleccionado con esta propeidad selecteditem

```
lblcantidad.Text = "el elemneto seleccionado es:" & l
```



ClearSelected

Es un método que desactiva el elemento seleccionado de un control ListBox, es decir , al ejecutar este método, ningún elemento del control ListBox esta seleccionado. Ejemplo, la siguiente instrucción desactiva el elemento seleccionado del control ListBox1.

```
ListBox1.ClearSelected()
```

Findstring

Es un metodo que permite buscar un elemento dentro del control ListBox o ComboBox. Este control devuelve el numero del elemento seleccionado empezando con cero(0) para el primer elemento. Si el elemento no se encuentra devuelve -1. La siguientes instrucciones buscan el elemento Tarde:

Dim l AsInteger

```
l = ListBox1.FindString("tarde")
```

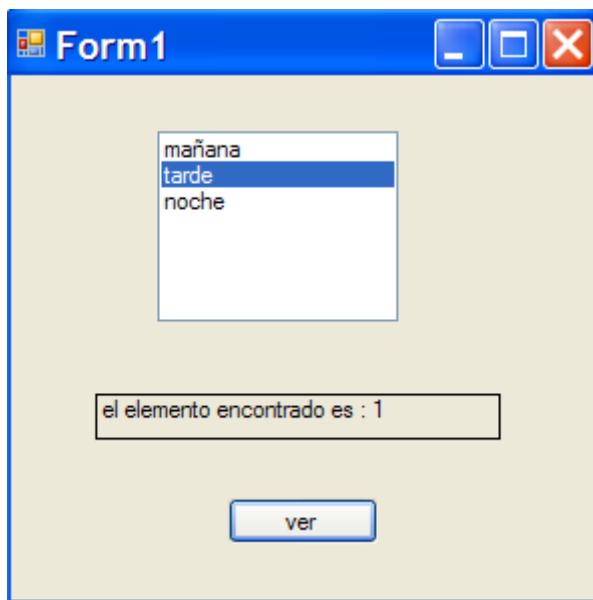
Con este metodo se puede encontrar un elemento sin escribir el texto exacto, las siguientes instrucciones tambien buscan y encuentran el elemento tarde:

Dim l AsInteger

```
l = ListBox1.FindString("tar")
```

'usamos findstring para encontrar el elemnto

```
lblcantidad.Text = "el elemento encontrado es : "& 1
```



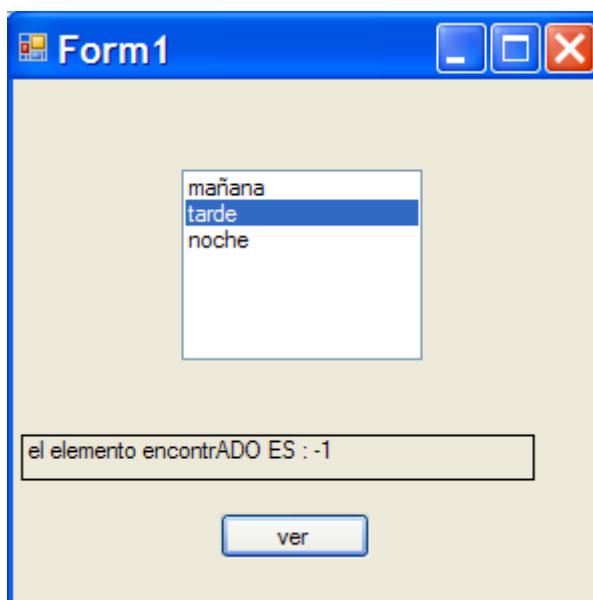
FindStringExact

Este metodo permite buscar un elemnto dentro del control ListBox o ComboBox, pero busca el texto exacto. Por ejemplo, SI EN EL EJEMPLO ANTERIOR USAMOS ESTE EMTODO, EL RESULTADO SERA -1.

```
Dim 1 AsInteger
```

```
1 = ListBox1.FindStringExact("tar")
```

```
lblcantidad.Text = "el elemento encontrADO ES : "& 1
```



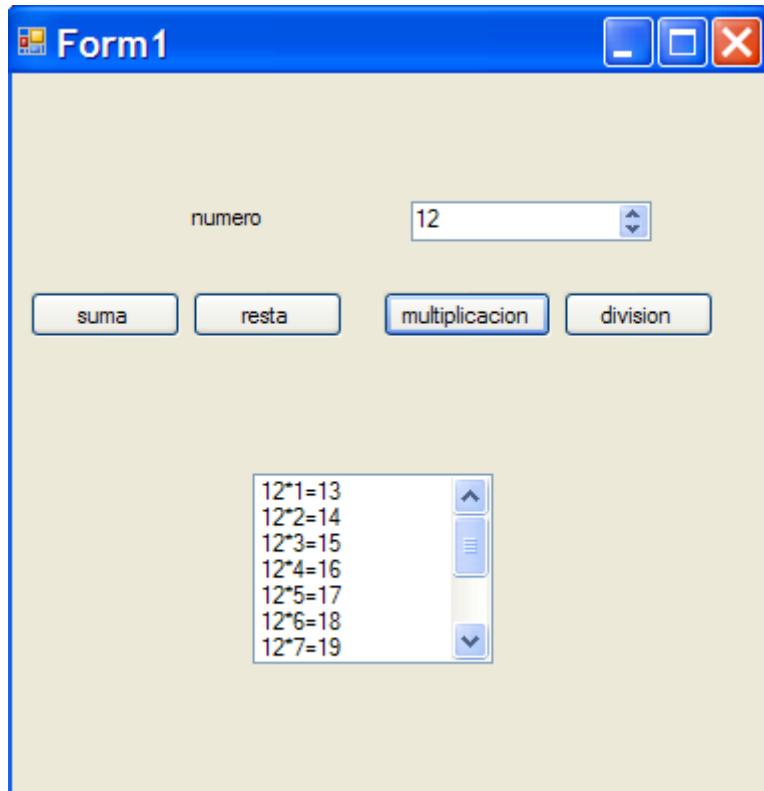
GetSelected

Este metodo permite saber si un elemento esta seleccionado o no. Despues el valor True o False. Por ejemplo, la siguiente instrucción pregunta si el elemnto nro 1 del control ListBox1 esta seleccionado:

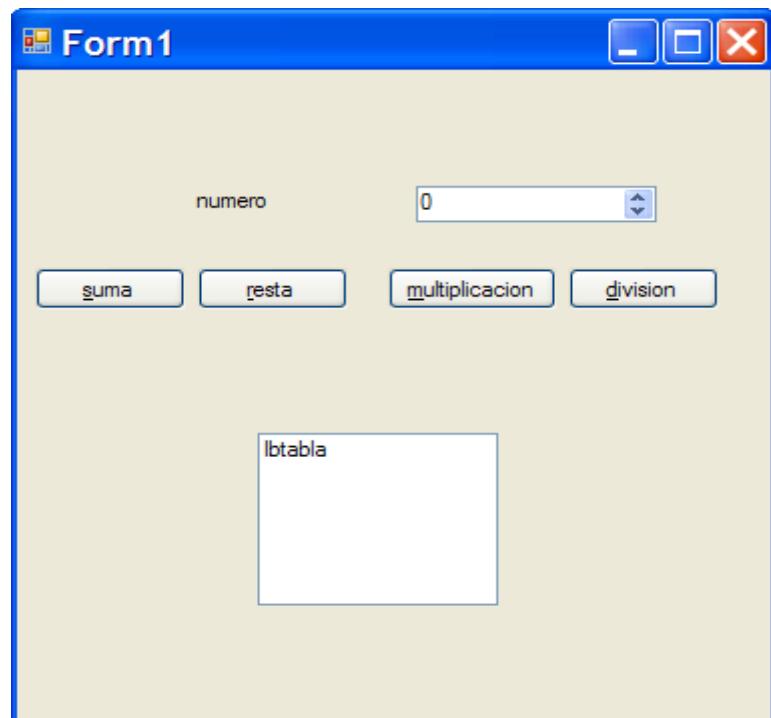
```
If ListBox.GetSelected(1)=True Then
```

Aplicación desarrollada nro IV-05

Este programa permite seleccionar un numero de un control NumericUpDown y mostrar su tabla de suma, resta, multiplicacion o division.



Para desarrollar este programa, debe dibujar un control NumericUpDown, cuatro botones de comandos y un ListBox.



```

PrivateSub btns_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btns.Click

    'se declara als vairables tipo entero

    Dim n, i, r AsInteger

    'almacena el numero ingresado

    n = nudnumero.Value

    'se digita un nuemro

    'limpia los elementos del control ListBox

    ltabla.Items.Clear()

    'muestra la tabla

    For i = 1 To 12

        r = n + i

        'este es un contador

        ltabla.Items.Add(n & "+" & i & "=" & r)

    Next

EndSub

```

```

PrivateSub btnr_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnr.Click

    Dim n, i, r AsInteger

    'almacena el numero ingresado

    n = nudnumero.Value

    'limpia los elementos del control ListBox

    ltabla.Items.Clear()

    'muestra la tabla

    For i = 1 To 12

        r = n + i

        'este es un contador

        ltabla.Items.Add(n & "-" & i & "=" & r)

    Next

```

```
EndSub
```

```
PrivateSub btmn_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btmn.Click
```

```
Dim n, i, r AsInteger
```

'almacena el numero ingresado

```
n = nudnumero.Value
```

'limpia los elementos del control ListBox

```
ltabla.Items.Clear()
```

'muestra la tabla

```
For i = 1 To 12
```

```
r = n + i
```

'este es un contador

```
ltabla.Items.Add(n &"*"& i &"="& r)
```

```
Next
```

```
EndSub
```

```
PrivateSub btnd_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnd.Click
```

```
Dim n, i, r AsInteger
```

'almacena el numero ingresado

```
n = nudnumero.Value
```

'limpia los elementos del control ListBox

```
ltabla.Items.Clear()
```

'muestra la tabla

```
For i = 1 To 12
```

```
r = n + i
```

'este es un contador

```
ltabla.Items.Add(n &"/"& i &"="& r.ToString("##,##0.00"))
```

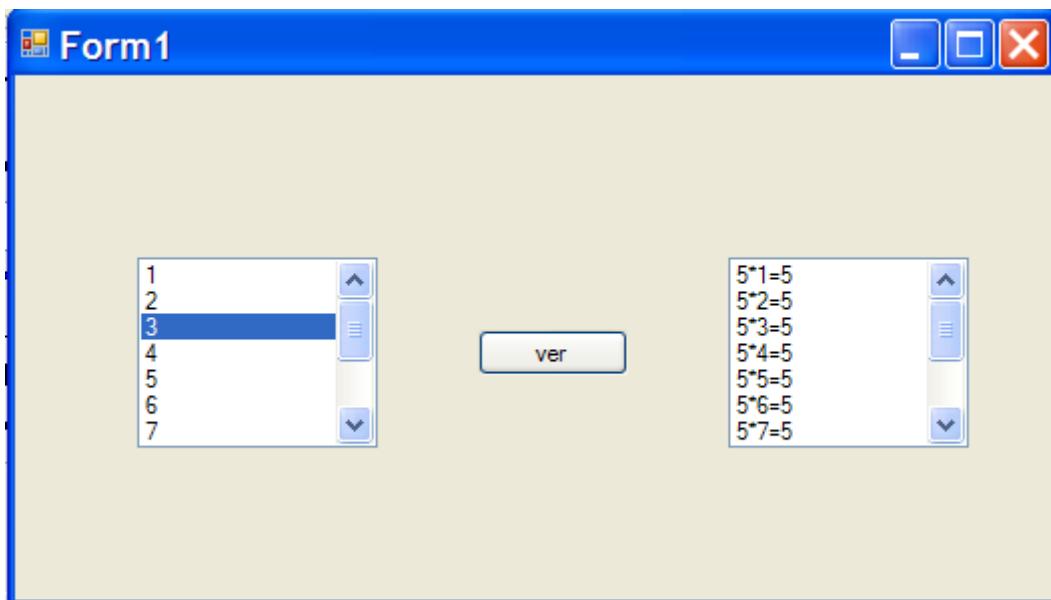
```
Next
```

Johan Guerreros Montoya

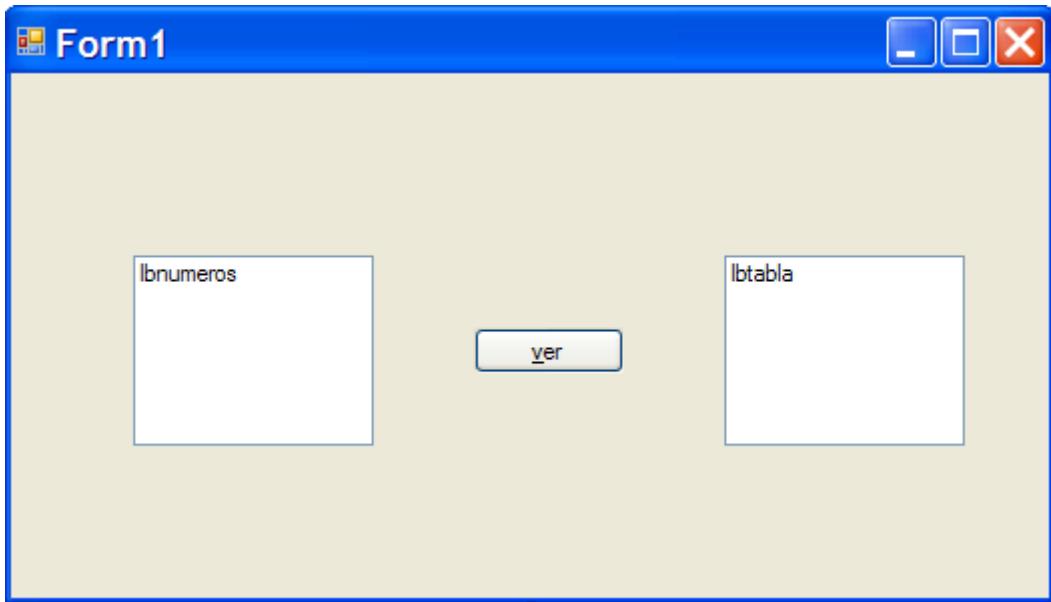
EndSub

Aplicación Desarrollada nro IV-06

Este programa permite seleccionar uno o mas numeros de un control ListBox y mostrar su tabla de multiplicar en otro control ListBox configurado para mostrar los resultados en varias columnas.



Para este programa solo debe de dibujar dos controles ListBox y un Boton.



Al control LblNumeros debe asignarle el valor MultiSimple en su propiedad SelectionMode. Este valor permitira que el usuario pueda seleccionar varios numeros.

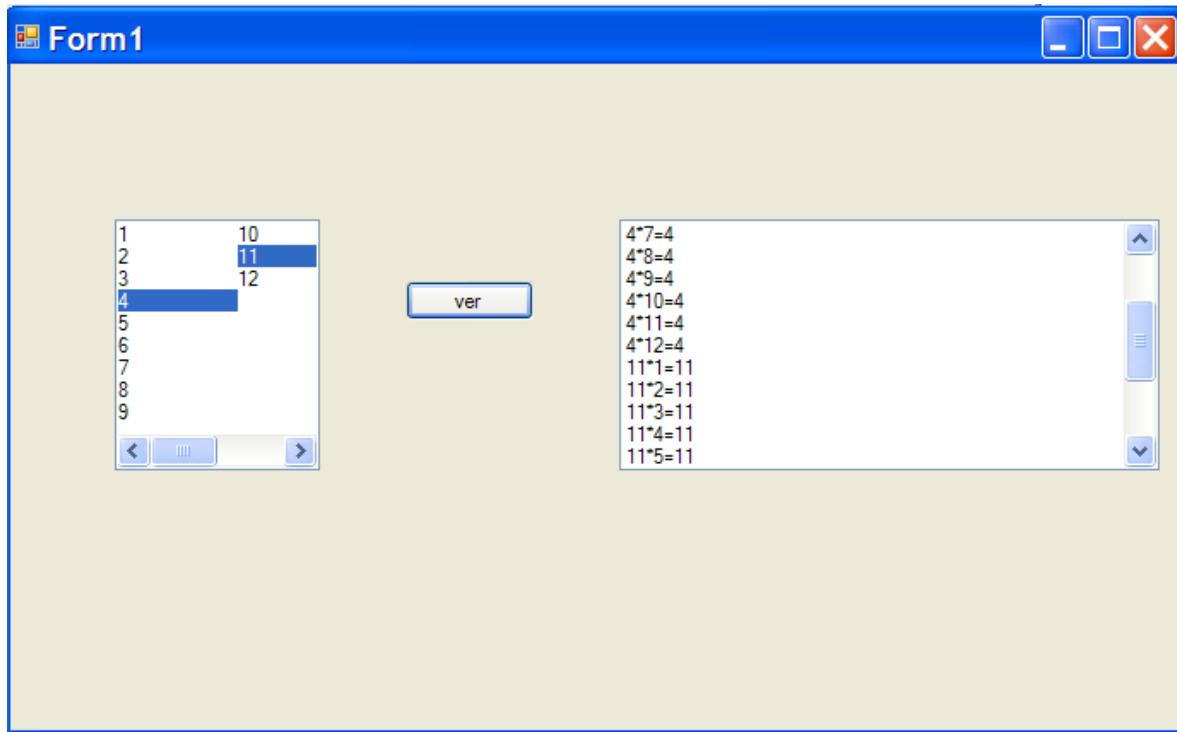
SelectionMode **MultiSimple**

Al control LblTabla le debe asignar las siguientes propiedades:

Johan Guerreros Montoya

ColumnWidth	70
MultiColumn	True
ScrollAlwaysVisible	True

Al control LblTabla lo debe dibujar con un tamaño que permita mostrar en una columna, una tabla de multiplicar diferente.



Instrucciones del evento Load del Formulario:

Estas instrucciones muestran en el control LblNumeros, los numeros del 1 al 12.

```
Dim N AsInteger
```

```
For N = 1 To 12
```

‘se hace una tabla

```
Ibnumeros.Items.Add(N.ToString)
```

```
Next
```

1
2
3
4
5
6
7
8
9
10
11
12

Instrucciones del botón btnver:

```
Dim N, I, R, Numero AsInteger
```

Johan Guerreros Montoya

'limpia el control lb tabla

```
Ibtabla.Items.Clear()
```

```
For N = 0 To 11
```

'pregunta si el elemento ha sido seleccionado

```
If Ibnumeros.GetSelected(N) Then
```

'muestra la tabla del numero seleccionado

```
Numero = N + 1
```

```
For I = 1 To 12
```

'este es un contador

R = Numero

```
Ibtabla.Items.Add(Numero &"*"& I &"="& R)
```

```
Next
```

```
EndIf
```

```
Next
```

Aplicación Desarrollada nro IV-07

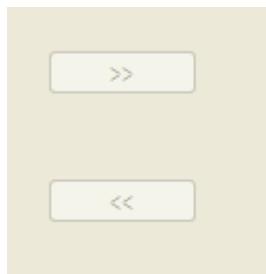
Este programa permite seleccionar valores de un control ListBox Origen y pasarlos a otro ListBox Destino.



