

Proyecto Casa del libro

Documentación

25/02/2021

Erick Cumbal

Alejandro Sebastián Mihai

Contenido

[INTRODUCCIÓN 3](#_Toc65209014)

[1.1 Requisitos mínimos 3](#_Toc65209015)

[1.2 Guía de instalación 3](#_Toc65209016)

[1.3 Software utilizado 3](#_Toc65209017)

[1. Definición y planteamiento del problema 3](#_Toc65209018)

[2. OBTENCIÓN DE REQUISITOS 4](#_Toc65209019)

[2.1 Requisitos funcionales: 4](#_Toc65209020)

[2.2 Requisitos no funcionales: 4](#_Toc65209021)

[2.3 Requisitos técnicos: 5](#_Toc65209022)

[3. Casos de uso 5](#_Toc65209023)

[4. ESTUDIO DE LA BASE DE DATOS 6](#_Toc65209024)

[5. Variables usuales 7](#_Toc65209025)

[6. Juego de pruebas 8](#_Toc65209026)

[7. Diagramas de Gannt 9](#_Toc65209027)

[7.1.1 Alejandro Mihai 9](#_Toc65209028)

[7.1.2 Erick Cumbal 9](#_Toc65209029)

[8. FORMULARIOS 9](#_Toc65209030)

[8.1 Formulario InicioSesion 9](#_Toc65209031)

[8.2 Formulario MenuPrincipal 13](#_Toc65209032)

[8.3 Formulario GestionLibros 16](#_Toc65209033)

[8.4 Formulario GestionLibrosAltas 16](#_Toc65209034)

[8.5 Formulario GestionLibrosModificaciones 16](#_Toc65209035)

[8.6 Formulario GestionArticulos 16](#_Toc65209036)

[8.7 Formulario GestionArticulosAltas 16](#_Toc65209037)

[8.8 Formulario GestionArticulosModificaciones 16](#_Toc65209038)

[8.9 Formulario GestionSocios 17](#_Toc65209039)

[8.10 Formulario GestionSociosAltas 35](#_Toc65209040)

[8.11 Formulario GestionSociosModificaciones 40](#_Toc65209041)

[8.12 Formulario GestionSociosInforme 45](#_Toc65209042)

[8.13 Formulario GestionEmpleados 46](#_Toc65209043)

[8.14 Formulario GestionEmpleadosAltas 63](#_Toc65209044)

[8.15 Formulario GestionEmpleadosModificaciones 69](#_Toc65209045)

[8.16 Clase Validacion 75](#_Toc65209046)

Análisis

## INTRODUCCIÓN

El propósito de este proyecto es el diseño, la programación y explotación de un software diseñado para la empresa “La Casa del Libro”, que permita la gestión de sus empleados, de los socios, libros y artículos de una de sus tiendas.

### 1.1 Requisitos mínimos

Los requisitos mínimos contemplados que aseguran el buen funcionamiento del programa son los siguientes:

* Sistema operativo: Windows 7 o superior.
* RAM: 2 GB.
* Disco duro: 50 MB.
* Resolución de pantalla: 1600x900 píxeles.
* CPU: i3 de 2.3GHz

### 1.2 Guía de instalación

La instalación se realizará a través de un PenDrive.

Para completar la instalación con éxito se deben seguir los siguientes pasos:

1. Conectar el PenDrive
2. Ejecutar el instalador
3. Seleccionar la ruta de instalación
4. Hacer click en siguiente hasta finalizar la instalación
5. En el escritorio se habrá generado un acceso directo, clickamos
6. El usuario con rol de administrador es el siguiente:
   1. Usuario: alex
   2. Contraseña: 1234

### 1.3 Software utilizado

Se han utilizado los siguientes softwares en el desarrollo de la aplicación:

* Visual Studio 2019: Para el desarrollo de la aplicación.
  + Advanced Installer 18.0: Para empaquetar el proyecto y generar el ejecutable.
* Microsoft Access: Para almacenar la Base de Datos

## Definición y planteamiento del problema

La empresa “La Casa del Libro” precisa el desarrollo de un sistema para la gestión de uno de sus establecimientos, donde el empleado pueda controlar libros, artículos, socios y empleados, gestionando el stock de los productos.

La gestión de todos estos elementos debe realizarse a través de una base de datos que permita dar de alta, eliminar y modificar cada uno de los registros en las diferentes tablas.

Es preciso que se creen dos ficheros, uno donde se puedan almacenar los registros de los accesos a la aplicación y otro que controle errores que puedan surgir durante el uso de la aplicación por parte del usuario.

### OBTENCIÓN DE REQUISITOS

A continuación, se exponen los requisitos de la aplicación.

### 2.1 Requisitos funcionales:

|  |  |
| --- | --- |
| Código | Descripción |
| RF1 | Gestión de libros siendo posible dar de alta, eliminar y modificar un registro. |
| RF2 | Gestión de productos siendo posible dar de alta, eliminar y modificar un registro. |
| RF3 | Gestión de socios siendo posible dar de alta, eliminar y modificar un registro. |
| RF4 | Gestión de empleados siendo posible dar de alta, eliminar y modificar un registro. |
| RF5 | Simulación de la busqueda de información acerca de x cosa. |

### 2.2 Requisitos no funcionales:

|  |  |
| --- | --- |
| Código | Descripción |
| RNF1 | Se deberá poder volver al menú principal desde cualquier ventana de la aplicación. |
| RNF2 | El menú principal de la aplicación debe ser sencillo y bonito a la vista. También debe permitir el acceso al resto de pantallas. |
| RNF3 | Se podrán generar informes impresos sobre los libros, los productos y los socios. |
| RNF4 | Se deberán programar ayudas complementarias al usuario. |
| RNF5 | Se deberá habilitar la posibilidad de ejecutar una calculadora para realizar cálculos en el momento. |
| RNF6 | Deberá contener acceso a direcciones de Internet de interés. |
| RNF7 | Se deberán controlar todos los errores que puedan surgir durante su ejecución. |
| RNF8 | Es necesario controlar y registrar todos los accesos a la aplicación |

### 2.3 Requisitos técnicos:

|  |  |
| --- | --- |
| Código | Descripción |
| RT1 | Se deberá empaquetar el proyecto para mayor facilidad de uso. |
| RT2 | Se deberán guardar los errores en un fichero aparte. |
| RT3 | Se deberán guardar los accesos en un fichero aparte. |

## Casos de uso

Los siguientes cuatro formularios tienen el mismo flujo principal:

Gestionar libros, artículos, socios y empleados

|  |  |
| --- | --- |
| Flujo Principal | 1. El usuario utiliza los filtros de búsqueda para buscar. 2. No encuentra lo que busca y lo añade pulsando el botón “Añadir” 3. Rellena todos los datos y confirma la operación 4. Decide modificar el registro que acaba de crear, le da al botón modificar y modifica los valores que quiere. 5. Completa la operación, pero ya no quiere ese registro asi que lo selecciona y presiona el botón “Eliminar” |