Cuando los valores pasan de un ListBox a otro se eliminan del origen. En el ejemplo anterior se han eliminado los números 5,8 y 10.

El primer botón pasa los valores del ListBox origen al destino y el segundo botón hace lo contrario.

Mientras no se seleccione ningún elemento, los dos primeros botones permanecen desactivados:



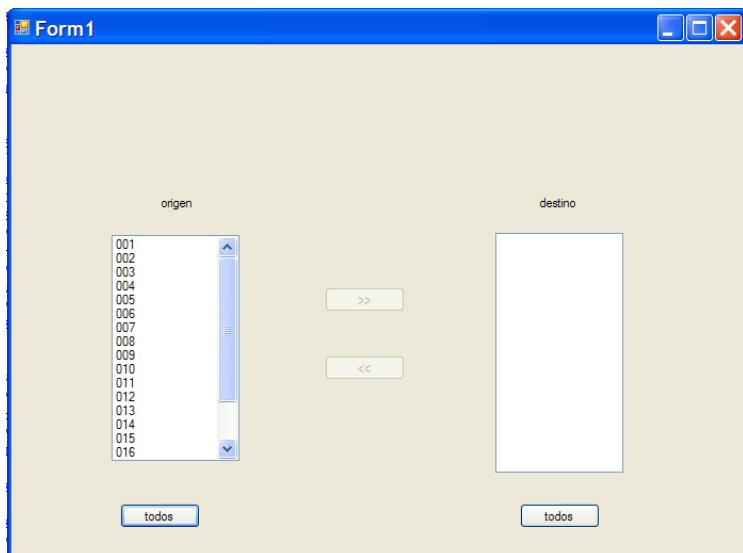
El botón que se activa depende si se han seleccionado elementos del ListBox origen o destino.

Los botones que tienen el título Todos seleccionan o quitan la selección a todos los elementos de su control ListBox respectivo.

En el siguiente ejemplo se ha hecho clic en el botón Todos del ListBox origen

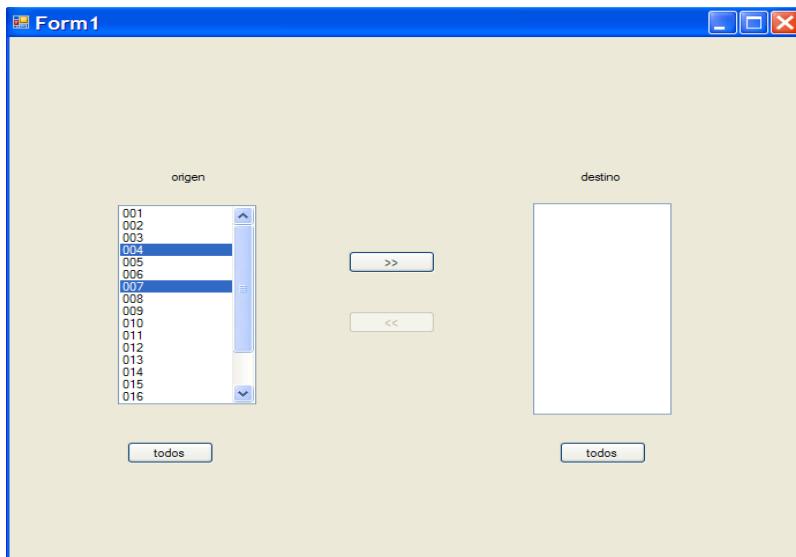


En el siguiente ejemplo se ha hecho clic por segunda vez en el botón Todos del ListBox origen.

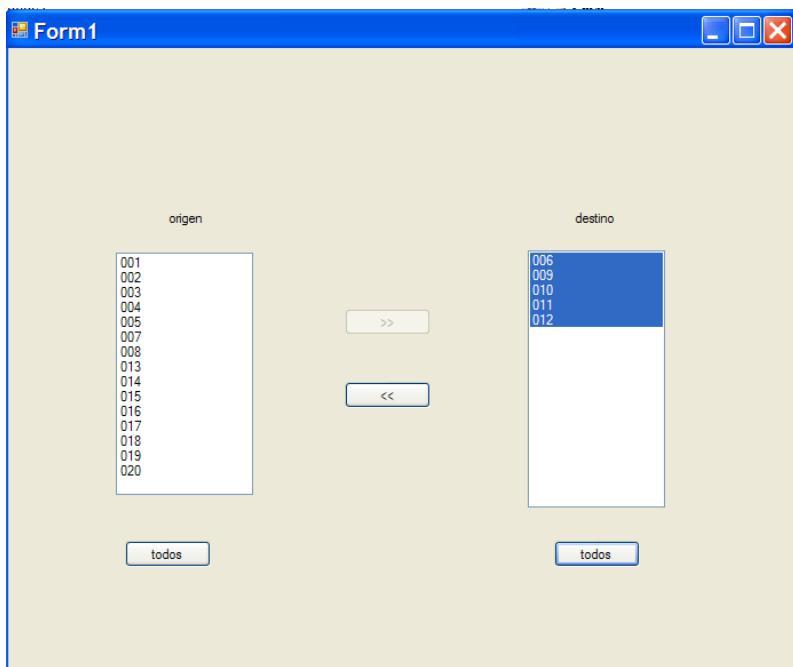


Los dos primeros botones funcionan como inteligentes, porque solo se Activan cuando se elige por lo menos un elemento de su control ListBox.

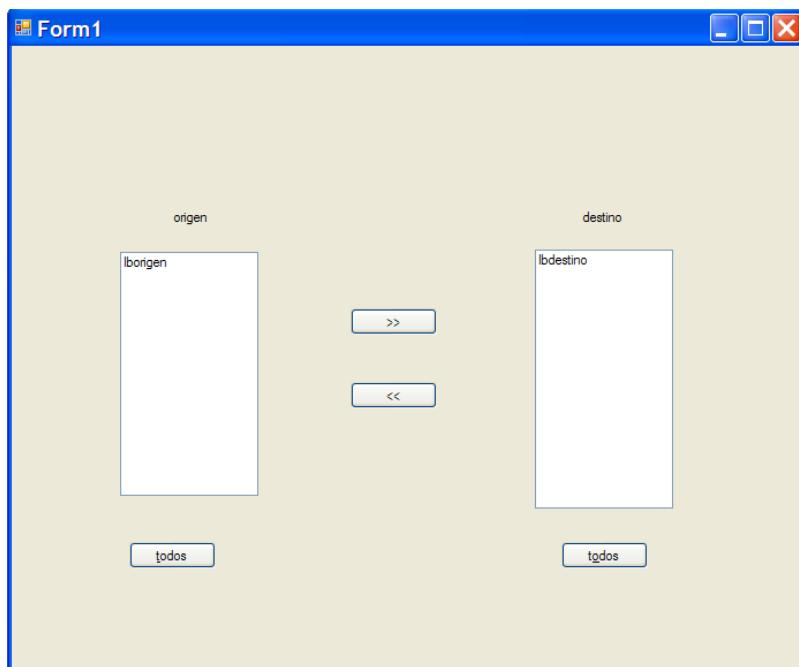
En el siguiente ejemplo se han activado dos elementos del ListBox origen por lo que se ha activado solo el primer botón.



En el siguiente ejemplo se ha activado un elemnto del ListBox destino, por lo que se ha activado solo el segundo botón.



Controles del formulario



Los controles ListBox deben mostrar los números ordenados y deben permitir seleccionar varios elementos por lo tanto, se les debe asignar las siguientes propiedades.

SelectionMode	MultiSimple
Sorted	True

Instrucciones del evento Load del formulario

Estas instrucciones llenan el control lborigen con 20 numeros.

Dim i AsShort

'limpia el contenido de los ListBox

```
Iborigen.Items.Clear()
```

```
Ibdestino.Items.Clear()
```

'agrega 20 numero al control LbOrigen

```
For i = 1 To 20
```

```
    Iborigen.Items.Add(i.ToString("000"))
```

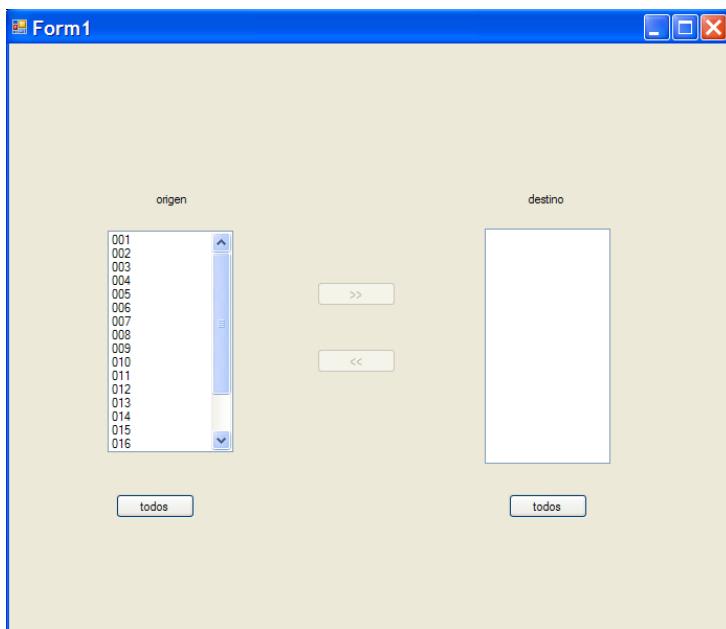
Next

'desactiva los botones btnenviar y btnrecibir

```
btnenviar.Enabled = False
```

```
btnrecibir.Enabled = False
```

Con las instrucciones del evento load el aspecto inicial del formulario.



Instrucciones del evento Click del control Lborigen.

'activa el boton btnenviar si hay elementos seleccionados

```
If Iborigen.SelectedItems.Count > 0 Then
```

```
    btnenviar.Enabled = True
```

Else

```
    btnenviar.Enabled = False
```

EndIf

'desacativa el boton btnrecibir

btnrecibir.Enabled = False

Instrucciones del evento click del control Lbdestino.

'activa el boton btnenviar si hay elementos seleccionados

If lbdestino.SelectedItems.Count > 0 Then

 btnenviar.Enabled = True

Else

 btnenviar.Enabled = False

EndIf

'desacativa el boton btnrecibir

btnrecibir.Enabled = False

Instrucciones del boton btnenviar

Dim i, n AsShort

Dim elementoAsString

'lee la cantidad de elementos

n = lborigen.Items.Count - 1

For i = 0 To n

If lborigen.GetSelected(i) = TrueThen

'lee el elemento seleccionado

 elemento = lborigen.Items(i)

'agrega el elemento seleccionado

 lbdestino.Items.Add(elemento)

EndIf

Next

'elimikna los elementos enviados con un for descendente

For i = n To 0 Step -1

If lborigen.GetSelected(i) = TrueThen

Johan Guerreros Montoya

```
'eliminma el elemnto
    lborigen.Items.RemoveAt(i)
```

EndIf

Next

'desactiva el boton

```
btnenviar.Enabled = False
```

Ejemplo del resultado de las instrucciones



Instrucciones del boton btnrecibir

```
Dim i, n AsShort
```

```
Dim elemento AsString
```

'lee la cantidad de elementos

```
n = lbdestino.Items.Count - 1
```

```
For i = 0 To n
```

```
If lbdestino.GetSelected(i) = TrueThen
```

'lee el elemento seleccionado

```
elemento = lbdestino.Items(i)
```

'agrega el elemento seleccionado

```
lborigen.Items.Add(elemento)
```

Johan Guerreros Montoya

```

EndIf

Next

'elimina los elementos enviados con un for descendente

For i = n To 0 Step -1

If lbdestino.GetSelected(i) = TrueThen

'eliminma el elemnto

lbdestino.Items.RemoveAt(i)

```

```

EndIf

Next

'desactiva el boton

btnrecibir.Enabled = False

```

Instrucciones del boton btntodos1

```

Dim i, n AsShort

Static valor AsBoolean

'lee la cantidad de elemntos

n = lborigen.Items.Count - 1

'cambia entre el valor true y false

valor = Not valor

'selecciona o quita la seleccion a todos los elementos

For i = 0 To n

lborigen.SetSelected(i, valor)

```

```

Next

'desactiva el boton btnrecibir

btnrecibir.Enabled = False

'si hay elementos seleccionados se activa el boton btnenviar

If lborigen.SelectedItems.Count > 0 Then

btngeniar.Enabled = True

```

```

Else
    btnenviar.Enabled = False
EndIf

'se ubica en el primer elemento
lborigen.TopIndex = 0

Instrucciones del boton btntodos2
Dim i, n AsShort
Static valor AsBoolean
'Tee la cantidad de elemntos
n = lbdestino.Items.Count - 1
'cambia entre el valor true y false
valor = Not valor
'selecciona o quita la seleccion a todos los elementos
For i = 0 To n
    lbdestino.SetSelected(i, valor)

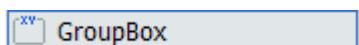
Next
'desactiva el boton btnrecibir
btnenviar.Enabled = False
'si hay elementos seleccionados se activa el boton btnenviar
If lbdestino.SelectedItems.Count > 0 Then
    btnrecibir.Enabled = True
Else
    btnrecibir.Enabled = False
EndIf

'se ubica en el primer elemento
lbdestino.TopIndex = 0

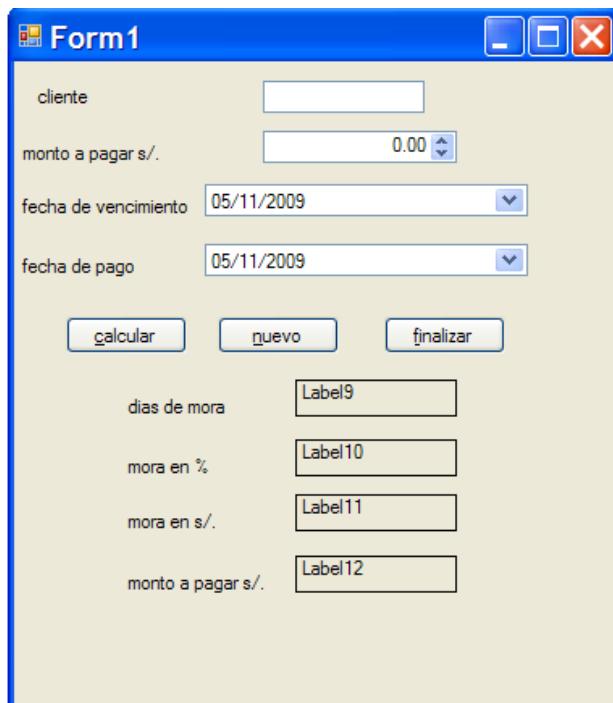
```

El control GroupBox

Johan Guerreros Montoya



Este control permite agrupar controles o mejorar el aspecto de un formulario. Con este control, el ejemplo II-12 puede quedar de la siguiente manera:

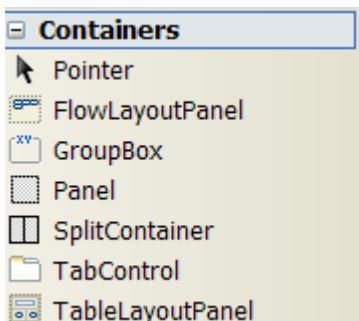


Los controles que se encuentran dentro de un GroupBox se pueden manejar como uno solo.

Por ejemplo, si queremos desactivar los tres botones y el control que la contiene se llama GroupBox2, podemos utilizar la siguiente instrucción:

```
GroupBox2.Enabled=False
```

El control GroupBox se encuentra en el panel Contenedores del cuadro de herramientas:



Sus principales propiedades son:

AutoSize

Permite indicar si el tamaño del control depende del espacio que se utiliza en su contenido. Autosize trabaja junto con la propiedad AutosizeMode.

AutoSizeMode

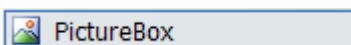
Esta propiedad permite indicar la forma como deben trabajar las propiedades de Autosize.



Text

Se utiliza para indicar el título del control GroupBox.

El Control PictureBox



Este control permite mostrar imágenes en los formularios de muestras, aplicaciones, por ejemplo:



Sus principales propiedades son:

BackGroundImage

Esta propiedad se utiliza para seleccionar la imagen que se debe mostrar como fondo del control. La forma de seleccionar un grafico se explica en la propiedad Image.

BorderStyle

Esta propiedad se utiliza para seleccionar un borde para el control PictureBox.

ErrorImage

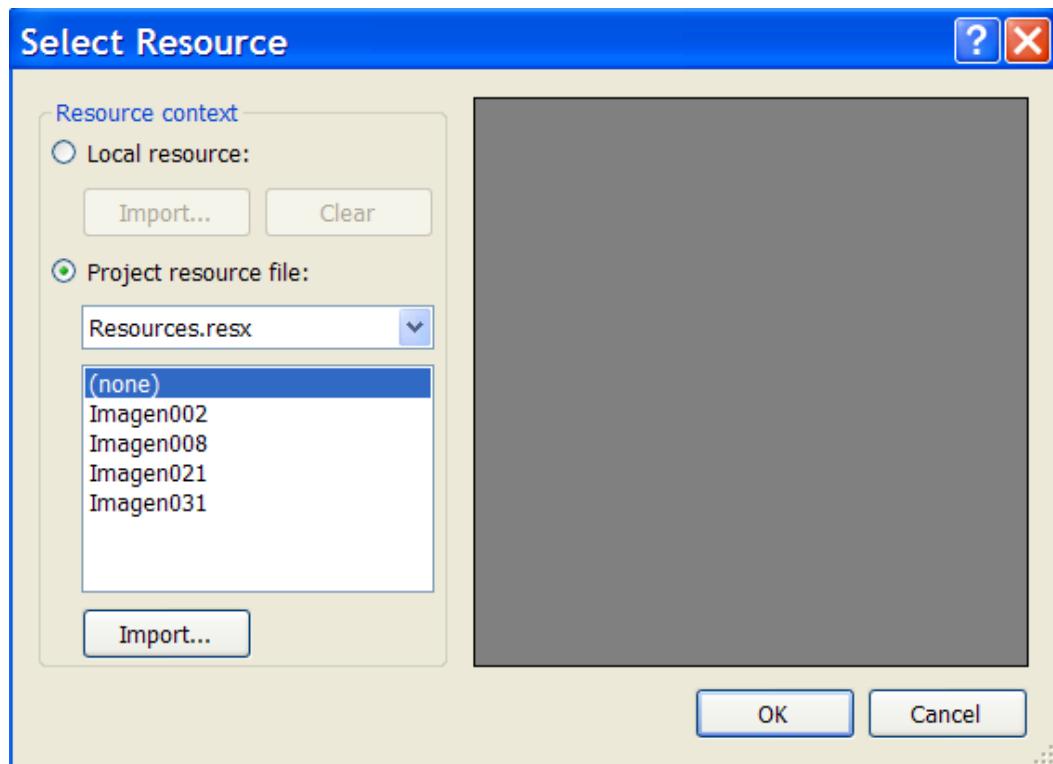
Esta propiedad se utiliza para seleccionar la imagen que se debe mostrar cuando existe un error en la imagen seleccionada en la propiedad Image.

Image

Esta propiedad se utiliza para seleccionar la imagen que desea mostrar en el control.

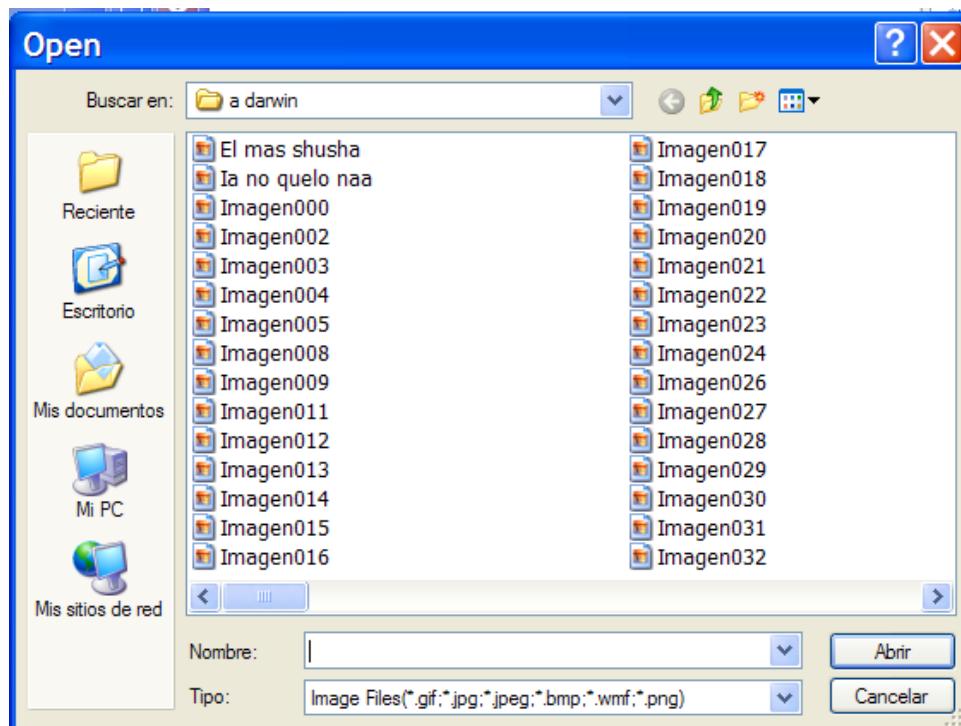


Al ingresar a esta propiedad se visualiza la siguiente ventana:

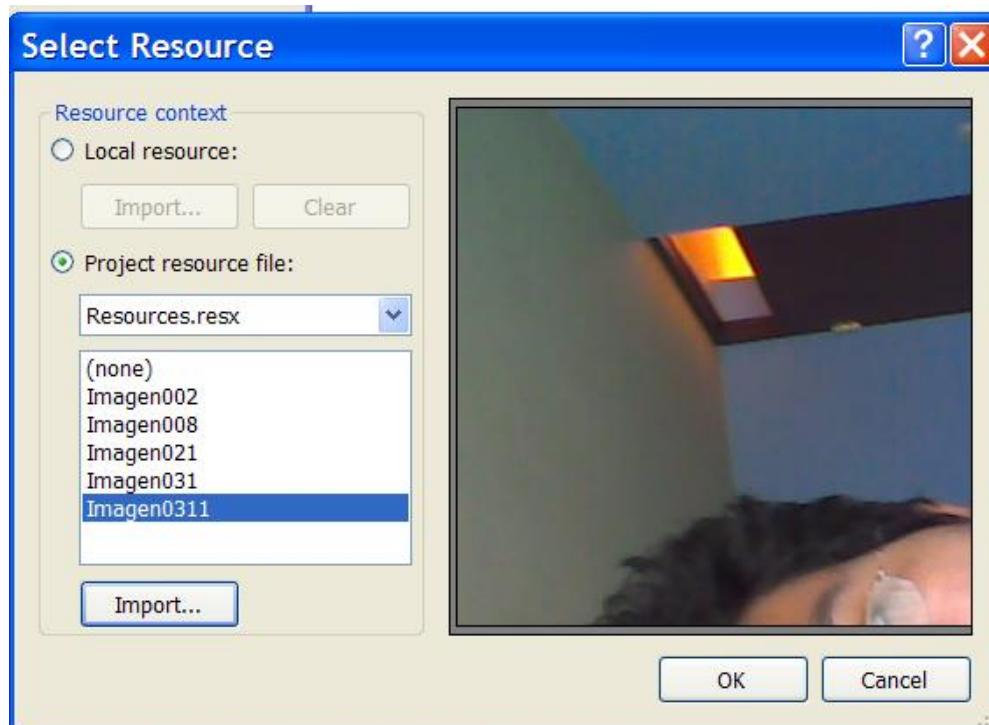


Si desea que las imágenes formen parte de la aplicación, elija Archivo de recursos del proyecto, de lo contrario elija Recurso local. Para seleccionar la imagen haga clic en el botón importar.

Al hacer clic en el botón importar se visualiza la siguiente ventana, donde se debe seleccionar la siguiente carpeta y el archivo que contiene la imagen. Para terminar haga clic en el botón Abrir.



Al hacer clic en el botón Abrir se visualiza la ventana anterior con la vista previa de la imagen seleccionada. Para terminar, haga clic en Aceptar.

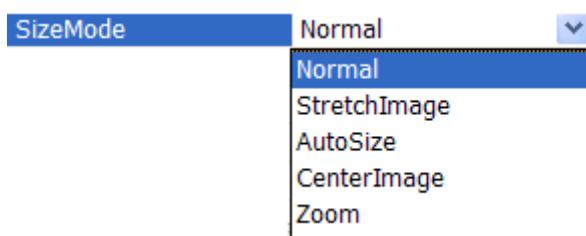


Initial Image

Esta propiedad se utiliza para seleccionar la imagen que se debe mostrar en el control mientras se carga una nueva imagen.

SizeMode

Esta proeidad se utiliza para seleccionar el modo como se debe ajustar la imagen seleccionada dentro del control. Esta proeidad tiene las sigueitnes opciones:



Normal

Esta opcion permite que la imagen se muesatre dentro del control con su tamaño normal.

StretchImage

Esta opcion permite que el tamaño de la imagen se ajuste al tamaño que se ha dibujado el control PictureBox.

AutoSize

Esta opcion permite que el tamaño del control PictureBox se adapte al tamaño de la imagen seleccioando.

CenterImage

Esta opcion permite que la imagen seleccioanda se muestre en el centro del control PictureBox.

Zoom

Esta opcion permite que se muestre toda la imagen y en el centro del control PictureBox.

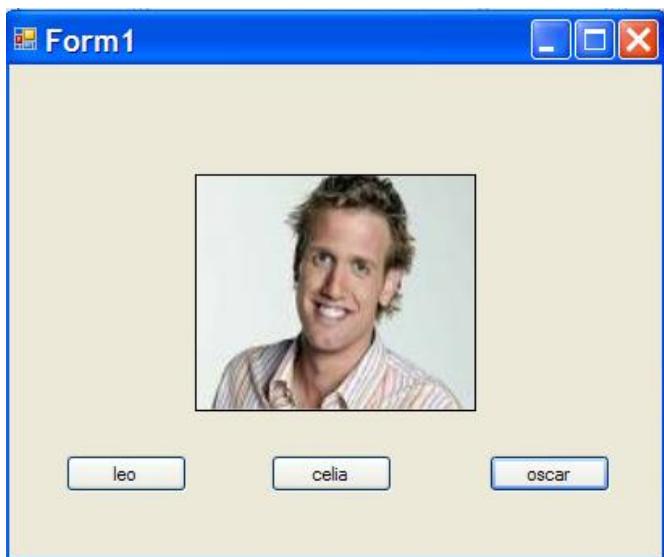
Aplicación practica nro IV-07B

Este programa permite mostrar imágenes en un control PictureBox en tiempo de ejecucion.



Cada uno de los botones muestra una imagen diferente, como se muestra a continuacion.



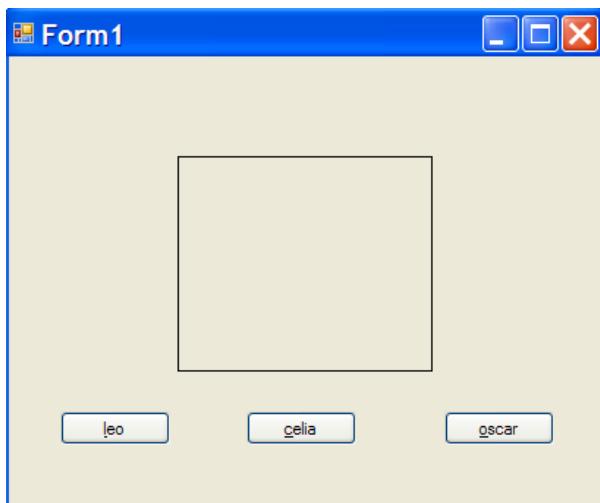


Para esta aplicación se deben tener grabadas en la unidad D:\Fotos los archivos Leo.Jpg, Celia.Jpg y Oscar.Jpg.

En esta aplicación se controlan las excepciones y en caso que ocurra, por ejemplo cuando no se muestra algun archivo, se visualiza el siguiente mensaje.



Controles del formulario.



Al control PictureBox1 asignele las siguientes propiedades.

BorderStyle

FixedSingle

SizeMode

StretchImage

Instrucciones del botón BtnLeo

Johan Guerreros Montoya

Try

```
PictureBox1.Image = New Bitmap("d:\fotos1\leo.jpg")
```

Catch ex As Exception

```
MessageBox.Show(ex.Message, ex.Source)
```

```
PictureBox1.Image = Nothing
```

EndTry

Instrucciones del boton BtnCelia

Try

```
PictureBox1.Image = New Bitmap("d:\fotos1\celia.jpg")
```

Catch ex As Exception

```
MessageBox.Show(ex.Message, ex.Source)
```

```
PictureBox1.Image = Nothing
```

EndTry

Instrucciones del boton BtnOscar

Try

```
PictureBox1.Image = New Bitmap("d:\fotos1\oscar.jpg")
```

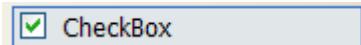
Catch ex As Exception

```
MessageBox.Show(ex.Message, ex.Source)
```

```
PictureBox1.Image = Nothing
```

EndTry

El Control CheckBox

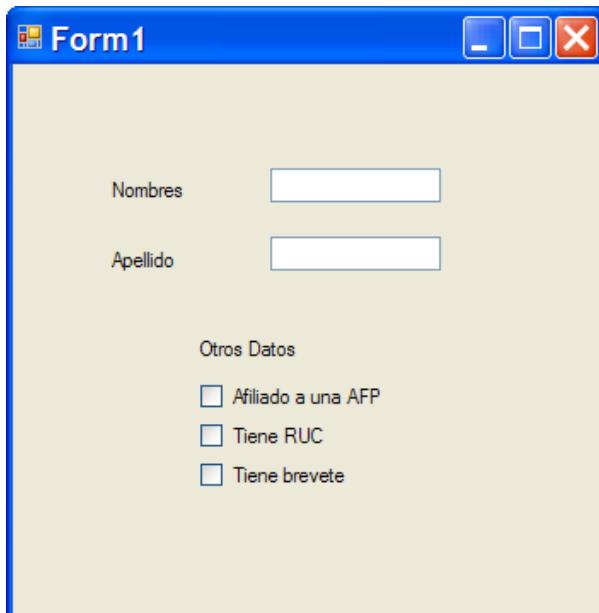


Este control permite indicar dentro de una aplicación si algo es verdadero o falso mediante una casilla de verificación que tiene este control. En el formulario se pueden utilizar varios checkboxes.

Johan Guerreros Montoya

controles CheckBox y tienen como característica principal que el usuario puede seleccionar hasta todas las casillas de verificación. Se puede dibujar dentro de un control GroupBox.

Ejemplo:



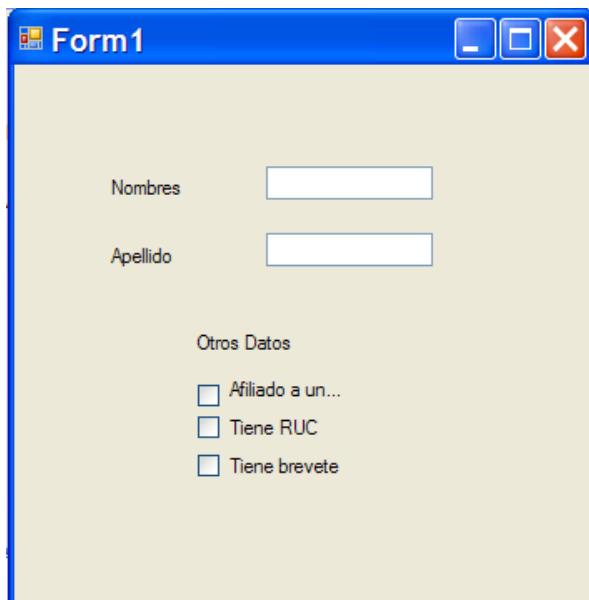
Sus principales propiedades son :

AutoCheck

Si tiene el valor True indicará que el control CheckBox debe cambiar su estado cada vez que se seleccione.

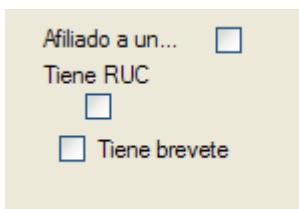
AutoEllipsis

Esta propiedad se utiliza para indicar si se debe visualizar el texto cuando el ancho del control sea menor que la cantidad de letras.



CheckAlign

Esta propiedad se utiliza para indicar la posición de la casilla de control.



Checked

Esta propiedad se utiliza para indicar si la casilla debe estar activada o desactivada.

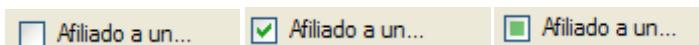
False True



CheckState

Esta propiedad se utiliza para indicar el estado de la casilla del control. Es casi similar a la propiedad anterior con la diferencia que tiene una opción más:

Unchecked Checked Indeterminate



Text

Esta propiedad se utiliza para escribir el texto que debe tener la casilla del control.

ThreeState

Esta propiedad permite establecer o no tres estados para el control cuando se ejecuta la aplicación. Dentro de una aplicación podemos utilizar la propiedad CheckState para preguntar el estado de la casilla:

```
if checkbox1.CheckState=
```

CheckState.Checked
 CheckState.Indeterminate
 CheckState.Unchecked

También podemos preguntar si la casilla está activada o no utilizando la propiedad Checked:

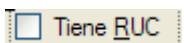
```
if checkbox1.Checked=
```

False
 True

UseMnemonic

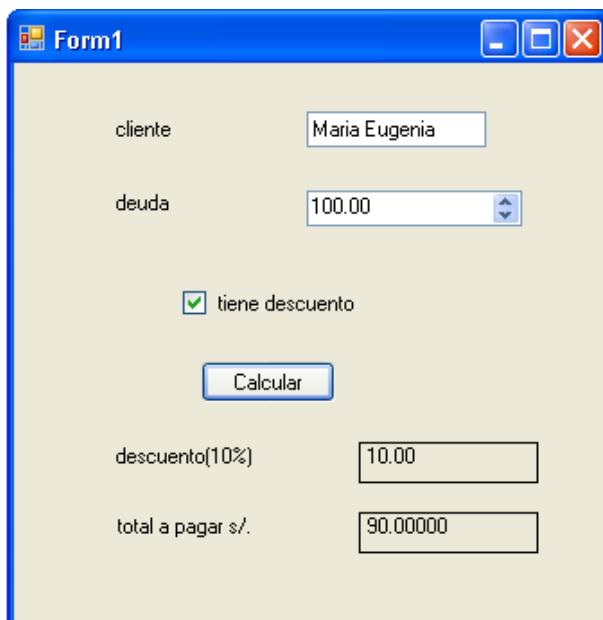
Esta propiedad se utiliza para indicar si dentro del texto se puede utilizar una letra rápida con el símbolo & para activar o desactivar la casilla.

En el ejemplo se utiliza la letra R.



Aplicación desarrollada Nº IV-08:

Este programa permite ingresar el nombre y la deuda de un cliente indicar mediante un control CheckBox si el cliente tiene o no un descuento que consiste en el 10% de su deuda.



Los controles que se utilicen para desarrollar este programa son:



El control NudDeuda tiene las siguientes propiedades:

DecimalPlaces	2
---------------	---

Johan Guerreros Montoya

Increment	10
Maximum	10000
Minimum	1

En control CHKDescuento tiene antes de la letra D el símbolo &en su propiedad text para que se active o desactive la casilla al pulsar las teclas Alt + D.

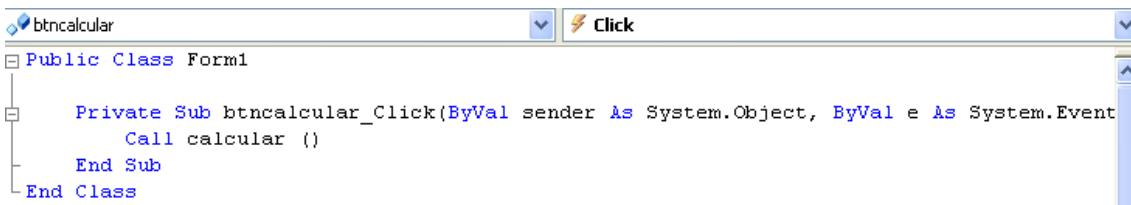
Text	tiene &descuento
------	-----------------------------

Los controles lbldescuento y lbltotal tiene las siguientes propiedades .

AutoSize	False
BorderStyle	FixedSingle
TextAlign	MiddleRight

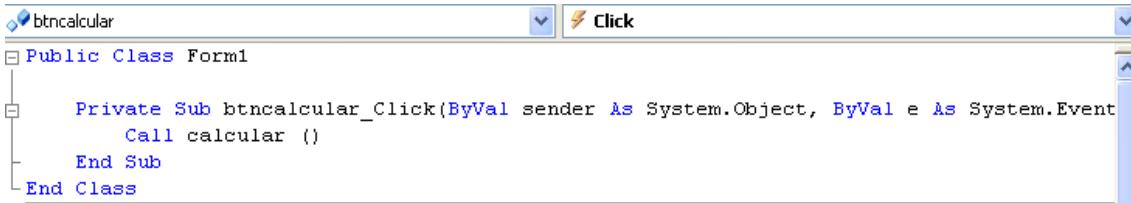
Las instrucciones del programa se escribe dentro de un procedimiento llamado calcular el cual es llamado cuando se hace clic en el botón calcular y cuando se activa o desactiva la casilla .