CHM, ayuda para usuario.

|  |  |
| --- | --- |
| Pre-Condición | Encontrarse en un formulario desde el cual se pueda acceder a ella. |
| Flujo Principal | Para cualquiera de los formularios anteriores, si el usuario necesita ayuda, puede pulsar “F1” para leer una ayuda o hacer lo mismo pulsando en la barra superior el botón “Ayuda”. |
| Post-Condición | Cerrar la ayuda. |

Generación de informes (Libros, Articulos, Socios)

|  |  |
| --- | --- |
| Pre-Condición | El usuario debe ser capaz de acceder a toda la funcionalidad del programa, así que debe estar con la sesión iniciada. |
| Flujo Principal | Debe presionar el botón “Informe” para generar el informe. |
| Post-Condición |  |

Archivo de gestión de accesos

|  |  |
| --- | --- |
| Pre-Condición | Que un usuario inicie sesión. |
| Flujo Principal | Tras iniciar sesión, su usuario y fecha se guarda en un txt |
| Post-Condición | Cerrar el fichero tras terminar la operación. |

Archivo de gestión de errores

|  |  |
| --- | --- |
| Pre-Condición | Que un error salte. |
| Flujo Principal | El catch ejecutará un método que registrará su mensaje, fecha y hora. |
| Post-Condición | Cerrar el fichero tras terminar la operación. |

## ESTUDIO DE LA BASE DE DATOS

La base de datos denominada “cas\_lib\_db”, creada en Access posee la siguiente estructura:

Tablas:

* libros

Campos:

* + ISBN: *Clave primaria*, Númerico longitud 13
  + Titulo del libro : Texto longitud 100
  + Autor/es : Texto longitud 50
  + Número de páginas : Númerico longitud 4
  + Editorial : Texto longitud 50
  + Idioma : Texto longitud 50
  + Encuadernacion : Texto longitud 20
  + Plaza de edición : Texto longitud 4
  + Traductor : Texto longitud 59
  + Formato : Texto longitud 15
  + Precio : Texto longitud 6
  + Stock : Númerico longitud 4
* articulos

Campos:

* + ISBN *: Clave Primaria*, Númerico longitud 13
  + Nombre : Texto longitud 50
  + Categoria : Texto longitud 50
  + Stock : Númerico longitud 4
  + Precio : Númerico longitud 6
* socios

Campos:

* + Numero de socio : *Clave Primaria*, Númerico longitud 8
  + Nombre : Texto longitud 50
  + Apellidos : Texto longitud 50
  + Telefono : Númerico longitud 9
  + Correo : Texto longitud 120
* empleados

Campos:

* + DNI: *Clave Primaria*, Texto longitud 9
  + Nombre : Texto longitud 50
  + Apellidos : Texto longitud 50
  + Correo : Texto longitud 120
  + Telefono: Númerico longitud 9
  + Usuario : Texto longitud 30
  + Contraseña : Númerico longitud 4
  + Roles : Texto longitud 20

## Variables usuales

Aquí explicamos las variables que suelen aparecer de forma recurrente en los formularios.

|  |  |  |  |
| --- | --- | --- | --- |
| **Variable** | **Tipo** | **Nivel de visibilidad** | **Descripción** |
| conexion | OleDbConnection | local | Para la conexión |
| adaptador | OleDbDataAdapter | local | Para los datos |
| midataset | DataSet | local | Set de datos |
| comando | OleDbCommand | local | Comando |
|  | local | local |  |
|  |  |  |  |
|  |  |  |  |

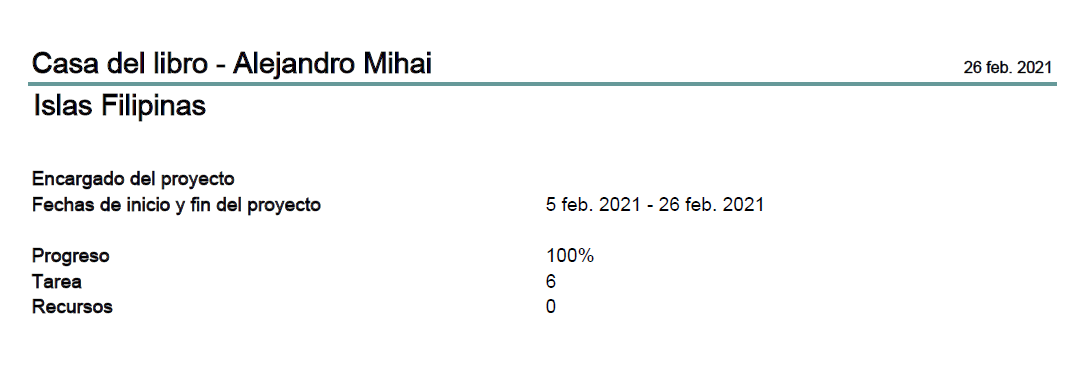
## Juego de pruebas

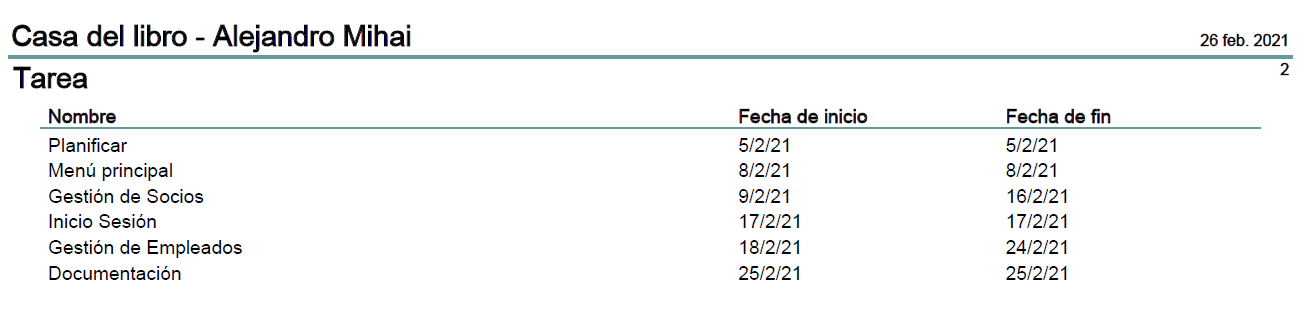
En todos los textBoxes se ha realizad la prueba con código 1V siguiente:

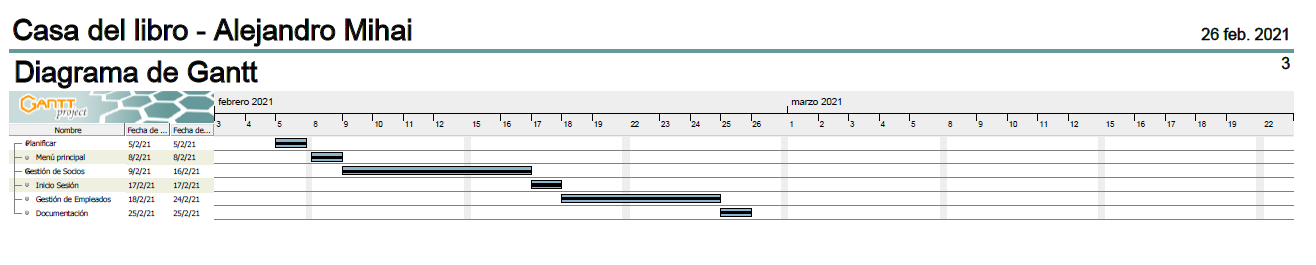
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Codigo** | **Descripción** | **Datos introducidos** | **Resultado esperado** | **Resultado obtenido** |
| 1V | Se intenta provocar excepción introduciendo caracteres incorrectos. | Caracteres prohibidos, incorrectos | Excepción tratada por los métodos de validación | Excepción tratada por los métodos de validación |
| 2V | Se intenta (en todos los filtros de busqueda), dejar todo en blanco y buscar | Nnguno | Menaaje de no se puede, es necesario al menos un dato. | Menaaje de no se puede, es necesario al menos un dato. |
| 3V | Al añadir se intenta dejar un campo obligatorio con menos caracteres que los que necesita | Aquellos que estén admitidos por el textBox en concreto | Mensaje avisando al usuario de que debe rellenar la longitud completa de la caja | Mensaje avisando al usuario de que debe rellenar la longitud completa de la caja |
| 4V | Se pulsa tres veces el botón de mostrar calculadora | Ninguno | Que se ejecute la calculadora de Windows 2 veces y a la tercera salte un mensaje avisando de que no se puede más de 2 | El esperado. |

## Diagramas de Gannt

### Alejandro Mihai







### Erick Cumbal

## 8. FORMULARIOS

### 8.1 Formulario InicioSesion

#### Explicación de formulario

Este formulario permite que el usuario inice sesión con su correspondiente cuenta.

Se utiliza para evitar que terceras personas puedan acceder a la aplicación sin autorización expresa del administrador.

Para iniciar sesión, el usuario debe introducir un nombre de usuario y una contraseña. Tras pulsar el botón “Iniciar Sesión”, la aplicación contactará con la base de datos y cotejara los datos introducidos en busca de coincidencias.

En caso contrario, aparecerá un mensaje que informará al usuario de que el nombre de usuario o la contraseña eran incorrectos.

En caso afirmativo, el estado del formulario (que el usuario puede observar en la zona inferior) pasará a “en ejecución” y la aplicación procederá a cargar los datos y formularios necesarios (carga representada de forma gráfica por la barra inferior).

Tras completarse el proceso, la aplicación dará paso al usuario al menú principal, desde donde el usuario podrá acceder a las funcionalidades del programa.

Si el usuario desea detener la ejecución del programa, solo debe pulsar el botón “Salir”, lo que mostrara un mensaje de confirmación, actuando acorde a lo que seleccione el usuario.



|  |
| --- |
| Imports System.Data.OleDb  Public Class Pantalla\_Inicio  Public conexion As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=Repsol\_db.accdb")  Public adaptador As New OleDbDataAdapter("Select \* from empleados", conexion)  Public midataset As New DataSet  Public activo As Boolean = True  ' Instancia de la clase que gestiona los errores  Public errores As New GestionErrores  ' Instancia de la clase que gestiona los errores  Public guardarErrores As New ErroresApp  ' Instanciamos clase de las validaciones  Public validaciones As Validaciones = New Validaciones  'Instanciamos clase que se ocupa de guardar los accesos  Public logueado As AccesosApp = New AccesosApp  Private Sub BtLogin\_Click(sender As Object, e As EventArgs) Handles btLogin.Click  'MenuPrincipal.Show()  'Me.Hide()  VerificarLogeo(tbUsuarioInicio.Text, tbContraInicio.Text)  End Sub  Private Sub VerificarLogeo(ByVal user As String, ByVal contrasena As String)  Try  adaptador = New OleDbDataAdapter("Select usuario from empleados WHERE usuario='" + user + "' AND contrasena='" + contrasena + "'", conexion)  adaptador.Fill(midataset, "empleados")  If midataset.Tables("empleados").Rows.Count = 1 Then  ' Guardamos datos del acceso en el archivo  logueado.accesosApp("Login OK. usuario: " & tbUsuarioInicio.Text & " contraseña: " & tbContraInicio.Text)  ' Dejamos limpios los campos  tbUsuarioInicio.Text = ""  tbContraInicio.Text = ""  ' Cambiamos de pantalla  activo = False  midataset.Tables("empleados").Clear()  ProgressBar1.Visible = True  'Bucle que se recorre 100 veces  For i As Integer = 1 To 100  'Aumentamos un paso  ProgressBar1.PerformStep()  Next  MsgBox("Entrando en la aplicación")  Me.Hide()  MenuPrincipal.Show()  Else  ' Guardamos datos del acceso en el archivo  logueado.accesosApp("Login Erroneo. usuario: " & tbUsuarioInicio.Text & " contraseña: " & tbContraInicio.Text)  ' Mostramos mensaje de error  MsgBox("El usuario o contraseña introducido es incorrecto, por favor introduzca otro", MsgBoxStyle.Information, "Error en la verificación")  End If  Catch ex As Exception  guardarErrores.erroresApp("Excepción controlada nº" & Err.Number & " : " & ex.Message & " generada en el formulario de Gestión de Empleados")  Dim encontradoError As Boolean = errores.mensajeError(Err.Number, "Por favor, revise la base de datos. ", False)  If encontradoError = False Then  MsgBox("Error nº: " & Err.Number & ". " & ex.Message & " Por favor, revise la base de datos. ", MsgBoxStyle.Exclamation, ex.Message)  End If  End Try  End Sub  Public Sub ActivDesactivBotones()  If tbUsuarioInicio.Text.Length > 0 And tbContraInicio.Text.Length > 0 Then  btLogin.Enabled = True  Else  btLogin.Enabled = False  End If  End Sub  Private Sub BtSalir\_Click(sender As Object, e As EventArgs) Handles btSalir.Click  If MsgBox("¿Esta seguro de querer salir?", 36, "Salir") = 6 Then  Application.Exit()  End If  End Sub  Private Sub TbUsuarioInicio\_TextChanged(sender As Object, e As EventArgs) Handles tbUsuarioInicio.TextChanged  If tbUsuarioInicio.Text.Length > 0 Then  Dim validacion As String = validaciones.soloLetrasYnumeros(tbUsuarioInicio.Text, Nothing)  If validacion <> tbUsuarioInicio.Text Then  tbUsuarioInicio.Text = validacion  tbUsuarioInicio.Select(tbUsuarioInicio.Text.Length, tbUsuarioInicio.Text.Length)  End If  End If  ActivDesactivBotones()  End Sub  Private Sub TbContraInicio\_TextChanged(sender As Object, e As EventArgs) Handles tbContraInicio.TextChanged  If tbContraInicio.Text.Length > 0 Then  Dim validacion As String = validaciones.soloLetrasYnumeros(tbContraInicio.Text, Nothing)  If validacion <> tbContraInicio.Text Then  tbContraInicio.Text = validacion  tbContraInicio.Select(tbContraInicio.Text.Length, tbContraInicio.Text.Length)  End If  End If  ActivDesactivBotones()  End Sub  Private Sub Timer1\_Tick(sender As Object, e As EventArgs) Handles Timer1.Tick  End Sub  Private Sub Pantalla\_Inicio\_Load(sender As Object, e As EventArgs) Handles MyBase.Load  ProgressBar1.Visible = False  End Sub  End Class |