Instrucciones del botón Btncalcular.



```
btncalcular
Public Class Form1
    Private Sub btncalcular_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
        Call calcular()
    End Sub
End Class
```

Instrucciones del evento Checkedchanged del control chkdescuento .



```
btncalcular
Public Class Form1
    Private Sub btncalcular_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
        Call calcular()
    End Sub
End Class
```

Instrucciones del procedimiento calcular

Dim deuda, descuento, total AsDecimal

'deuda =decimal.parce(nuddeuda.value)

deuda = nuddeuda.Value

If chkdescuento.Checked = TrueThen

descuento = deuda * 10 / 100

Johan Guerreros Montoya

```
Else
```

```
descuento = 0
```

```
EndIf
```

```
total = deuda - descuento
```

```
lbldescuento.Text = descuento.ToString("###,##0.00")
```

```
lbltotal.Text = total.ToString("###.##0.00")
```

Resultado del programa sin asignar descuento.



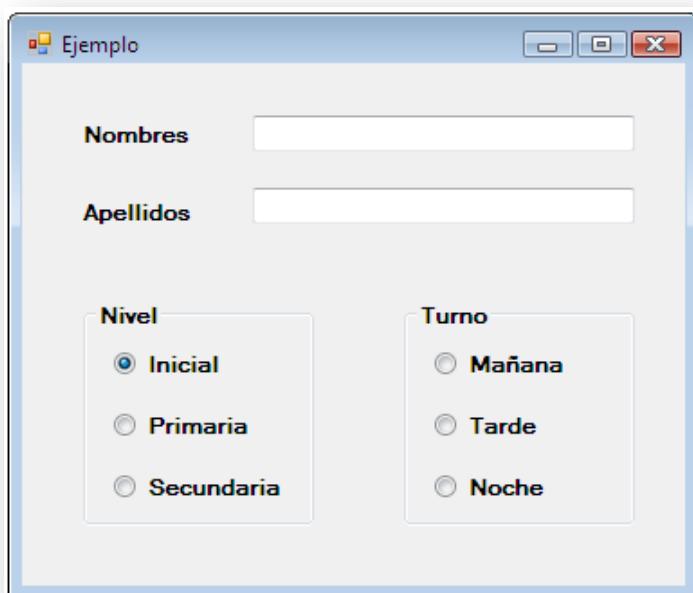
El Control RadioButton



Este control permite seleccionar dentro de una aplicación solo una de un grupo de opciones. Cada opción disponible para el usuario es un control RadioButton y cada grupo de opciones deben estar dentro de un control GroupBox. Ejemplo:

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Nombres
Label2	Text	Apellidos
TextBox1	Name	TxtNombres
TextBox2	Name	TxtApellidos
GroupBox1	Text	Nivel
GroupBox2	Text	Turno
RadioButton1	Name	RbInicial
	Text	Inicial
RadioButton2	Name	RbPrimaria
	Text	Primaria
RadioButton3	Name	RbSecundaria
	Text	Secundaria
RadioButton4	Name	RbMañana
	Text	Mañana
CONTROL	PROPIEDAD	VALOR
RadioButton5	Name	RbTarde
	Text	Tarde
RadioButton6	Name	RbNoche
	Text	Noche



Sus principales propiedades son:

Name

Johan Guerreros Montoya

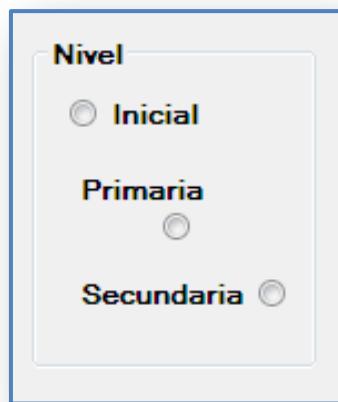
Se utiliza para asignar un nombre particular al control . Se recomienda que empiece con las letras Rb.

Appearance

Permite cambiar la apariencia del RadioButton por un botón de comandos.

CheckAlign

Esta propiedad se utiliza para indicar la posición del botón de opción.



Checked

Esta propiedad se utiliza para indicar si el botón de opción debe estar activado o desactivado.

False



True



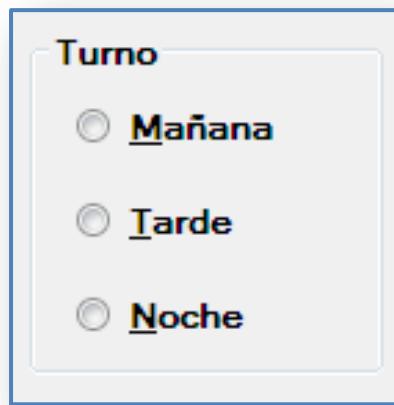
Text

Esta propiedad se utiliza para escribir el texto que debe acompañar al botón de opción.

UseMnemonic

Esta propiedad se utiliza para indicar si dentro del texto se puede utilizar una letra rápida con el símbolo & para activar o desactivar el botón de opción.

En el siguiente ejemplo se va a utilizar el símbolo & en la primera letra de cada turno:



Aplicación Desarrollada Nro IV-09

Este programa permite calcular el pago mensual que debe realizar cada alumno según el nivel y turno en el que se matricula. El pago mensual también depende si se le asigna o no un descuento del 20%.

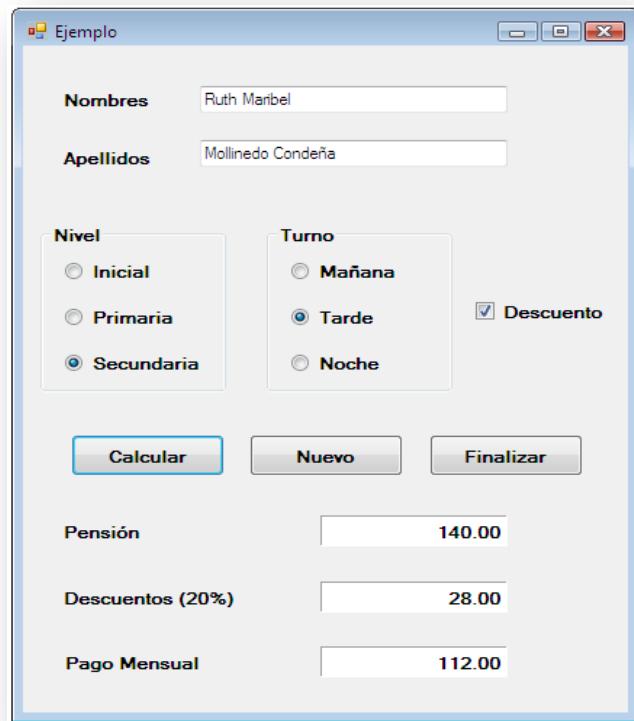
FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Nombres
Label2	Text	Apellidos
Label3	Text	Pensión
Label4	Text	Descuento (20%)
Label5	Text	Pago Mensual
Label6	Name	LblPensión
Label7	Name	LblDescuento
Label8	Name	LblPago
TextBox1	Name	TxtNombres
TextBox2	Name	TxtApellidos
GroupBox1	Text	Nivel
GroupBox2	Text	Turno
CONTROL	PROPIEDAD	VALOR
CheckBox1	Name Text	ChkDescuento &Descuento
RadioButton1	Name Text	RbInicial &Inicial
RadioButton2	Name Text	RbPrimaria &Primaria
RadioButton3	Name Text	RbSecundaria &Secundaria
RadioButton4	Name Text	RbMañana &Mañana
RadioButton5	Name Text	RbTarde Tarde
RadioButton6	Name Text	RbNoche Noche
Button1	Name	BtnCalcular

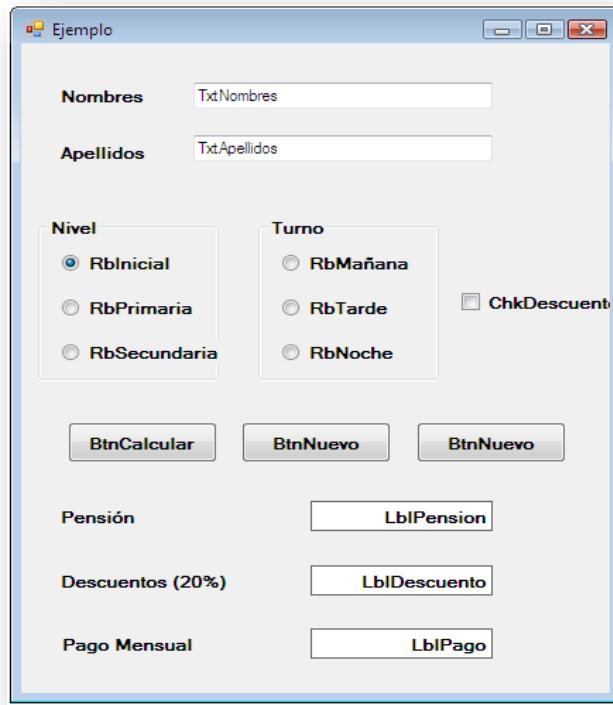
	Text	&Calcular
Button2	Name Text	BtnNuevo &Nuevo
Button3	Name Text	BtnFinalizar &Finalizar

Primero se calcula la pension que depende de la siguiente tabla:

NIVEL	MAÑANA	TARDE	NOCHE
Inicial	80.00	80.00	80.00
Primaria	100.00	120.00	90.00
Secundaria	140.00	160.00	110.00



Los controles para este programa son:



A cada una de las cajas de texto asigneles el valor 35 en su propiedad MaxLength para limitar la cantidad de caracteres que se ingresen en los nombres y apellidos.

MaxLength	35
-----------	----

A cada uno de los botones de opciones y al control ChkDescuento asignele el simbolo & antes de la priemra letra de la aplabra que escribe en su propiedad Text para poder activarlos o desactivarlos en forma rapida pulsando la tecla Alt + la letra indicada. Ejemplo:

Text	&Inicial
------	----------

Text	&Tarde
------	--------

Text	&Descuento
------	------------

Los controles LblDescuento y LblPago tiene las sigueitnes propiedades:

AutoSize	<input type="checkbox"/>	False
----------	--------------------------	-------

BorderStyle	<input type="checkbox"/>	Fixed3D
-------------	--------------------------	---------

TextAlign	<input type="checkbox"/>	MiddleRight
-----------	--------------------------	-------------

Instrucciones del botón Calcular

Johan Guerreros Montoya

```

'Colocamos el codigo dentro del evento click del Boton
BtnCalcular, este se ejecutara al hacer click en este.

PrivateSub BtnCalcular_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles BtnCalcular.Click

'Declara las variables Pension, Descuento y Pago como Decimal
que será el factor de ajuste especifica el número de dígitos a
la derecha del separador decimal; comprendidos entre 0 y 28

Dim Pension, Descuento, Pago AsDecimal

'Pregunta el nivel de la matricula

If RbInicial.Checked = TrueThen

'Pension para el nivel Inicial

    Pension = 80

'Pregunta el Nivel Primario

ElseIf RbPrimaria.Checked = TrueThen

'Pregunta la pension para el Turno Mañana

If RbMañana.Checked = TrueThen

'Pension para el Turno Mañana

    Pension = 100

'Pregunta la pension para el Turno de Tarde

ElseIf RbTarde.Checked = TrueThen

'Pension para el Turno Tarde

    Pension = 120

'Pregunta la pension para el Turno de Noche

ElseIf RbNoche.Checked = TrueThen

'Pension para el Turno Noche

    Pension = 90

'Comprueba la elección de la condicion

Else

'Muestra el mensaje

    MsgBox("Seleccione el Turno",
MsgBoxStyle.Critical, "Por Favor")

'Salimos de la condición

```

```

EndIf

'Pregunta el Nivel secundario

ElseIf RbSecundaria.Checked = TrueThen

'Pregunta la pension para el Nivel Secundaria

If RbMañana.Checked = TrueThen

'Pension para el Turno Mañana

    Pension = 160

'Pregunta la pension para el Turno de Tard

ElseIf RbTarde.Checked = TrueThen

'Pension para el Turno Tarde

    Pension = 140

'Pregunta la pension para el Turno de Noche

ElseIf RbNoche.Checked = TrueThen

'Pension para el Turno Noche

    Pension = 110

'Comprueba la eleccion de la condicion

Else

'Muestra el mensaje

    MsgBox("Seleccione el Turno",
        MsgBoxStyle.Critical, "Por Favor")

'Salimos de la condición

EndIf

'Comprueba la eleccion de la condicion

Else

'Muestra el mensaje

    MsgBox("Seleccione el Nivel", MsgBoxStyle.Critical,
        "Por Favor")

'Salimos de la condición

EndIf

'Calcula el descuento

If ChkDescuentos.Checked = TrueThen

'Realiza la operacion para hallar el 20% de descuento

```

```

        Descuento = Pension * 20 / 100

'Comprobando la condicion

Else

'Descuento es igual a cero

        Descuento = 0

'Salimos de la condición

EndIf

'Calcula el total a pagar

        Pago = Pension - Descuento

'Muestra los resultados de la Pension en la etiqueta LblPension

        LblPension.Text = Pension.ToString("###,##0.00")

'Muestra los resultados del descuento en la etiqueta LblDescuento

        LblDescuento.Text = Descuento.ToString("###,##0.00")

'Muestra los resultados del pago total en la etiqueta LblPago

        LblPago.Text = Pago.ToString("###,##0.00")

'Salimos de la Sub-Rutina

EndSub

```

Instrucciones del botón Nuevo

```

    'Colocamos el codigo dentro del evento click del Boton
    BtnNuevo, este se ejecutara al hacer click en este.

PrivateSub BtnNuevo_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles BtnNuevo.Click

'Limpia la caja de texto TxtNombres

        TxtNombres.Clear()

'Limpia la caja de texto TxtApellidos

        TxtApellidos.Clear()

'Desactiva el control RbInicial

        RbInicial.Checked = False

'Desactiva el control RbPrimaria

```

```

        RbPrimaria.Checked = False

'Desactiva el control RbSecundaria

        RbSecundaria.Checked = False

'Desactiva el control RbMañana

        RbMañana.Checked = False

'Desactiva el control RbTarde

        RbTarde.Checked = False

'Desactiva el control RbNoche

        RbNoche.Checked = False

'Desactiva el control ChkDescuentos

        ChkDescuentos.Checked = False

'Regres a la etiqueta LblPension a cero o vacio decimales

        LblPension.Text = String.Empty

'Regres a la etiqueta LblDescuento a cero o vacio decimales

        LblDescuento.Text = String.Empty

'Regres a la etiqueta LblPago a cero o vacio decimales

        LblPago.Text = String.Empty

'Posiciona el cursor el la caja de texto

        TxtNombres.Focus()

'Salimos de la Sub-Rutina

EndSub

```

Instrucción del botón Finalizar

```

'Colocamos el codigo dentro del evento click del Boton
BtnFinalizar, este se ejecutara al hacer click en este.

PrivateSub BtnFinalizar_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles BtnFinalizar.Click

'Cerramos el formulario

        Close()

'Salimos de la Sub-Rutina

EndSub

```

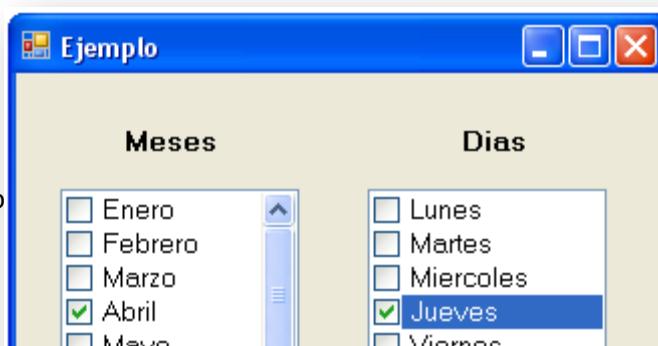
Johan Guerreros Montoya

Aplicación Desarrollada Nro. IV-09**El Control CheckedListBox**

Este control es similar al control ListBox con la diferencia que cada uno de sus elementos se muestran acompañados de una casilla de verificación, es decir, con un control CheckBox para que el usuario los pueda seleccionar.

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Meses
Label2	Text	Días
CheckedListBox1	Name	ChkLstMeses
CheckedListBox2	Name	ChkLstDías



Sus principales propiedades son:

Name

Esta propiedad se utiliza para asignarle un nombre particular al control.

Se recomienda que empiece con las iniciales ChkLst o ChkLb.

CheckOnClick

Con esta propiedad se indica si la casilla cambia de estado al hacer clic una sola vez en el elemento. Si esta propiedad tiene el valor False, la casilla de verificacion cambia de estado a la segunda vez que se haga clic en el elemento.

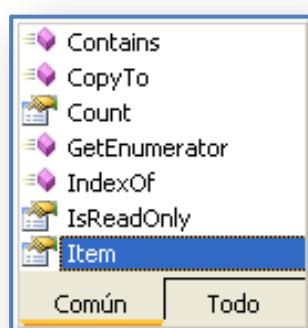
CheckedItems

Esta propiedad representa a todos los elementos activados en el control CheckBoxList, es decir, a todos los elementos que tienen un aspa en su casilla.

Esta propiedad tiene metodos y otras propiedades que permiten manipular los elementos que contienen.

En el siguiente ejemplo pertenece a un control CheckBoxList llamado ChkLstDias.

ChkLstDias.CheckedItems.



Por ejemplo:

Count

Cuenta la cantidad de elementos que estan activados en el control CheckedListBox.

Item

Devuelve el texto del elemento. Se le debe indicar como parametro el indice del elemento.

ColumnWidth

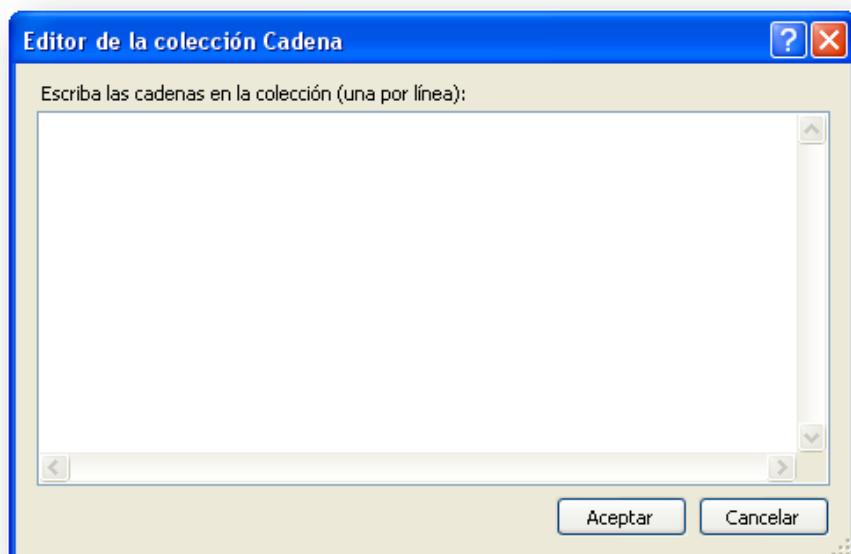
Esta propiedad se utiliza cuando el control CheckedListBox esta configurado para mostrar sus elementos en varias columnas y permite indicar el ancho de cada una de ellas.