### 8.2 Formulario MenuPrincipal

#### Explicación de formulario

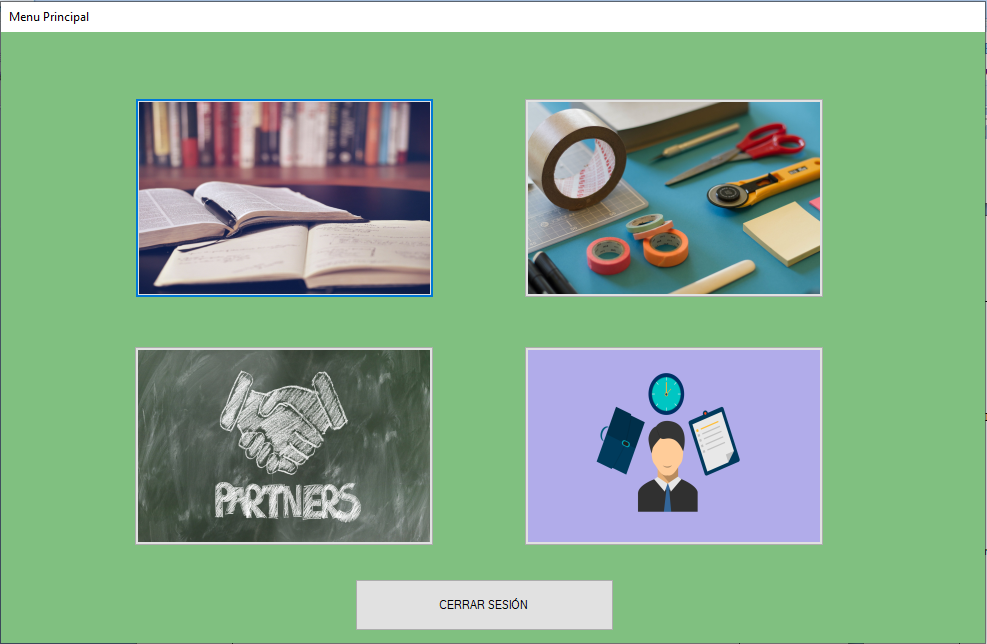
Este formulario permite al usuario navegar entre los diferentes formularios que componen la aplicación.

Los formularios a los que el usuario puede acceder son:

* Gestión de libros
* Gestión de artículos
* Gestión de socios
* Gestión de empleados

Dependiendo de su nivel de autorización (Admin – Encargado - Empleado), el usuario podrá realizar una serie de operaciones u otras.

En caso de querer cerrar sesión para dar paso a otro usuario, el usuario solo debe pulsar el botón “CERRAR SESIÓN”, lo que cerrará el formulario actual y dara paso al formulario de inicio de sesión.



|  |
| --- |
| ' Necesitamos importar el módelo de base de datos que vamos a utilizar, este es de access.  Imports System.Data.OleDb  Public Class MenuPrincipal  ' Especificamos la base de datos a la que nos vamos a conectar.  Public conexion As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=cas\_lib\_dib.accdb")  ' Al adaptador le asignamos la conexion que acabamos de realizar y una consulta  Public adaptador As New OleDbDataAdapter("Select \* from Socios", conexion)  ' Aquí alojaremos los datos de la DB  Public midataset As New DataSet  'Metodo que detecta que botón se ha pulsado y envia la orden de posicionar a:  'GestionArticulos/GestionLibros/GestionEmpleados/GestionSocios  Private Sub posicionarFormulariosGestiones(ByVal btn\_Name As String)  'Especifica que el tipo de posicion de arranque será manual para Gestión Articulos  GestionArticulos.StartPosition = FormStartPosition.Manual  'Especifica que el tipo de posicion de arranque será manual para Gestión Libros  GestionLibros.StartPosition = FormStartPosition.Manual  'Especifica que el tipo de posicion de arranque será manual para Gestión Empleados  GestionEmpleados.StartPosition = FormStartPosition.Manual  'Especifica que el tipo de posicion de arranque será manual para Gestión Socios  GestionSocios.StartPosition = FormStartPosition.Manual  ' Variable tipo Integer que calculará la posición vertical en la que ubicará el nuevo formulario  Dim a As Integer  a = My.Computer.Screen.Bounds.Size.Width - (My.Computer.Screen.Bounds.Size.Width \* 0.97)  ' Variable tipo Integer que calculará la posición horizontal en la que ubicará el nuevo formulario  Dim b As Integer  b = My.Computer.Screen.Bounds.Size.Width - (My.Computer.Screen.Bounds.Size.Width \* 0.97)  'Condición que compará el nombre del botón recibido como parametro con la cadena correspondiente a dicho botón.  If btn\_Name.Equals("button\_gestionlibros") Then  GestionLibros.Location = New Point(a, b) 'Posiciona en las coordenadas a y b  ElseIf btn\_Name.Equals("button\_gestionpapeleria") Then  GestionArticulos.Location = New Point(a, b) 'Posiciona en las coordenadas a y b  ElseIf btn\_Name.Equals("button\_gestionempleados") Then  GestionEmpleados.Location = New Point(a, b) 'Posiciona en las coordenadas a y b  ElseIf btn\_Name.Equals("button\_gestionsocios") Then  GestionSocios.Location = New Point(a, b) 'Posiciona en las coordenadas a y b  End If  End Sub  ' Método que se ejecuta si el botón libros es pulsado, mostrando el formulario de gestión de libros  Private Sub Button\_GestionLibros\_Click(sender As Object, e As EventArgs) Handles Button\_GestionLibros.Click  ' Comprobación de que la base de datos es accesible obligando a acceder a datos, en caso contrario, error  Try  'Variable que nos dice cual es el Nombre del botón pulsado  Dim btn\_Name As String = DirectCast(sender, Button).Name.ToLower.Replace("clear", "")  ' Cargar la memoria del cache con datos.  adaptador.Fill(midataset, "Socios")  ' Re-posicionamos el formulario donde lo queremos  posicionarFormulariosGestiones(btn\_Name)  ' Mostramos el formulario de gestión de libros  GestionLibros.Show()  ' Ocultamos el formulario de menú principal  Me.Hide()  Catch ex As System.Data.OleDb.OleDbException  MsgBox("Parece que algo ha salido mal. Revise que la base de datos no esté abierta durante la ejecución.", MsgBoxStyle.OkOnly, "Error - Base de datos")  End Try  End Sub  ' Método que se ejecuta si el botón papeleria es pulsado, mostrando el formulario de gestión de papeleria  Private Sub Button\_GestionPapeleria\_Click(sender As Object, e As EventArgs) Handles Button\_GestionPapeleria.Click  ' Comprobación de que la base de datos es accesible obligando a acceder a datos, en caso contrario, error  Try  'Variable que nos dice cual es el Nombre del botón pulsado  Dim btn\_Name As String = DirectCast(sender, Button).Name.ToLower.Replace("clear", "")  ' Cargar la memoria del cache con datos.  adaptador.Fill(midataset, "Socios")  ' Re-posicionamos el formulario donde lo queremos  posicionarFormulariosGestiones(btn\_Name)  ' Mostramos el formulario de gestión de papeleria  GestionArticulos.Show()  ' Ocultamos el formulario de menú principal  Me.Hide()  Catch ex As System.Data.OleDb.OleDbException  MsgBox("Parece que algo ha salido mal. Revise que la base de datos no esté abierta durante la ejecución.", MsgBoxStyle.OkOnly, "Error - Base de datos")  End Try  End Sub  ' Método que se ejecuta si el botón socios es pulsado, mostrando el formulario de gestión de socios  Private Sub Button\_GestionSocios\_Click(sender As Object, e As EventArgs) Handles Button\_GestionSocios.Click  ' Comprobación de que la base de datos es accesible obligando a acceder a datos, en caso contrario, error  Try  'Variable que nos dice cual es el Nombre del botón pulsado  Dim btn\_Name As String = DirectCast(sender, Button).Name.ToLower.Replace("clear", "")  ' Cargar la memoria del cache con datos.  adaptador.Fill(midataset, "Socios")  ' Re-posicionamos el formulario donde lo queremos  posicionarFormulariosGestiones(btn\_Name)  ' Mostramos el formulario de gestión de socios  GestionSocios.Show()  ' Ocultamos el formulario de menú principal  Me.Hide()  Catch ex As System.Data.OleDb.OleDbException  MsgBox("Parece que algo ha salido mal. Revise que la base de datos no esté abierta durante la ejecución.", MsgBoxStyle.OkOnly, "Error - Base de datos")  End Try  End Sub  ' Método que se ejecuta si el botón empleados es pulsado, mostrando el formulario de gestión de empleados  Private Sub Button\_GestionEmpleados\_Click(sender As Object, e As EventArgs) Handles Button\_GestionEmpleados.Click  ' Comprobación de que la base de datos es accesible obligando a acceder a datos, en caso contrario, error  Try  'Variable que nos dice cual es el Nombre del botón pulsado  Dim btn\_Name As String = DirectCast(sender, Button).Name.ToLower.Replace("clear", "")  ' Cargar la memoria del cache con datos.  adaptador.Fill(midataset, "Socios")  ' Re-posicionamos el formulario donde lo queremos  posicionarFormulariosGestiones(btn\_Name)  ' Mostramos el formulario de gestión de empleados  GestionEmpleados.Show()  ' Ocultamos el formulario de menú principal  Me.Hide()  Catch ex As System.Data.OleDb.OleDbException  MsgBox("Parece que algo ha salido mal. Revise que la base de datos no esté abierta durante la ejecución.", MsgBoxStyle.OkOnly, "Error - Base de datos")  End Try  End Sub  ' Método que se ejecuta si el botón "CerrarSesion" es pulsado, mostrando de nuevo el formulario de inicio de sesión  Private Sub Button\_CerrarSesion\_Click(sender As Object, e As EventArgs) Handles Button\_CerrarSesion.Click  ' Mostramos el formulario de inicio de sesión  InicioSesion.Show()  ' Cerramos el formulario de menú principal  Me.Close()  End Sub  End Class |

### 8.3 Formulario GestionLibros

#### Explicación de formulario

|  |
| --- |
|  |

### 8.4 Formulario GestionLibrosAltas

#### Explicación de formulario

|  |
| --- |
|  |

### 8.5 Formulario GestionLibrosModificaciones

#### Explicación de formulario

|  |
| --- |
|  |

### 8.6 Formulario GestionArticulos

#### Explicación de formulario

|  |
| --- |
|  |

### 8.7 Formulario GestionArticulosAltas

#### Explicación de formulario

|  |
| --- |
|  |

### 8.8 Formulario GestionArticulosModificaciones

#### Explicación de formulario

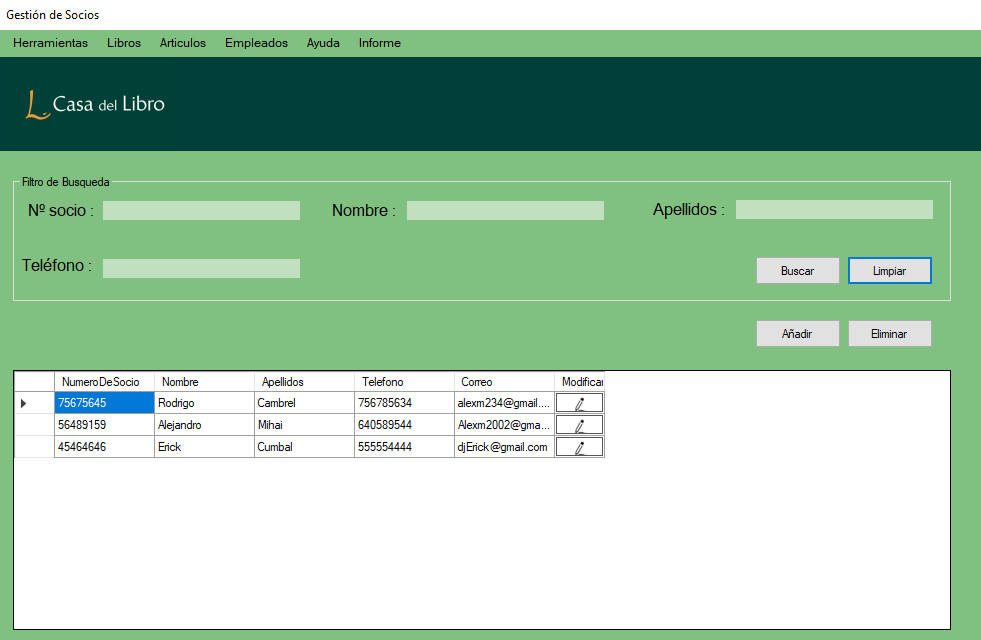
|  |
| --- |
|  |

### 8.9 Formulario GestionSocios

#### Explicación de formulario

Este formulario permite al usuario realizar una búsqueda de socios en la base de datos, visualizar los socios existentes, añadir nuevos socios, eliminarlos y modificarlos.