HorizontalScrollBar

Esta propiedad se utiliza paraq indicar si el control debe mostrar una barra de dezplazamiento horizontal cuando no se pueda visualizar el texto completo de los elementos que contiene.

Items

En tiempo de diseño esta propiedad se utiliza para escribir los elementos que el control CheckedListBox debe mostrar. Al ingresar a esta propiedad se muestra la siguiente ventana donde debe escribir los elementos y para finalizar haga clic en Aceptar.

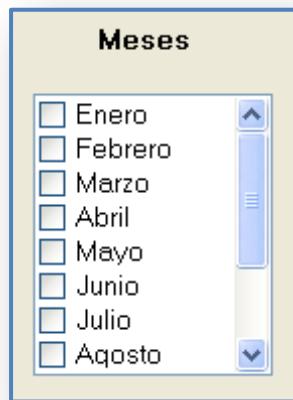


MultiColumn

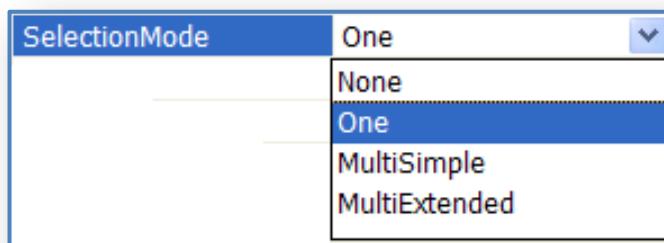
Esta propeidad se utiliza para indicar si los elementos que muestra el control ListBox deben mostrarse ocupando mas de una columna.

ScrollAlwaysVisible

Esta propiedad permite indicar si el control CheckedListBox debe mostrar siempre una barra de desplazamiento horizontal.

**SelectionMode**

Esta propiedad se utiliza para indicar como se pueden seleccionar los elementos que muestra el control CheckedListBox. Las opciones que tienen esta propiedad son:



- La opción **None** no permite seleccionar ningun elemento del control.
- La opcion **One** solo permite seleccionar un elemento del control.
- La opcion **MultiSelect** permite seleccionar varios elementos del control.
- **CheckedListBox** haciendo clic en cada uno de ellos.
- La opcion **MultiExtended** permite seleccionar varios elementos del control **CheckedListBox** utilizando la tecla **Ctrl** o **Shift**.

Sorted

Esta propiedad permite indicar si el control CheckedListBox debe mostrar los elementos ordenados.

ThreeDCheckBoxs

Esta propiedad permite indicar si la casilla de verificacion se debe visualizar en 3D cuando se seleccione.

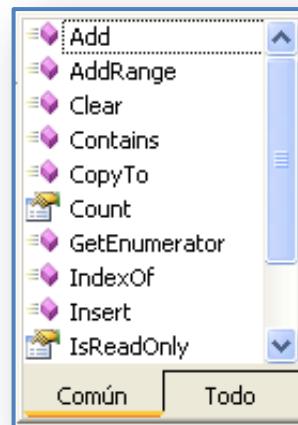
Items

Esta propiedad tambien se puede utilizar mediante codigo, es decir, mediante instrucciones donde representa a todos los elementos que contiene el control.

La propiedad Items tiene varias funciones que permiten administrar los elementos que contiene el control CheckedListBox.

Ejemplo:

```
CheckedListBox1.Items.
```



Add:

Permite agregar elementos al control.

Ejemplo, las siguientes instrucciones agregan tres elementos al control CheckedListBox1.

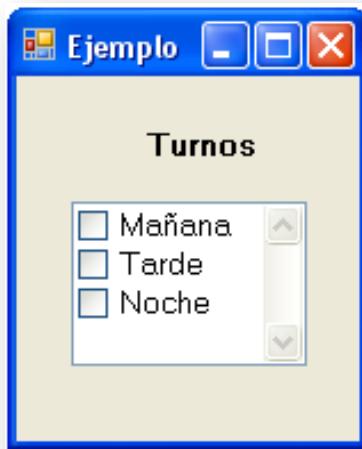
FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Turnos
CONTROL	PROPIEDAD	VALOR
CheckedListBox1	Name	CheckedListBox1

```
'Agrega elementos al control CheckedListBox1
```

```
CheckedListBox1.Items.Add("Mañana")
```

```
'Agrega elementos al control CheckedListBox1
    CheckedListBox1.Items.Add("Tarde")
'Agrega elementos al control CheckedListBox1
    CheckedListBox1.Items.Add("Noche")
```

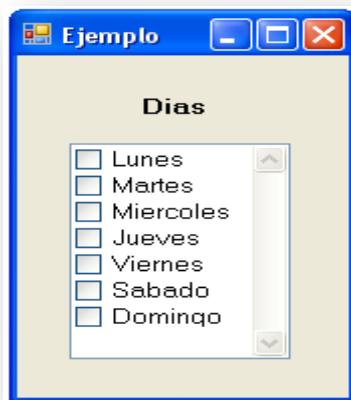


AddRange:

Permite agregar elementos al control que se encuentran en un array. Ejemplo, las siguientes instrucción agregan los días de la semana al control CheckedListBox1 utilizando un array.

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Días
CheckedListBox1	Name	CheckedListBox1



```

'Declaramos un array de 6 caracteres como cadena
Dim A(6) AsString

'El primer caracter es Lunes
A(0) = "Lunes"

'El primer caracter es Martes
A(1) = "Martes"

'El primer caracter es Miercoles
A(2) = "Miercoles"

'El primer caracter es Jueves
A(3) = "Jueves"

'El primer caracter es Viernes
A(4) = "Viernes"

'El primer caracter es Sábado
A(5) = "Sabado"

'El primer caracter es Domingo
A(6) = "Domingo"

'Añade el rango de 6 caracteres en el control CheckedListBox1
CheckedListBox1.Items.AddRange(A)

```

Clear:

Elimina todos los elementos que contiene el control. Ejemplo, la siguiente instrucción elimina todos los elementos del control CheckedListBox1.

```
CheckedListBox1.Items.Clear()
```

Count:

Devuelve la cantidad de elementos que contiene el control ListBox o ComboBox. Ejemplo, las siguientes instrucciones muestran en el control LblCantidad la cantidad de elementos que tiene el control ListBox1.

FORMULARIO EJEMPLO

Johan Guerreros Montoya

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Días
Label2	Name	LblCantidad
Button1	Name Text	BtnVer &Ver
CheckedListBox1	Name	CheckedListBox1



```
'Declaramos la variable N como entero
Dim N As Integer

'Almacenamos en la valrible N la cantidad de elementos que hay
en el control CheckedListBox1

N = CheckedListBox1.Items.Count

'Muestra la cantidad de elementos en la etiqueta LblCantidad

LblCantidad.Text = "Son: "& N &" Elementos"
```

Insert:

Inserta un nuevo elemento en el control. Se debe indicar la posición y el elemento a insertar. La primera posición es 0. Ejemplo, la siguiente instrucción inserta la palabra Vacaciones después del día martes.

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Días
Label2	Name	LblCantidad
Button1	Name Text	BtnVer &Ver
CheckedListBox1	Name	CheckedListBox1

Johan Guerreros Montoya



```

'Inserta un elemento "Vacaciones" después del carácter 2
CheckedListBox1.Items.Insert(2, "Vacaciones")

'Declara la variable N como entero
Dim N As Integer

'Almacenamos en la variable N la cantidad de elementos que hay
en el control CheckedListBox1
N = CheckedListBox1.Items.Count

'Muestra la cantidad de elementos en la etiqueta LblCantidad
LblCantidad.Text = "Son " & n & " Elementos"

```

Remove:

Permite eliminar un elemento del control. Se le debe enviar como parámetro el texto del elemento a eliminar. Por ejemplo, la siguiente instrucción elimina el día Jueves.

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Días
Label2	Name	LblCantidad
Button1	Name Text	BtnVer &Ver
CheckedListBox1	Name	CheckedListBox1



```

'Remueve un elemento "Jueves"

CheckedListBox1.Items.Remove("Jueves")

'Declara la variable N como entero

Dim N As Integer

'Almacenamos en la variable N la cantidad de elementos que hay
en el control CheckedListBox1

N = CheckedListBox1.Items.Count

'Muestra la cantidad de elementos en la etiqueta LblCantidad

LblCantidad.Text = "Son " & N & " Elementos"

```

RemoveAt:

Permite eliminar un elemento del control. Se le debe enviar como parámetro el número del elemento a eliminar. El primer elemento tiene el valor cero (0). Por ejemplo, la siguiente instrucción elimina el día martes.

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Días
Label2	Name	LblCantidad
Button1	Name Text	BtnVer &Ver
CheckedListBox1	Name	CheckedListBox1



```
'Remueve de la cuadro al parametro en el que se encuentra el
elemento

    CheckedListBox1.Items.RemoveAt(1)

'Declara la variable N como entero

Dim N As Integer

'Almacenamos en la variable N la cantidad de elementos que hay
en el control CheckedListBox1

    N = CheckedListBox1.Items.Count

'Muestra la cantidad de elementos en la etiqueta LblCantidad

    LblCantidad.Text = "Son "& N &" Elementos"
```

GetItemChecked

Es un método que devuelve el valor True si un elemento especificado está activado o no. El primer elemento es cero (0).

Ejemplo, las siguientes instrucciones indican si el segundo elemento del control CheckedListBox1 está activado o no.

```
'Verifica si se selecciona los elementos del control
CheckedListBox1

If CheckedListBox1.GetItemChecked(1) = FalseThen

    'Si se selecciona se asigna el mensaje "Si"

        MsgBox("si")

    'Verifica si la condición es falsa

Else

    'Si no se selecciona se asigna el mensaje "No"

        MsgBox("No")
```

```
'Cerramos la condición
EndIf

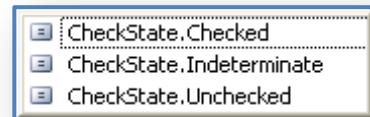
'Salimos de la Sub-Rutina
EndSub
```

GetItemCheckState

Este método permite saber el estado de un elemento del control. Se debe indicar el índice del elemento siendo cero (0) el primer elemento.

Los valores que devuelve lo podemos visualizar al utilizarlo dentro de una instrucción If Then. En el siguiente ejemplo se está preguntando por el estado del segundo elemento del control CheckedListBox1.

```
If CheckedListBox1.GetItemCheckState(1) =
```



GetSelected

Este método se utiliza para saber si un elemento del control esta seleccionado. Si esta selecciona devuelve True. En el siguiente ejemplo las instrucciones muestran la palabra Si.

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Días
CheckedListBox1	Name	CheckedListBox1



```

'Verifica la selección del parametro específico
If CheckedListBox1.GetSelected(1) = TrueThen
  'Si es el parametro correcto muestra el mensaje "Si"
  MsgBox("si")
  'Verifica si la condición es falsa
Else
  'Si no es el parametro correcto muestra el mensaje "No"
  MsgBox("no")
  'Cerramos la condición
EndIf
'Salimos de la Sub-Rutina
EndSub

```

SelectedItem

Esta propiedad devuelve el texto del elemento actualmente seleccionado. En el siguiente ejemplo la instrucción mostrara la palabra Miercoles.

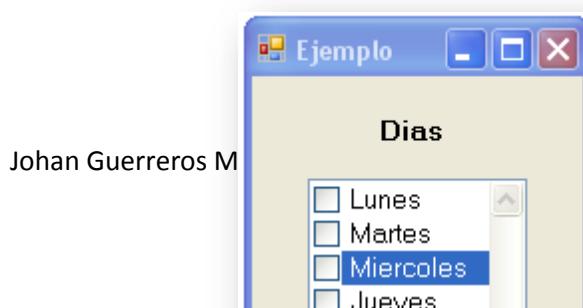
FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Días
CheckedListBox1	Name	CheckedListBox1

```

'Muestra el nombre del parametro seleccionado
MsgBox(CheckedListBox1.SelectedItem)

```





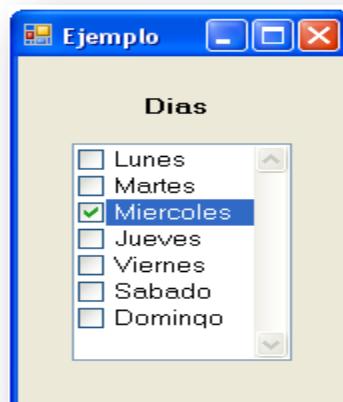
SetItemChecked

Este método permite activar o desactivar la casilla de cualquier elemento del control CheckedListBox. Se debe indicar el índice del elemento y el valor True para activar la casilla o False para desactivarla. La siguiente instrucción de ejemplo activa el tercer elemento del control CheckedListBox1.

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Días
CheckedListBox1	Name	CheckedListBox1

```
'Selecciona el parametro especificado
CheckedListBox1.SetItemChecked(2, True)
```



SetItemCheckState

Este método permite asignar un estado la casilla de cualquier elemento del control CheckedListBox. Se debe indicar el índice del elemento y el estado que le desea asignar. La siguiente instrucción de ejemplo activa el tercer elemento del control CheckedListBox1.

```
CheckedListBox1.SetItemCheckState(2, CheckState.Checked)
```

SetSelected

Este método permite seleccionar o quitar la selección a cualquier elemento del control CheckedListBox. Se debe indicar el índice del elemento y el valor True para seleccionarlo o False para quitarle la selección. La siguiente instrucción de ejemplo selecciona el primer elemento del control CheckedListBox1.

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
CONTROL	PROPIEDAD	VALOR
Label1	Text	Días
CheckedListBox1	Name	CheckedListBox1



```
CheckedListBox1.SetSelected(0, True)
```

Text

Esta propiedad devuelve el texto del elemento actualmente seleccionado. La siguiente instrucción de ejemplo mostrara la palabra Jueves:

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
CONTROL	PROPIEDAD	VALOR
Label1	Text	Días
CheckedListBox1	Name	CheckedListBox1

'Muestra el nombre del parámetro seleccionado

MsgBox (CheckedListBox1.Text)

**Aplicación Desarrollada Nº IV-10**

Este programa permite activar de un control CheckedListBox los días de la semana y pasarlos a un control ListBox.

En la siguiente ventana de ejemplo se ha seleccionado y pasado al control ListBox los días Miércoles, Viernes y Domingo.

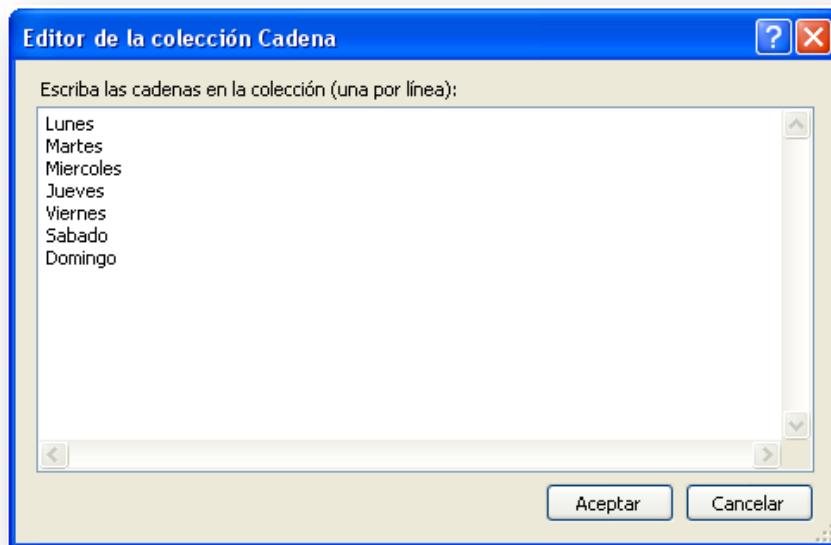
FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
ListBox1	Name	LstDías
CheckedListBox1	Name	CheckedListBox1
Button1	Name	BtnPasar
	Text	&Pasar

El programa tiene los siguientes controles:



En la propiedad Items del control ChkListDias agregue los nombres de los días de la semana.



Instrucciones del botón BtnPasar

'Colocamos el código dentro del evento click del Botón BtnPasar, este se ejecutara al hacer click en este.'

Johan Guerreros Montoya

```

PrivateSub BtnPasar_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles BtnPasar.Click
    'Declaramos la variable I como Entero
    Dim I AsInteger
    'Limpia el contenido del control listBox LstDias
    LstDias.Items.Clear()
    'Pregunta si no hay elementos activados
    If ChkLstDias.CheckedItems.Count = 0 Then
        'Verifica si ha seleccionado de lo contrario envia el mensaje
        MsgBox("Activa los elementos que desea pasar",
        MsgBoxStyle.Critical, "Por Favor")
        'Aparece el mensaje de error pidiendo q ingrese datos
        'Salimos de la Sub-Rutina
        ExitSub
        'Salimos de la Sub-Rutina
    EndIf
    'Con la función for contamos la cantidad de elementos del
    control CheckedListBox ChkLstDias
    For i = 0 To ChkLstDias.Items.Count - 1
        'Pregunta si el elemento esta seleccionado
        If ChkLstDias.GetItemChecked(i) = TrueThen
            'Pasa el elemento al control ListBox LstDias
            LstDias.Items.Add(ChkLstDias.Items(I))
            'Salimos de la Sub-Rutina
        EndIf
        'Continua con las instrucciones
    Next

```

Las instrucciones del botón BtnPasar también se puede desarrollar utilizando la propiedad CheckedItems que contiene solo los elementos seleccionados del control CheckedListBox.