Además, también permite crear un informe si el usuario pulsa en el botón de la barra superior “Informe”



|  |
| --- |
| ' Necesitamos importar el módelo de base de datos que vamos a utilizar  Imports System.Data.OleDb  ' No hace falta hacer imports libValidaciones para instanciar sus clases porque está incluida en el proyecto.  Imports System.IO  Public Class GestionSocios  ' Especificamos la base de datos a la que nos vamos a conectar.  Public conexion As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=cas\_lib\_dib.accdb")  ' Al adaptador le asignamos la conexion que acabamos de realizar y una consulta  Public adaptador As New OleDbDataAdapter("Select \* from Socios", conexion)  'Declaramos el botón Modificar que será creado dinamicamente junto a los resgistros del DataGridView.  Public button\_din\_modificar As New DataGridViewButtonColumn  'Creamos in estilo que aplicaremos a las caracteristicas del botón Modificar.  Public style As New DataGridViewCellStyle  ' Aquí alojaremos los datos de la DB  Public midataset As New DataSet  ' Número de control para controlar el dataBinding de los text boxes del formulario modificaciones, evitando que se relacionen dos veces.  Public numeroDeControlBindingModificaciones As Long  Public numeroDeControlBindingAltaSocios As Long  Public posicionDataGridSeleccionada As Integer  ' Método que se ejecuta al iniciarse el formulario  Private Sub GestionSocios\_Load(sender As Object, e As EventArgs) Handles MyBase.Load  Try  ' Cargar la memoria del cache con datos.  adaptador.Fill(midataset, "Socios")  Me.TextBox\_NumeroSocioOCULTO.DataBindings.Add("text", midataset, "Socios.NumeroDeSocio")  ' cargar en el datagridview, le decimos de donde sacamos los datos  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  ' Inicializamos el número de control  numeroDeControlBindingModificaciones = 0  numeroDeControlBindingAltaSocios = 0  ' Inicializamos su valor  controlCalculadora = 0  'Creación en la ultima columna del DataGridView el botón de modificar en cada registro.  crearButtonDataGridView()  Catch ex As System.Data.OleDb.OleDbException  MsgBox("Parece que algo ha salido mal. Revise que la base de datos no esté abierta durante la ejecución.", MsgBoxStyle.OkOnly, "Error - Base de datos")  ' Mostramos el menú principal.  MenuPrincipal.Show()  ' Cerramos este formulario  Me.Close()  End Try  End Sub  ' Método que se ejecuta al pulsar en una de las cajas del DataGridView  Private Sub DataGridView\_Socios\_CellClick(sender As Object, e As DataGridViewCellEventArgs) Handles DataGridView\_Socios.CellClick  posicionDataGridSeleccionada = BindingContext(midataset, "Socios").Position  Button\_Eliminar.Enabled = True  End Sub  ' Método que permite posicionar la ventana en la posición especificada del formulario "GestionSociosAltas".  ' En este caso para evitar que quede encima del formulario anterior.  Private Shared Sub posicionarGestionAltas()  GestionSociosAltas.StartPosition = FormStartPosition.Manual  Dim a As Integer  a = My.Computer.Screen.Bounds.Size.Width - (My.Computer.Screen.Bounds.Size.Width \* 0.55)  Dim b As Integer  b = My.Computer.Screen.Bounds.Size.Height - (My.Computer.Screen.Bounds.Size.Height \* 0.8)  GestionSociosAltas.Location = New Point(a, b)  End Sub  ' Método que permite posicionar la ventana en la posición especificada del formulario "GestionSociosModificaciones".  ' En este caso para evitar que quede encima del formulario anterior.  Private Shared Sub posicionarGestionModificaciones()  GestionSociosModificaciones.StartPosition = FormStartPosition.Manual  Dim a As Integer  a = My.Computer.Screen.Bounds.Size.Width - (My.Computer.Screen.Bounds.Size.Width \* 0.55)  Dim b As Integer  b = My.Computer.Screen.Bounds.Size.Height - (My.Computer.Screen.Bounds.Size.Height \* 0.8)  GestionSociosModificaciones.Location = New Point(a, b)  End Sub  ' Método que se ejecuta cuando el botón "Añadir" es pulsado.  Private Sub Button\_Annadir\_Click(sender As Object, e As EventArgs) Handles Button\_Annadir.Click  ' Posicionamos el formulario que vamos a mostrar.  posicionarGestionAltas()  ' Mostramos el formulario  GestionSociosAltas.ShowDialog()  End Sub  'Método que crea y da formato al botón de modificar, en cada una de los registros del DataGridView.  'Este bóton abrirá el formulario de GestionSociosModificaciones.  Public Sub crearButtonDataGridView()  'style.BackColor when the button Is Not selected  style.BackColor = Color.White  'Le asignamos un nombre  button\_din\_modificar.Name = "Modificar"  'Le asignamos un tamaño  button\_din\_modificar.Width = 50  'declare a DataGridViewCellStyle  button\_din\_modificar.FlatStyle = FlatStyle.Popup  'Cargar en la ultima columna del DataGridView, la creación y el estilo del botón interno.  DataGridView\_Socios.Columns.Add(button\_din\_modificar)  End Sub  'Metodo evento que capta la puslación en la celda relativa al botón.  Private Sub DataGridView1\_CellContentClick(sender As System.Object, e As DataGridViewCellEventArgs) Handles DataGridView\_Socios.CellContentClick  Try  'Convierte el objeto en sender  Dim senderGrid = DirectCast(sender, DataGridView)  'Comprueba que es una columna del data gridview que tiene el evento y que tiene indice mayor que 0  'Si es correcto se ejecutará el comando de abrir GestionSociosModificaciones  If TypeOf senderGrid.Columns(e.ColumnIndex) Is DataGridViewButtonColumn AndAlso e.RowIndex >= 0 Then  ' Posicionamos el formulario que vamos a mostrar.  posicionarGestionModificaciones()  ' Mostramos el formulario  GestionSociosModificaciones.ShowDialog()  End If  Catch ex As System.ArgumentException  MsgBox("Uno de los argumentos ha fallado, intentelo de nuevo.", MsgBoxStyle.OkOnly, "Error - Argumento")  End Try  End Sub  'Metodo que pinta el Icono asociado al botón dinámico Modificar, el la ultima Columna del DataGridView  Private Sub DataGridView1\_CellPainting(ByVal sender As Object, ByVal e As DataGridViewCellPaintingEventArgs) Handles DataGridView\_Socios.CellPainting  If e.ColumnIndex = 5 AndAlso e.RowIndex >= 0 Then  e.Paint(e.CellBounds, DataGridViewPaintParts.All)  'Liena de código que calcula la posicion de el dibujo en concreto.  