Instrucciones del botón BtnPasar con la propiedad CheckedItems

```
'Colocamos el código dentro del evento click del Botón BtnPasar,
este se ejecutara al hacer click en este.

Private Sub BtnPasar_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles BtnPasar.Click

'Declaramos la variable I como Entero

Dim I As Integer

'Limpia el contenido del control listbox LstDias

    LstDias.Items.Clear()

'Pregunta si no hay elementos activados

If ChkLstDias.CheckedItems.Count = 0 Then

'Verifica si no hay elementos seleccionados, muestra el mensaje

    MsgBox("Activa los elementos que desea pasar",
MsgBoxStyle.Critical, "Por favor")

'Aparece el mensaje de error pidiendo q ingrese datos

'Salimos de la Sub-Rutina

Exit Sub

'Salimos de la condición

End If

'Basta un For con la cantidad de elementos del control
CheckedListBox

For i = 0 To ChkLstDias.CheckedItems.Count - 1

'Pregunta si el elemento esta seleccionado

If ChkLstDias.GetItemChecked(I) = True Then

'Pasa el elemento al control ListBox LstDias

    LstDias.Items.Add(ChkLstDias.CheckedItems(I))

'Salimos de la condición

End If

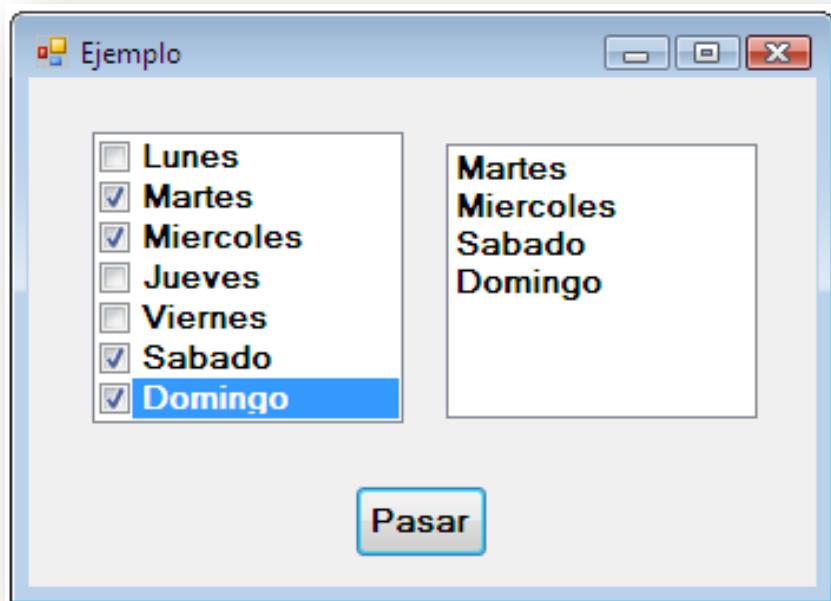
'Continua con las instrucciones

Next

'Salimos de la Sub-Rutina

End Sub
```

Con estas instrucciones el resultado se mostrara mas rapido:

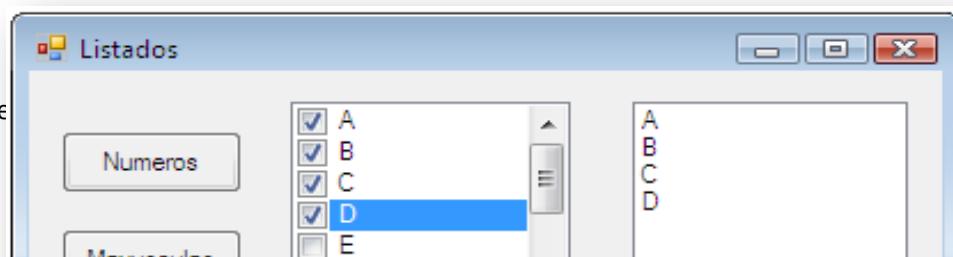


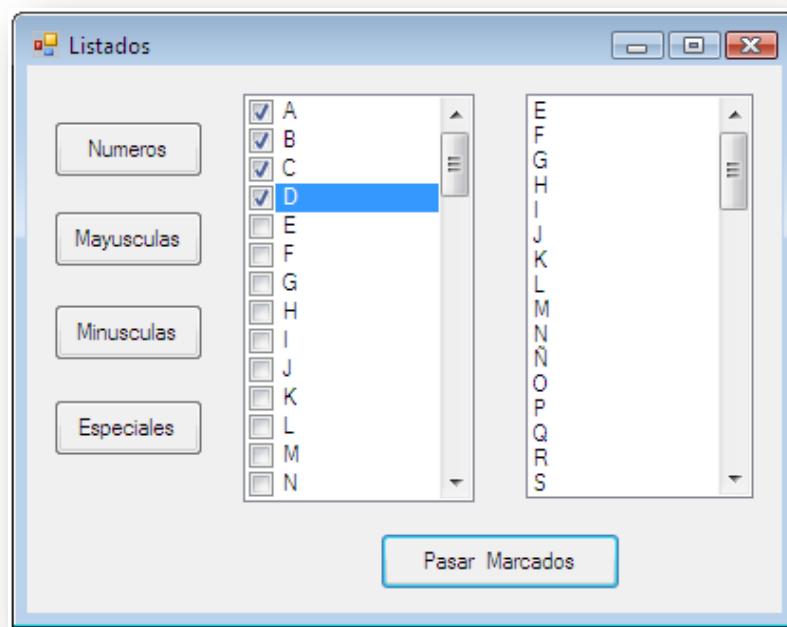
Aplicación Desarrollada N° IV - 11

Este programa de ejemplo permite agregar a un control CheckedListBox los números del 0 al 9, las letras mayúsculas, minúsculas y los caracteres especiales: á, é, í, ó, ú, ñ, Ñ. También permite seleccionar cualquier elemento y pasarlo a un control ListBox, teniendo la posibilidad de pasar también los que no se ha seleccionado utilizando un solo botón.

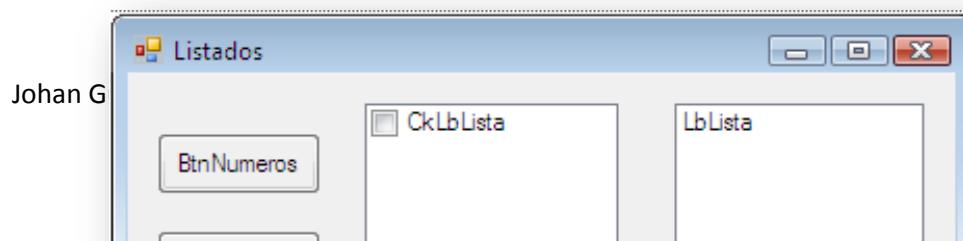
FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Listados
ListBox1	Name	LbDías
CheckedListBox1	Name	CkLbLista
Button1	Name Text	BtnNumeros &Numeros
CONTROL	PROPIEDAD	VALOR
Button2	Name Text	BtnMayusculas &Mayusculas
Button3	Name Text	BtnMinusculas &Minusculas
Button4	Name Text	BtnEspeciales &Especiales
Button5	Name Text	BtnPasar &Pasar

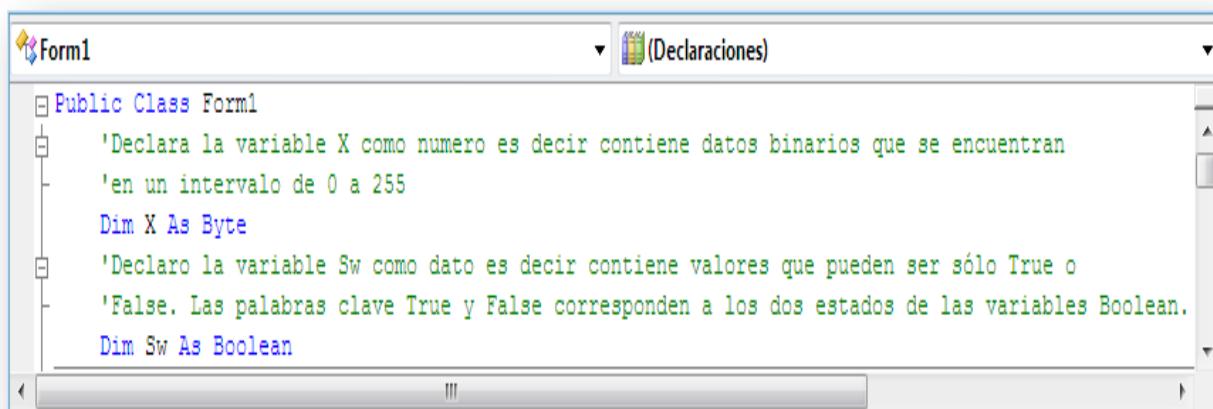




Controles utilizados



Para desarrollar este programa, primero debe declarar dos variables a nivel del formulario:



```

Form1
Public Class Form1
    'Declara la variable X como numero es decir contiene datos binarios que se encuentran
    'en un intervalo de 0 a 255
    Dim X As Byte
    'Declaro la variable Sw como dato es decir contiene valores que pueden ser sólo True o
    'False. Las palabras clave True y False corresponden a los dos estados de las variables Boolean.
    Dim Sw As Boolean

```

Instrucciones del botón BtnNumeros

'Colocamos el código dentro del evento click del Botón BtnNumeros, este se ejecutara al hacer clic en este

```
PrivateSub BtnNumeros_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles BtnNumeros.Click
```

'Limpia los items del control CkLbLista

```
CkLbLista.Items.Clear()
```

'Limpia los items del control LbLista

```
LbLista.Items.Clear()
```

Johan Guerreros Montoya

```

'Muestra la tabla, la variable X alamacena los datos del codigo
askin del 48 al 57

For X = 48 To 57

'Muestra el codigo ingresado en el control CheckedListBox1
CkLbLista

CkLbLista.Items.Add(Chr(X))

'Continua con las instrucciones

Next

'Salimos de la Sub-Rutina

EndSub

```

Instrucciones de botón BtnMayúsculas

```

'Colocamos el codigo dentro del evento click del Boton
BtnMayusculas, este se ejecutara al hacer clic en este

PrivateSub BtnMayusculas_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles BtnMayusculas.Click

'Limpia los items del control CkLbLista

CkLbLista.Items.Clear()

'Limpia los items del control LbLista

LbLista.Items.Clear()

'Muestra la tabla, la variable X alamacena los datos del codigo
askin del 65 al 92

For X = 65 To 92

'Muestra el codigo ingresado en el control CheckedListBox1
CkLbLista

CkLbLista.Items.Add(Chr(X))

'Continua con las instrucciones

Next

'Salimos de la Sub-Rutina

EndSub

```

Instrucciones del botón BtnMinúsculas

Johan Guerreros Montoya

```

'Colocamos el codigo dentro del evento click del Boton
BtnMinusculas, este se ejecutara al hacer clic en este

PrivateSub BtnMinusculas_Click(ByVal sender As System.Object,
 ByVal e As System.EventArgs) Handles BtnMinusculas.Click

'Limpia los items del control CkLbLista

    CkLbLista.Items.Clear()

'Limpia los items del control LbLista

    LbLista.Items.Clear()

'Muestra la tabla, la variable X alamacena los datos del codigo
askin del 97 al 122

For X = 97 To 122

'Muestra el codigo ingresado en el control CheckedListBox1
CkLbLista

    CkLbLista.Items.Add(Chr(X))

Next

'Salimos de la Sub-Rutina

EndSub

```

Instrucciones del botón BtnEspeciales

```

'Colocamos el codigo dentro del evento click del Boton
BtnEspeciales, este se ejecutara al hacer click en este

PrivateSub BtnEspeciales_Click(ByVal sender As System.Object,
 ByVal e As System.EventArgs) Handles BtnEspeciales.Click

'Limpia el control CkLbLista

    CkLbLista.Items.Clear()

'Limpia los items del control LbLista

    LbLista.Items.Clear()

'Muestra del codigo ascii (225 - ß)

    CkLbLista.Items.Add(Chr(225))

'Muestra del codigo ascii (233 - ø)

```

Johan Guerreros Montoya

```

        CkLbLista.Items.Add(Chr(233))

'Muestra del codigo ASCII (237 - Ø)

        CkLbLista.Items.Add(Chr(237))

'Muestra del codigo ASCII (243 - ≤)

        CkLbLista.Items.Add(Chr(243))

'Muestra del codigo ASCII (250 - ·)

        CkLbLista.Items.Add(Chr(250))

'Muestra del codigo ASCII (241 - ±)

        CkLbLista.Items.Add(Chr(241))

'Muestra del codigo ASCII (209 - ¯)

        CkLbLista.Items.Add(Chr(209))

'Salimos de la Sub-Rutina

EndSub

```

Instrucciones del botón BtnPasar

```

'Colocamos el codigo dentro del evento clic del Boton BtnPasar,
este se ejecutara al hacer clic en este

PrivateSub BtnPasar_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles BtnPasar.Click

'La variable Sw es igual a la negacion de la misma

        Sw = Not Sw

'Verifica la seleccion de las elementos

        If Sw Then

'Muestra los elementos no marcados

                BtnPasar.Text = "Pasar & No marcados"

'Verifica si la condicion es falsa

        Else

'Muestra los elementos marcados

                BtnPasar.Text = "Pasar & Marcados"

```

```

'Cerramos el condicional
EndIf

'Limpia el control LbLista
    LbLista.Items.Clear()

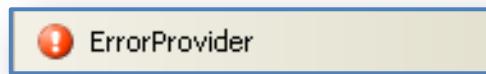
'Pasa los elementos marcados o no marcados
For I AsByte = 0 To CkLbLista.Items.Count - 1
'Verifica los elementos seleccionados en el control CkLbLista
If CkLbLista.GetItemChecked(I) = Sw Then
    'Almacena los elementos seleccionados del control CkLbLista y
    'los guarda en el control LbLista
    LbLista.Items.Add(CkLbLista.Items(I))
'Cerramos el condicional
EndIf

'Continua con las instrucciones
Next

'Salimos de la Sub-Rutina
EndSub

```

El Control ErrorProvider

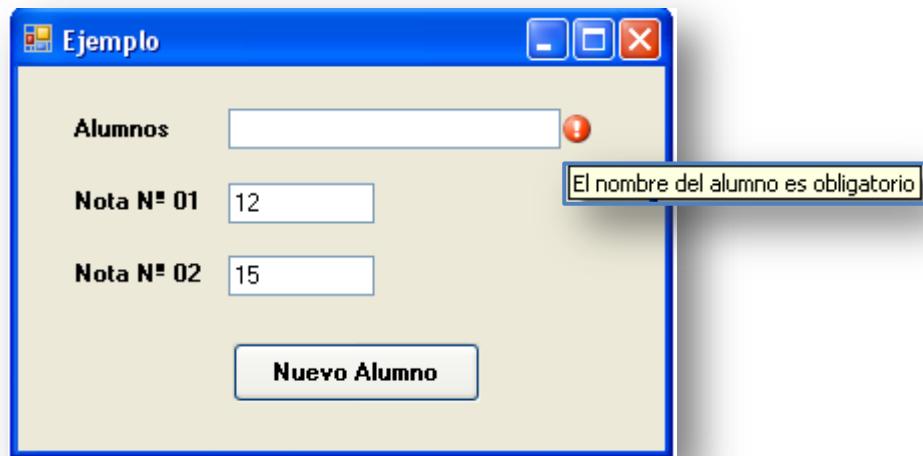


Este control permite realizar consistencias de datos en un formulario, es decir, evita que los usuarios de nuestra aplicación cometan errores especialmente en el ingreso de los datos.

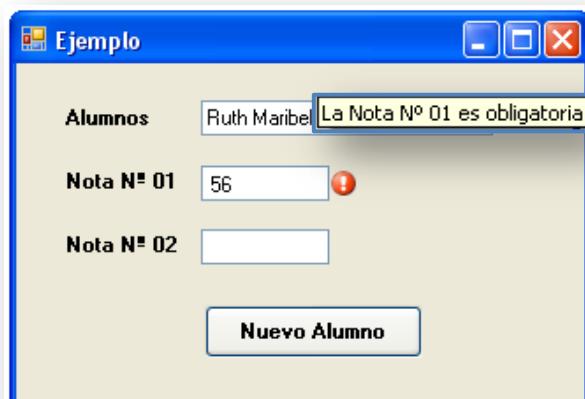
Por ejemplo en la siguiente ventana se muestra un mensaje de error para el usuario indicando que el nombre es obligatorio.

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Alumnos
Label2	Text	Nota N°01
Label3	Text	Nota N°02
TextBox1	Name	TxtNombre
TextBox2	Name	TxtNota01
TextBox3	Name	TxtNota02
Button1	Name Text	BtnNuevo &NuevoAlumno
ErrorProvider1	Name	ErrorProvider1



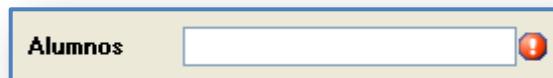
En el siguiente error indica que la nota debe estar entre 0 y 20.



Sus principales propiedades son:

BlinkRate

Esta propiedad se utiliza para indicar el tiempo que debe demorar cada parpadeo del icono de error que muestra el control. El valor predeterminado es 250 milisegundos.



BlinkStyle

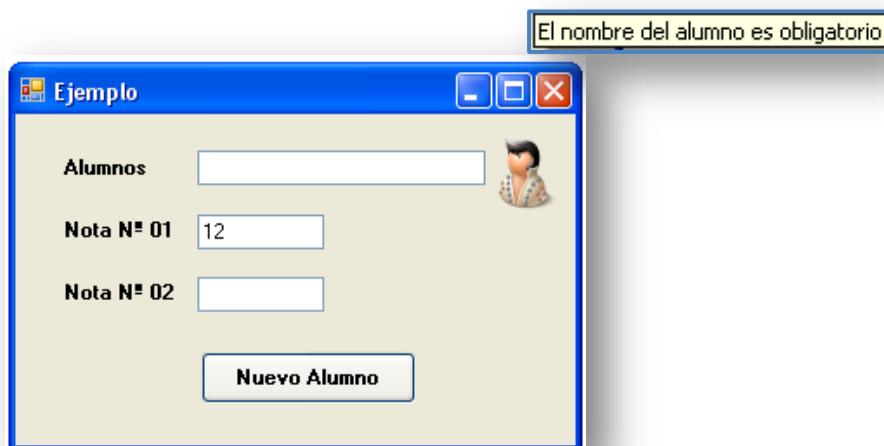
Esta propiedad se utiliza para indicar el tipo de parpadeo del icono de error. La opción NeverBlink muestra el icono sin parpadeo.



Icon

Esta propiedad se utiliza para seleccionar el ícono de error.

En la siguiente ventana de ejemplo se ha cambiado el ícono de error:



Al utilizar el ErrorProvide los controles deben tener el valor True en su propeidad CausesValidation.

CausesValidation	True
------------------	------

La consistencia de los datos o el control de los errores se realiza en el evento Validating de cada control.

El mensaje de error que se debe msotrar cuando el usuario pase el puntero del mouse por el icono de error se asigna con el metodo SetError del control ErrorProvider cuya sintaxis es la siguiente.

```
NombErrorProvider.SetError(NombControl,"Mensaje")
```

NombErrorProvider

Es el nombre del control ErrorProvider que se esta utilizando en el formulario.

NombControl

Es el nombre del control donde se esta realizando el control de errores.

Las siguientes instrucciones de ejemplo controlan que el ingreso de un nombre sea obligatorio. Si el nombre esta vacio se asigna un mensaje de error. Estas instrucciones pertencen al evento Validating del control TxtNombre.

```
'Colocamos el código en el evento Validating de la caja de texto
TxtNombre ya que en el evento Validating verifican que la
propeidad .text no sea nula.
```

```
Private Sub TextBox1_Validating(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles
TxtNombre.Validating
```

```
'Verifica si los datos en la caja de texto TxtNombre son nulos y
vacios
```

```
If TxtNombre.Text = String.Empty Then
```

```
'Si hay error, asigna el mensaje del control TxtNombre
```

```
    ErrorProvider1.SetError(TxtNombre, "El nombre del
alumno es obligatorio")
```

```
'Verifica si la condicion es falsa
```

```

Else
'Si no hay error, quita el mensaje del control TxtNombre
    ErrorProvider1.SetError(TxtNombre, "")

'Cerramos el condicional
EndIf

'Salimos de la Sub-Rutina
EndSub

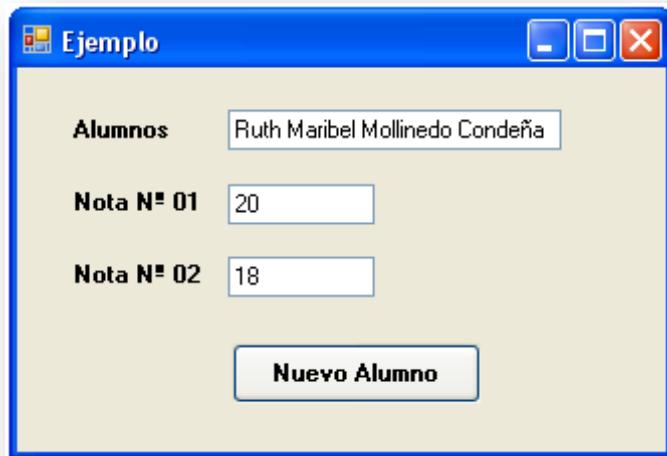
```

Aplicación Desarrollada N° IV-12

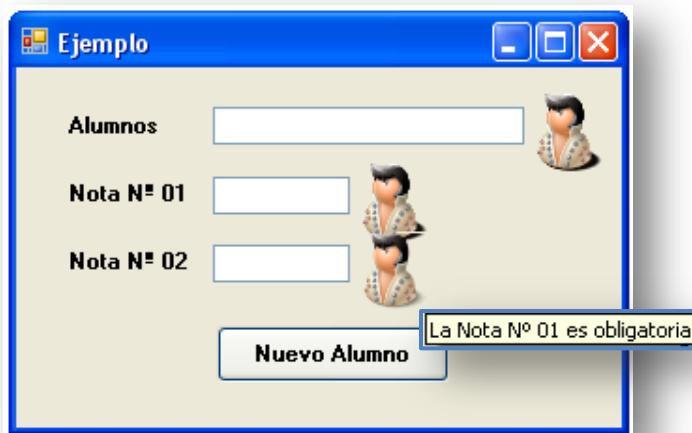
Este programa permite ingresar el nombre y dos notas de un alumno y realizar una consistencia de los datos mediante el control ErrorProvider de tal manera que el nombre y las dos notas se ingresen obligatoriamente, además que las notas estén entre 0 y 20.

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	Alumnos
Label2	Text	Nota N°01
Label3	Text	Nota N°02
TextBox1	Name	TxtNombre
TextBox2	Name	TxtNota01
TextBox3	Name	TxtNota02
CONTROL	PROPIEDAD	VALOR
Button1	Name Text	BtnNuevo &Nuevo Cliente
ErrorProvider1	Name	ErrorProvider1



Ejemplo, en la siguiente ventana no se ha ingresado ninguno de los datos por lo que se muestra el icono de error en las tres capas de texto.



Controles del formulario



Instrucciones del Evento Validating del control TxtNombre

'Colocamos el código en el evento Validating de la caja de texto TxtNombre ya que en el evento Validating verifican que la propiedad .text no sea nula.

```
Private Sub TextBox1_Validating(ByVal sender As Object, ByVal e As System.ComponentModel.CancelEventArgs) Handles TxtNombre.Validating
```

'Verifica si los datos en la caja de texto TxtNombre son nulos y vacíos

```
If TxtNombre.Text = String.Empty Then
```

'Si hay error, asigna el mensaje del control TxtNombre

```

        ErrorProvider1.SetError(TxtNombre, "El nombre del
alumno es obligatorio")

'Verifica si la condicion es falsa

Else

'Si no hay error, quita el mensaje del control TxtNombre

        ErrorProvider1.SetError(TxtNombre, "")

'Cerramos el condicional

EndIf

'Salimos de la Sub-Rutina

EndSub

```

Instrucciones del Evento Validating del control TxtNota1

```

'Colocamos el código en el evento Validating de la caja de texto
TxtNota1 ya que en el evento Validating verifican que la
propiedad .text no sea nula.

Private Sub TxtNota1_Validating(ByVal sender As System.Object,
ByVal e As System.ComponentModel.CancelEventArgs) Handles
TxtNota1.Validating

'Declaramos la variable N como números reales

Dim N As Double

'Verifica si los datos en la caja de texto TxtNota1 son nulos y
vacíos

If TxtNota1.Text = String.Empty Then

'Si hay error, asigna el mensaje del control TxtNota1

        ErrorProvider1.SetError(TxtNota1, "La Nota N° 01 es
obligatoria")

'Verifica si la condicion es falsa

Else

'Almacenamos en la variable N el texto convertido a numero real
almacenado en la caja de texto TxtNota1

N = Double.Parse(TxtNota1.Text)

```

```

'Verifica si los datos en la caja de texto TxtNota1 son nulos y
vacíos y la variable N sea menor a 0 ó N mayor a 20

If TxtNota1.Text = String.Empty And (N < 0 Or N > 20) Then

'Si hay error, asigna el mensaje del control TxtNota1

    ErrorProvider1.SetError(TxtNota1, "La Nota debe
    estar entre 0 y 20")

'Verifica si la condición es falsa

Else

'Si no hay error, quita el mensaje del control TxtNota1

    ErrorProvider1.SetError(TxtNota1, "")

'Cerramos el condicional

EndIf

'Cerramos el condicional

EndIf

Salimos de la Sub-Rutina

EndSub

```

Instrucciones del Evento Validating del control TxtNota2

```

'Colocamos el código en el evento Validating de la caja de texto
TxtNombre ya que en el evento Validating verifican que la
propiedad .text no sea nula.

PrivateSub TxtNota2_Validating(ByVal sender As System.Object,
ByVal e As System.ComponentModel.CancelEventArgs) Handles
TxtNota2.Validating

'Declaramos la variable N como números reales

Dim N AsDouble

'Verifica si los datos en la caja de texto TxtNota2 son nulos y
vacíos

If TxtNota2.Text = String.Empty Then

'Si hay error, asigna el mensaje del control TxtNota2

    ErrorProvider1.SetError(TxtNota2, "La Nota N° 02 es
    obligatoria")

```

Johan Guerreros Montoya

```

'Verifica si la condicion es falsa
Else

'Almacenamos en la variable N el texto convertido a numero real
almacenado en la caja de texto TxtNota2

N = Double.Parse(TxtNota1.Text)

'Verifica si los datos en la caja de texto TxtNota1 son nulos y
vacios y la variable N sea menor a 0 ó N mayor a 20

If TxtNota2.Text = String.Empty And (N < 0 Or N > 20) Then

'Si hay error, asigna el mensaje del control TxtNota2

    ErrorProvider1.SetError(TxtNota2, "La Nota debe
estar entre 0 y 20")

'Verifica si la condicion es falsa

Else

'Si no hay error, quita el mensaje del control TxtNota2

    ErrorProvider1.SetError(TxtNota2, "")

'Cerramos el condicional

EndIf

'Cerramos el condicional

EndIf

'Salimos de la Sub-Rutina

EndSub

```

Instrucciones del BtnNuevo

```

'Colocamos el codigo dentro del evento clic del Boton BtnNuevo,
este se ejecutara al hacer clic en este

PrivateSub BtnNuevo_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles BtnNuevo.Click

'Limpia el contenido de la caja de texto TxtNombre

    TxtNombre.Clear()

'Limpia el contenido de la caja de texto TxtNota1

    TxtNota1.Clear()

```

```
'Limpia el contenido de la caja de texto TxtNota2
TxtNota2.Clear()

'Posiciona el cursor en la caja de texto TxtNombre
TxtNombre.Focus()

'Salimos de la Sub-Rutina
EndSub
```

Aplicación Desarrollada N° IV-13

Este programa permite ingresar datos utilizando el control ErrorProvider para indicar los mensajes de error que el usuario puede cometer mientras ingresa datos.

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Text	&Nombre
Label2	Text	&Edad
Label3	Text	&Sexo
Label4	Text	&Ruc
TextBox1	Name	TxtNombre
TextBox2	Name	TxtEdad
TextBox3	Name	TxtSexo
TextBox4	Name	TxtRuc
Button1	Name Text	BtnNuevo &NuevoCliente
ErrorProvider1	Name	ErrorProv

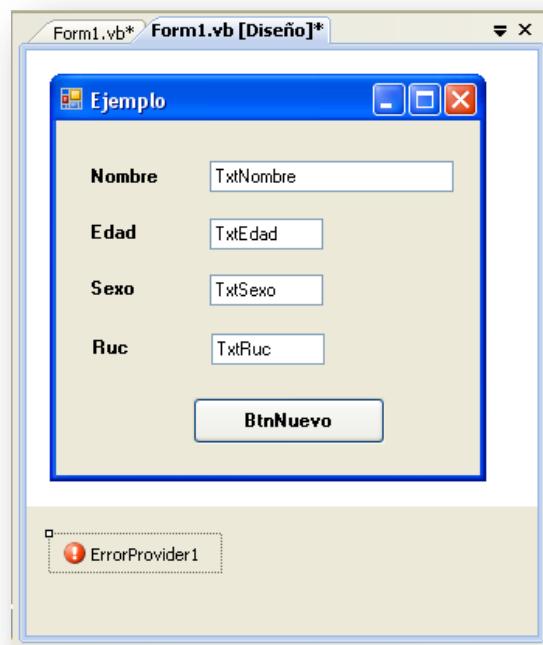


El programa tiene las siguientes características:

- Cuando se pulsa la tecla Enter en una caja de textos, el cursor se ubica en la siguiente.

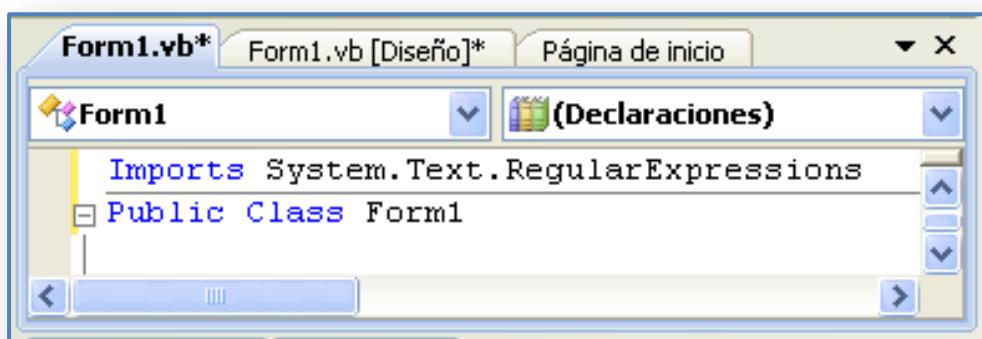
- ✓ El nombre es obligatorio y debe aceptar solo letras y espacios en blanco.
- ✓ La Edad solo debe aceptar dos dígitos y debe estar entre 18 y 60 años.
- ✓ El Sexo solo debe aceptar dos dígitos y debe estar entre 18 y 60 años.
- ✓ El Ruc es opcional, pero, si se ingresa se debe ingresar 11 dígitos.

Controles del formulario



A las cajas de texto debe asignarles un valor en su propiedad MaxLength para indicar la cantidad máxima de caracteres a ingresar: TxtNombre 50, TxtEdad 2, TxtSexo1 y TxtRuc11. Esto también se controla mediante el control ErrorProvider.

En este programa necesita importar Regular Expressions:



Instrucciones del Evento KeyPress de TxtNombre

'Colocamos el código en el evento KeyPress de la caja de texto TxtNombre ya que es útil para interceptar pulsaciones de teclas introducidas en un control.

```
PrivateSub TxtNombre_KeyPress(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles
TxtNombre.KeyPress

'Verifica que ingrese y devuelve un carácter de retorno

If Asc(e.KeyChar) = 13 Then

'Posiciona el cursor en la caja de texto TxtEdad

    TxtEdad.Focus()

'Cerramos el condicional

EndIf

'Salimos de la Sub-Rutina

EndSub
```

Instrucciones del Evento KeyPress de TxtEdad

'Colocamos el código en el evento KeyPress de la caja de texto TxtNombre ya que es útil para interceptar pulsaciones de teclas introducidas en un control.

```
PrivateSub TxtEdad_KeyPress(ByVal sender As System.Object, ByVal
e As System.Windows.Forms.KeyPressEventArgs) Handles
TxtEdad.KeyPress

'Verifica que ingrese y devuelve un carácter de retorno

If Asc(e.KeyChar) = 13 Then

'Posiciona el cursor en la caja de texto TxtSexo

    TxtSexo.Focus()

'Cerramos el condicional

EndIf

'Salimos de la Sub-Rutina

EndSub
```

Instrucciones del Evento KeyPress de TxtSexo

'Colocamos el código en el evento KeyPress de la caja de texto TxtSexo ya que es útil para interceptar pulsaciones de teclas introducidas en un control.

```
PrivateSub TxtSexo_KeyPress(ByVal sender As System.Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles TxtSexo.KeyPress
    'Verifica que ingrese y devuelve un carácter de retorno
    If Asc(e.KeyChar) = 13 Then
        'Posiciona el cursor en la caja de texto TxtRuc
        TxtRuc.Focus()
        'Cerramos el condicional
    EndIf
    'Salimos de la Sub-Rutina
EndSub
```

Instrucciones del Evento KeyPress de TxtRuc

'Colocamos el código en el evento KeyPress de la caja de texto TxtRuc ya que es útil para interceptar pulsaciones de teclas introducidas en un control.

```
PrivateSub TxtRuc_KeyPress(ByVal sender As System.Object, ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles TxtRuc.KeyPress
    'Verifica que ingrese y devuelve un carácter de retorno
    If Asc(e.KeyChar) = 13 Then
        'Posiciona el cursor en el botón BtnNuevo
        BtnNuevo.Focus()
        'Cerramos el condicional
    EndIf
    'Salimos de la Sub-Rutina
EndSub
```

Instrucciones del Evento Validating de TxtNombre

```

'Colocamos el código en el evento Validating de la caja de texto
TxtNombre ya que en el evento Validating verifican que la
propiedad .text no sea nula.

PrivateSub TxtNombre_Validating(ByVal sender As System.Object,
ByVal e As System.ComponentModel.CancelEventArgs) Handles
TxtNombre.Validating

'Verifica si los datos en la caja de texto TxtNombre son nulos,
vacíos y suprime ambos tipos de espacios

If TxtNombre.Text.Trim = String.Empty Then

'Si hay error, asigna el mensaje del control TxtNombre

    ErrorProv.SetError(TxtNombre, "El nombre es
Obligatorio")

'Verifica si la condición es falsa

Else

'Declara la variable ReglaNombre como una expresión regular
inmutable.

Dim ReglaNombre AsNew Regex("^[a-zA-ZñÑáéóú\s]+$")

'Verifica que solo se ingresen letras y espacios en blanco

IfNot ReglaNombre.IsMatch(TxtNombre.Text) Then

'Si hay error, asigna el mensaje del control TxtNombre

    ErrorProv.SetError(TxtNombre, "Solo letras")

'Verifica si la condición es falsa

Else

'Si no hay error, quita el mensaje del control TxtNombre

    ErrorProv.SetError(TxtNombre, "")

'Cerramos el condicional

EndIf

'Cerramos el condicional

EndIf

'Salimos de la Sub-Rutina

EndSub

```

Instrucciones del Evento Validating de TxtEdad

```

'Colocamos el código en el evento Validating de la caja de texto
TxtEdad ya que en el evento Validating verifican que la
propiedad .text no sea nula.

PrivateSub TxtEdad_Validating(ByVal sender As System.Object,
ByVal e As System.ComponentModel.CancelEventArgs) Handles
TxtEdad.Validating

'Verifica que el valor convertido a número entero ingresado en
la caja de texto TxtEdad es mayor a 18 y menor a 60

If Integer.Parse(TxtEdad.Text) < 18 Or Integer.Parse(TxtEdad.Text)
> 60 Then

'Si hay error, asigna el mensaje del control TxtEdad

    ErrorProv.SetError(TxtEdad, "La edad entre 18 y 60
años")

'Verifica si la condicion es falsa

Else

'Otra forma de borrar el emnsaje de error

    ErrorProv.SetError(TxtEdad, String.Empty)

'Cerramos el condicional

EndIf

'Salimos de la Sub-Rutina

EndSub

```

Instrucciones del Evento Validating de TxtSexo

```

'Colocamos el código en el evento Validating de la caja de texto
TxtSexo ya que en el evento Validating verifican que la
propiedad .text no sea nula.

PrivateSub TxtSexo_Validating(ByVal sender As System.Object,
ByVal e As System.ComponentModel.CancelEventArgs) Handles
TxtSexo.Validating

'Verifica que se haya ingresado las letras M o F

If TxtSexo.Text.ToUpper <>"M" And TxtSexo.Text.ToUpper <>"F" Then

'Si hay error, asigna el mensaje del control TxtSexo

    ErrorProv.SetError(TxtSexo, "Debe ingresar sólo las
letras M o F")

'Verifica si la condicion es falsa

```

```

Else
'Si no hay error, quita el mensaje del control TxtSexo
    ErrorProv.SetError(TxtSexo, "")
'Cerramos el condicional
EndIf
'Salimos de la Sub-Rutina
EndSub

```

Instrucciones del Evento Validating de TxtRuc

```

'Colocamos el código en el evento Validating de la caja de texto
TxtRuc ya que en el evento Validating verifican que la propiedad
.text no sea nula.

PrivateSub TxtRuc_Validating(ByVal sender As System.Object,
ByVal e As System.ComponentModel.CancelEventArgs) Handles
TxtRuc.Validating

'Si se ingresa el Ruc, deben ser 11 dígitos
If TxtRuc.Text.Trim <> String.Empty Then
'Declara la variable oPatronRuc como una expresión regular
inmutable.

Dim oPatronRuc AsNew Regex("^\d{11}$")
'Verifica que solo se ingresen números
IfNot oPatronRuc.IsMatch(TxtRuc.Text) Then
'Si hay error, asigna el mensaje del control TxtRuc
    ErrorProv.SetError(TxtRuc, "Sólo 11 dígitos")

'Verifica si la condición es falsa
Else
'Si no hay error, quita el mensaje del control TxtRuc
    ErrorProv.SetError(TxtRuc, "")

'Cerramos el condicional
EndIf
'Cerramos el condicional
EndIf

```

```
'Salimos de la Sub-Rutina
EndSub
```

Instrucciones del Botón BtnNuevo

'Colocamos el código en el evento Click en el botón BtnNuevo ya que el usuario hace clic en el objeto con el botón principal del mouse.

```
PrivateSub BtnNuevo_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BtnNuevo.Click
    'Verifica si se han ingresado los datos correctamente
    ForEach Ctrl As Control InMe.Controls
        'Pregunta si es una caja de Textos o MaskedTextBox
        IfTypeof Ctrl Is TextBox Then
            'Pregunta si tiene un mensaje de error
            If ErrorProv.GetError(Ctrl) <>String.Empty Then
                'Verifica que el cursor este en el Ctrl
                Ctrl.Focus()
                'Si hay error, se muestra el mensaje
                MessageBox.Show(ErrorProv.GetError(Ctrl), "Verifica",
                    MessageBoxButtons.OK, MessageBoxIcon.Error)
            EndIf
            'Continua con las instrucciones
        Next
        'Limpia el contenido de la caja de texto TxtNombre
        TxtNombre.Text = ""
        'Limpia el contenido de la caja de texto TxtEdad
        TxtEdad.Text = ""
    EndFor
EndSub
```

```

'Limpia el contenido de la caja de texto TxtRuc
TxtRuc.Text = ""

'Limpia el contenido de la caja de texto TxtSexo
TxtSexo.Text = ""

'Posiciona el cursor en la caja de texto TxtNombre
TxtNombre.Focus()

'Muestra el mensaje
    MessageBox.Show("Los datos se grabaron
correctamente", "Felicitaciones", MessageBoxButtons.OK, MessageBoxIcon.Information)

'Salimos de la Sub-Rutina
EndSub

```

CAPÍTULO Nº 5

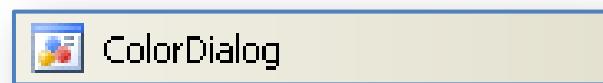
CUADROS DE DIÁLOGO

CONTENIDO:

En este capítulo usted aprenderá a ejecutar desde cualquier aplicación los diferentes cuadros de dialogo que tiene el Visual Basic.Net.