e.Graphics.DrawImage(My.Resources.modificar\_ico, CInt((e.CellBounds.Width / 2) - (My.Resources.modificar\_ico.Width / 2)) + e.CellBounds.X, CInt((e.CellBounds.Height / 2) - (My.Resources.modificar\_ico.Height / 2)) + e.CellBounds.Y)  e.Handled = True  End If  End Sub  'Metodo que detecta que botón se ha pulsado y envia la orden de posicionar a:  'GestionLibros/GestionEmpleados/GestionSocios  Private Sub posicionarFormulariosGestiones(ByVal btn\_Name As String)  'Especifica que el tipo de posicion de arranque será manual para Gestión Libros  GestionLibros.StartPosition = FormStartPosition.Manual  'Especifica que el tipo de posicion de arranque será manual para Gestión Empleados  GestionEmpleados.StartPosition = FormStartPosition.Manual  'Especifica que el tipo de posicion de arranque será manual para Gestión Articulos  GestionArticulos.StartPosition = FormStartPosition.Manual  'Varaible tipo Integer que calculará la posición vertical en la que ubicará el nuevo formulario  Dim a As Integer  a = My.Computer.Screen.Bounds.Size.Width - (My.Computer.Screen.Bounds.Size.Width \* 0.97)  'Varaible tipo Integer que calculará la posición horizontal en la que ubicará el nuevo formulario  Dim b As Integer  b = My.Computer.Screen.Bounds.Size.Width - (My.Computer.Screen.Bounds.Size.Width \* 0.97)  'Condición que compará el nombre del botón recibido como parametro con la cadena correspondiente a dicho botón.  If btn\_Name.Equals("gestiónlibrostoolstripmenuitem1") Then  GestionLibros.Location = New Point(a, b) 'Posiciona en las coordenadas a y b  ElseIf btn\_Name.Equals("gestiónempleadostoolstripmenuitem1") Then  GestionEmpleados.Location = New Point(a, b) 'Posiciona en las coordenadas a y b  ElseIf btn\_Name.Equals("gestiónarticulostoolstripmenuitem1") Then  GestionArticulos.Location = New Point(a, b) 'Posiciona en las coordenadas a y b  End If  End Sub  ' Método que se ejecuta cuando es pulsado el botón "Gestion de Libros" del menuStrip  Private Sub GestiónLibrosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestiónLibrosToolStripMenuItem1.Click  'Variable que nos dice cual es el Nombre del ToolStripMenuItem pulsado  Dim item\_Name As String = DirectCast(sender, ToolStripMenuItem).Name.ToLower.Replace("clear", "")  'Confirmación sobre cambio de formulario  If MsgBox("¿Está seguro de que desea cambiar de formulario?", 36, "Salir") = 6 Then  Me.Hide() 'Ocultamos este formulario  'Invocamos al método que posiciona el Formulario de Gestion  posicionarFormulariosGestiones(item\_Name)  GestionLibros.Show()  End If  End Sub  ' Método que se ejecuta cuando es pulsado el botón "Gestion de Socios" del menuStrip  Private Sub GestiónArticulosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestiónArticulosToolStripMenuItem1.Click  'Variable que nos dice cual es el Nombre del ToolStripMenuItem pulsado  Dim item\_Name As String = DirectCast(sender, ToolStripMenuItem).Name.ToLower.Replace("clear", "")  'Confirmación sobre cambio de formulario  If MsgBox("¿Está seguro de que desea cambiar de formulario?", 36, "Salir") = 6 Then  Me.Hide() 'Ocultamos este formulario  'Invocamos al método que posiciona el Formulario de Gestion  posicionarFormulariosGestiones(item\_Name)  GestionArticulos.Show()  End If  End Sub  ' Método que se ejecuta cuando es pulsado el botón "Gestion de Empleados" del menuStrip  Private Sub GestiónEmpleadosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestiónEmpleadosToolStripMenuItem1.Click  'Variable que nos dice cual es el Nombre del ToolStripMenuItem pulsado  Dim item\_Name As String = DirectCast(sender, ToolStripMenuItem).Name.ToLower.Replace("clear", "")  'Confirmación sobre cambio de formulario  If MsgBox("¿Está seguro de que desea cambiar de formulario?", 36, "Salir") = 6 Then  Me.Hide() 'Ocultamos este formulario  'Invocamos al método que posiciona el Formulario de Gestion  posicionarFormulariosGestiones(item\_Name)  GestionEmpleados.Show()  End If  End Sub  Dim controlCalculadora As Integer  ' Método que se ejecuta cuando es pulsado el botón "Calculadora" del menuStrip  Private Sub CalculadoraToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles CalculadoraToolStripMenuItem.Click  If controlCalculadora < 2 Then  ' Try catch para atrapar el error en caso de que el ordenador del usuario  Try  Dim program As String  program = "calc.exe"  Process.Start(program)  Catch ex As System.ComponentModel.Win32Exception '  MsgBox("Ha ocurrido un error, no se pudo iniciar la calculadora.", MsgBoxStyle.OkOnly, "Error (proceso calculadora)")  End Try  controlCalculadora = controlCalculadora + 1  Else  MsgBox("Solo se pueden iniciar dos calculadoras por sesión.", MsgBoxStyle.OkOnly, "Aviso")  End If  End Sub  ' Método que se ejecuta cuando el botón "Salir..." del ToolStrip es pulsado y que nos lleva al formulario "MenuPrincipal"  Private Sub SalirToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles SalirToolStripMenuItem.Click  ' Mostramos el menú principal.  MenuPrincipal.Show()  ' Reiniciamos su valor para la próxima vez  controlCalculadora = 0  ' Cerramos este formulario  Me.Hide()  End Sub  ' Método que se ejecuta al pulsar el botón "Eliminar"  Private Sub Button\_Eliminar\_Click(sender As Object, e As EventArgs) Handles Button\_Eliminar.Click  Try  If TextBox\_NumeroSocioOCULTO.Text = "" Then  MsgBox("Debes seleccionar un registro para eliminarlo", MsgBoxStyle.OkOnly, "Aviso")  Else  ' #################### 1º Informaros a la DB de que vamos a eliminar un registro ##############################  Dim cb As New OleDbCommandBuilder(adaptador)  adaptador.DeleteCommand = cb.GetDeleteCommand  ' #################### 2º Cambiamos el estado de los botones del menuStrip ##############################  Dim res As Integer  res = MsgBox("¿Estás seguro de que quieres eliminar?", MsgBoxStyle.YesNo, "Importante")  If res = vbYes Then  If BindingContext(midataset, "Socios").Count > 0 Then  BindingContext(midataset, "Socios").RemoveAt(BindingContext(midataset, "Socios").Position)  End If  If BindingContext(midataset, "Socios").Count = 0 Then  Button\_Eliminar.Enabled = False  End If  End If  ' #################### 3º Cambiamos el estado de los botones del menuStrip ##############################  adaptador.Update(midataset.Tables("Socios"))  midataset.Clear()  adaptador.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  ' registro()  ' #################### 4º Cambiamos el estado de los botones del menuStrip ##############################  Button\_Eliminar.Enabled = False  End If  Catch ex As Exception  MsgBox(ex.Message)  End Try  End Sub  ' Método que se ejecuta cuando el botón limpiar es presionado.  ' Limpia todos los textBoxes del formulario  Private Sub Button\_Limpiar\_Click(sender As Object, e As EventArgs) Handles Button\_Limpiar.Click  TextBox\_NumeroSocio.Clear()  TextBox\_Nombre.Clear()  TextBox\_Apellidos.Clear()  TextBox\_Telefono.Clear()  ' Actualizamos el datagriview  midataset.Clear()  ' Cargar la memoria del cache con datos.  adaptador.Fill(midataset, "Socios")  ' cargar en el datagridview, le decimos de donde sacamos los datos  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End Sub  ' Método que se ejecuta cuando el botón "Buscar" es pulsado  ' Buscará en la DB utilizando los datos introducidos por el usuario en los TextBoxes  Private Sub Button\_Buscar\_Click(sender As Object, e As EventArgs) Handles Button\_Buscar.Click  Try  ' Comprobamos que haya datos en los textBoxes (por lo menos en uno de ellos)  If TextBox\_NumeroSocio.Text = "" And TextBox\_Nombre.Text = "" And TextBox\_Apellidos.Text = "" And TextBox\_Telefono.Text = "" Then  MsgBox("No se puede buscar , debe rellenar al menos una caja con datos.", MsgBoxStyle.OkOnly, "Error al buscar.")  