- ColorDialog
- FontDialog
- FolderBrowserDialog
- OpenFileDialog

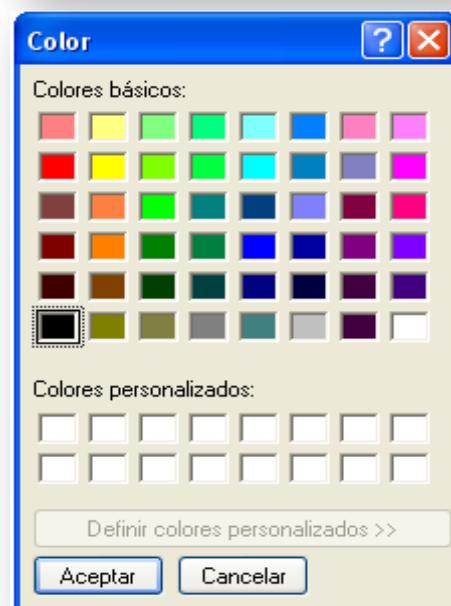
colordialog



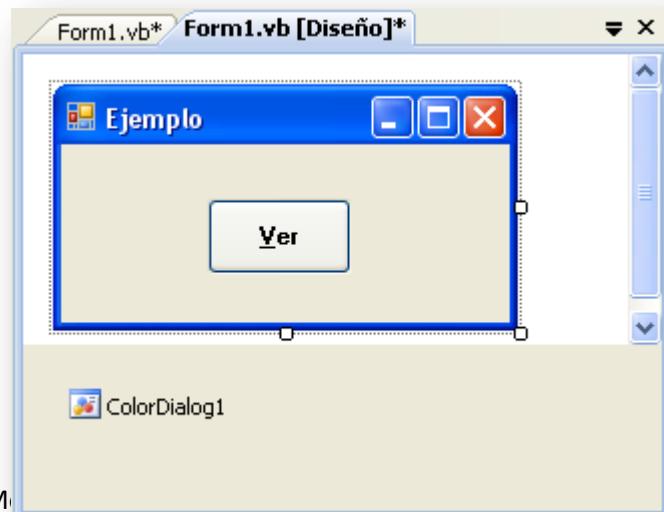
El ColorDialog muestra una cuadro de dialogo donde el usuario puede seleccionar un color o definir un nuevo color para utilizarlo en una aplicación. El cuadro de diálogo predeterminado es el siguiente:

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Button1	Name	BtnVer
	Text	&Ver
ColorDialog1	Name	ColorDialog1



Al dibujarlo en el formulario se ubica en la parte inferior del formulario.



Principales propiedades

AutoFullOpen

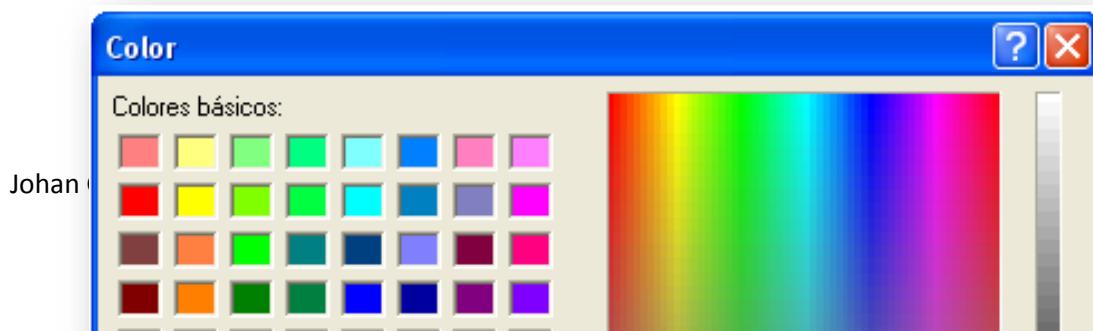
Esta propiedad permite configurar el cuadro de dialogo para que muestre o no el botón para definir colores personalizados. En el siguiente cuadro de ejemplo se asignado el valor False a esta propiedad por lo que el botón se muestra desactivado:



Si se le asigna el valor True a esta propiedad, el botón para definir colores personalizados estará activo.



La ventana para definir colores personalizados es:



En esta ventana puede hacer clic en el color que desea o escribir en su respectiva caja un valor entre 0 y 255 con el que se definirá el nuevo color.

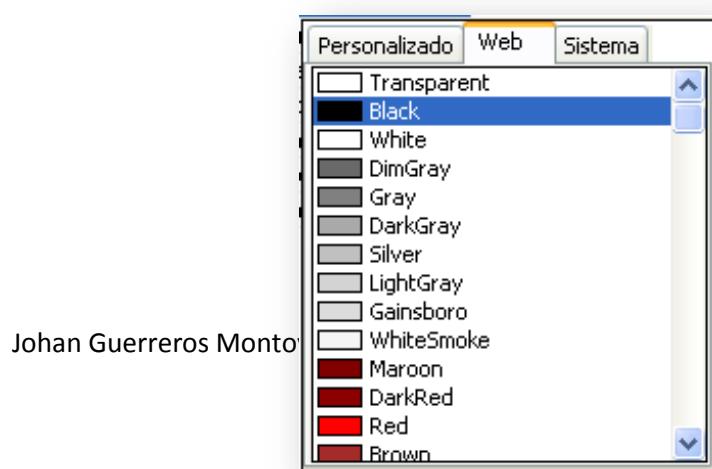
La barra de la derecha permite establecer la intensidad del color seleccionado



Color

Esta propiedad permite establecer el valor predeterminado del cuadro de dialogo y también almacena el color que el usuario selecciona del cuadro de dialogo.

Al ingresar a esta propiedad se visualiza la siguiente ventana donde se puede elegir el color que mostrará inicialmente el cuadro de dialogo.



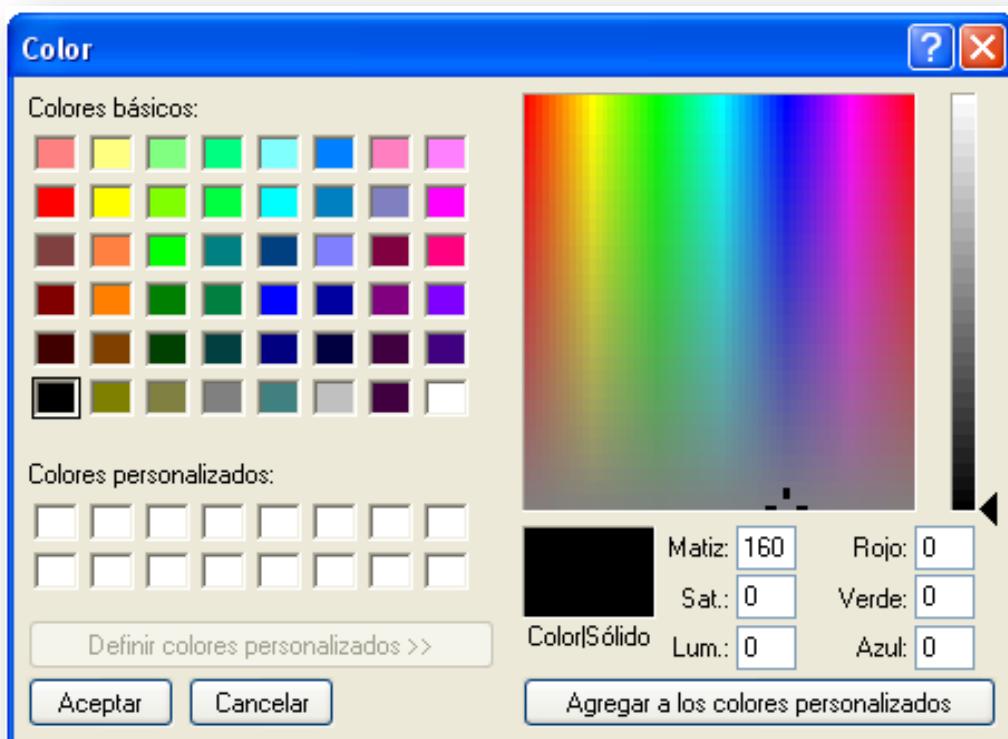
Johan Guerreros Monto

En tiempo de ejecución, esta propiedad almacena el color que el usuario selecciona después de hacer clic en el botón Aceptar.



FullOpen

Con esta propiedad se puede indicar si el cuadro de dialogo debe mostrar en forma predeterminada la sección para establecer colores personalizados. El valor True indica que debe mostrarse las secciones de colores personalizados al mostrarse el cuadro.



ShowHelp

Con esta propiedad se puede indicar si el cuadro de dialogo debe mostrar el botón Ayuda.

SolidColorOnly

Con esta propiedad se puede indicar si sólo se puede seleccionar colores sólidos.

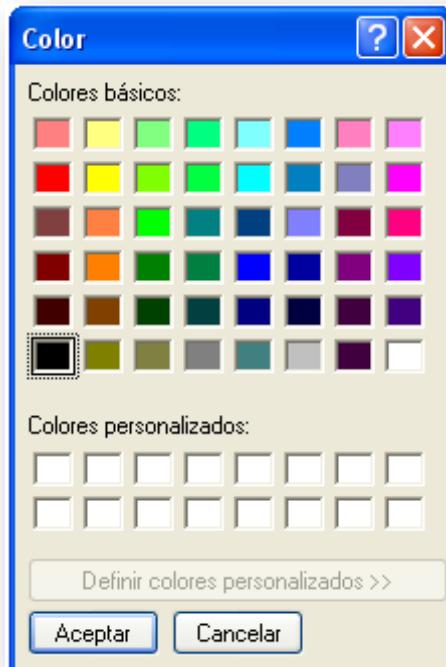
Esta control tiene el método ShowDialog que permite mostrar el cuadro de dialogo. Por ejemplo, en el siguiente programa de ejemplo se ha escrito la instrucción: **ColorDialog1.ShowDialog()** dentro del botón Ver que muestra el cuadro de dialogo establecido en el control ColorDialog1:

FORMULARIO EJEMPLO

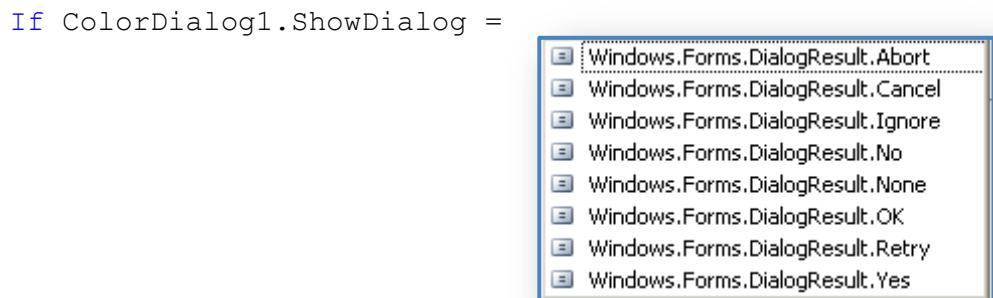
CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Button1	Name Text	BtnVer &Ver
ColorDialog1	Name	ColorDialog1



El cuadro de dialogo se muestra según los valores establecidos en sus propiedades:



Dentro de un programa podemos preguntar por el resultado de la ejecución del cuadro de dialogo para asignar o no el color establecido. Esto se puede realizar al mostrar el cuadro de dialogo, de la siguiente manera:



Por ejemplo, en la siguiente instrucción se pregunta si el usuario ha hecho clic en el botón Aceptar.

```
If ColorDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
```

El siguiente ejemplo permite mostrar el cuadro de dialogo para que el usuario seleccione un color y si pulsa la tecla Enter asigne el color seleccionado al fondo del formulario.

```
ColorDialog1.ShowDialog()
If ColorDialog1.ShowDialog = Windows.Forms.DialogResult.OK Then
    Me.BackColor = ColorDialog1.Color
EndIf
```

Aplicación Desarrollada N° V - 01

Este programa permite asignar un color a las letras o fondo de un Label.

FORMULARIO EJEMPLO

CONTROL	PROPIEDAD	VALOR
Form1	Text	Ejemplo
Label1	Name	LblTitulo
GroupBox1	Text	&Asignar Color A:
RadioButton1	Name	RbLetras
	Text	&Letras
RadioButton2	Name	RbFondo
	Text	&Fondo

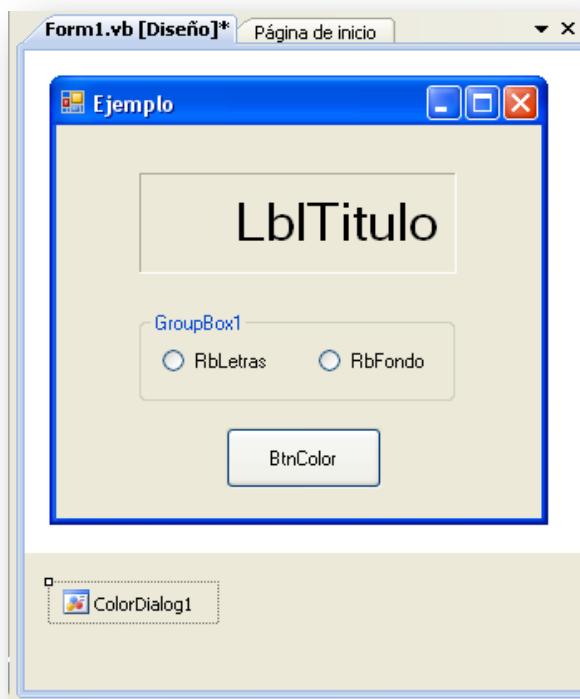
Button1	Name Text	BtnColor &Color
ColorDialog1	Name	ColorDialog1



El usuario debe seleccionar mediante controles RadioButton si desea asignar el color a las letras o al fondo del control Label. Al hacer clic en el botón Color se muestra el cuadro de dialogo y si se hace clic en el botón Aceptar se asigna el color seleccionado.



Controles del formulario



El control LblTitulo debe tener en su propiedad Text: RUTH MARIBEL y las siguientes propiedades:

AutoSize	False
----------	-------

BorderStyle	Fixed3D
-------------	---------

TextAlign	MiddleRight
-----------	-------------

El diseño del formulario debe quedar de la siguiente manera:



Instrucciones del botón BtnColor

```
'Colocamos el código dentro del evento clic del Botón BtnColor,
este se ejecutara al hacer clic en este

Private Sub BtnColor_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles BtnColor.Click

'Pregunta si se ha hecho clic en el botón Aceptar

If ColorDialog1.ShowDialog() = Windows.Forms.DialogResult.OK
Then

'Pregunta si se desea asignar el color a las letras

If RbLetras.Checked = True Then

'Asigna el color seleccionado a las letras

    LblTitulo.ForeColor = ColorDialog1.Color

'Verifica si la condicion es falsa

Else

'Asigna el color seleccionado al fondo

    LblTitulo.BackColor = ColorDialog1.Color

'Cerramos el condicional

End If

'Cerramos el condicional

End If

'Salimos de la Sub-Rutina

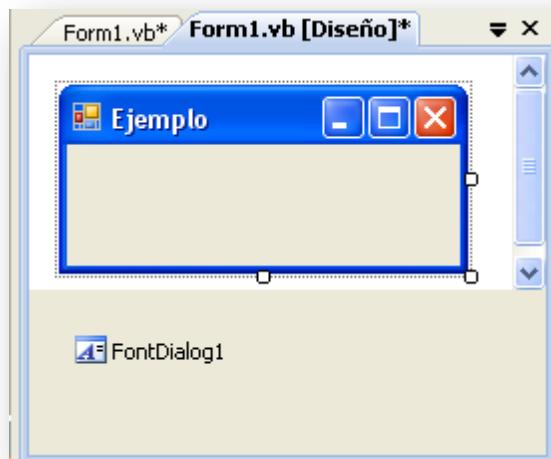
End Sub
```

FontDialog



Este control permite mostrar un cuadro de dialogo donde se puede seleccionar el tipo, estilo, tamaño y algún efecto que se le puede asignar a las letras. El cuadro de dialogo predeterminado es el siguiente:

Cuando se dibuja este control en el formulario también se ubica en la parte inferior del formulario.



Principales propiedades:

AllowScriptChange

Esta propiedad permite indicar si el usuario puede cambiar en el cuadro de dialogo el juego de caracteres. El valor True permite que el usuario pueda cambiar el juego de caracteres.

Color

Permite seleccionar el color para la letra que selecciona el usuario desde el cuadro de dialogo.

Font

Este propiedad almacena el tipo de fuentes seleccionado del cuadro de dialogo.

MaxSize

Esta propiedad se utiliza para indicar el tamaño máximo de letra que se puede seleccionar en el cuadro de dialogo.