Else  ' Si se ha introducido todo  If TextBox\_NumeroSocio.Text <> "" And TextBox\_Nombre.Text <> "" And TextBox\_Apellidos.Text <> "" And TextBox\_Telefono.Text <> "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 2)  Dim resultado2 As Boolean = validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 2)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  If resultado1 = False Then  Exit Sub  ElseIf resultado2 = False Then  Exit Sub  ElseIf resultado3 = False Then  Exit Sub  ElseIf resultado4 = False Then  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE NumeroDeSocio LIKE? and Nombre LIKE? and Apellidos LIKE? and Telefono LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_NumeroSocio.Text)  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_Nombre.Text  comando.Parameters.Add("@var3", OleDbType.VarChar, 50).Value = TextBox\_Apellidos.Text  comando.Parameters.Add("@var4", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_Telefono.Text)  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Numero Nombre Apellido  If TextBox\_NumeroSocio.Text <> "" And TextBox\_Nombre.Text <> "" And TextBox\_Apellidos.Text <> "" And TextBox\_Telefono.Text = "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 2)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  If resultado1 = False Then  'MsgBox(" El dato numero socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 8.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado3 = False Then  'MsgBox(" El dato nombre, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado4 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE NumeroDeSocio LIKE? and Nombre LIKE? and Apellidos LIKE? ", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_NumeroSocio.Text)  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_Nombre.Text  comando.Parameters.Add("@var3", OleDbType.VarChar, 50).Value = TextBox\_Apellidos.Text  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Nombre Apellidos Telefono  If TextBox\_NumeroSocio.Text = "" And TextBox\_Nombre.Text <> "" And TextBox\_Apellidos.Text <> "" And TextBox\_Telefono.Text <> "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado2 As Boolean = validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 2)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  If resultado2 = False Then  'MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 9.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado3 = False Then  'MsgBox(" El dato nombre, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado4 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE Nombre LIKE? and Apellidos LIKE? and Telefono LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 50).Value = TextBox\_Nombre.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_Apellidos.Text  comando.Parameters.Add("@var3", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_Telefono.Text)  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Numero Apellidos Telefono  If TextBox\_NumeroSocio.Text <> "" And TextBox\_Nombre.Text = "" And TextBox\_Apellidos.Text <> "" And TextBox\_Telefono.Text <> "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 2)  Dim resultado2 As Boolean = validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 2)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  If resultado1 = False Then  'MsgBox(" El dato numero socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 8.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado2 = False Then  'MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 9.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado4 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE NumeroDeSocio LIKE? and Apellidos LIKE? and Telefono LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_NumeroSocio.Text)  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_Apellidos.Text  comando.Parameters.Add("@var3", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_Telefono.Text)  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Numero Nombre Telefono  If TextBox\_NumeroSocio.Text <> "" And TextBox\_Nombre.Text <> "" And TextBox\_Apellidos.Text = "" And TextBox\_Telefono.Text <> "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 2)  Dim resultado2 As Boolean = validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 2)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  If resultado1 = False Then  'MsgBox(" El dato numero socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 8.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado2 = False Then  'MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 9.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado3 = False Then  'MsgBox(" El dato nombre, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE NumeroDeSocio LIKE? and Nombre LIKE? and Telefono LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_NumeroSocio.Text)  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_Nombre.Text  comando.Parameters.Add("@var3", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_Telefono.Text)  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Numero Nombre  If TextBox\_NumeroSocio.Text <> "" And TextBox\_Nombre.Text <> "" And TextBox\_Apellidos.Text = "" And TextBox\_Telefono.Text = "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 2)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  If resultado1 = False Then  'MsgBox(" El dato numero socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 8.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado3 = False Then  'MsgBox(" El dato nombre, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE NumeroDeSocio LIKE? and Nombre LIKE? ", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_NumeroSocio.Text)  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_Nombre.Text  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Numero Apellidos  If TextBox\_NumeroSocio.Text <> "" And TextBox\_Nombre.Text = "" And TextBox\_Apellidos.Text <> "" And TextBox\_Telefono.Text = "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 2)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  If resultado1 = False Then  'MsgBox(" El dato numero socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 8.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado4 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE NumeroDeSocio LIKE? and Apellidos LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_NumeroSocio.Text)  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_Apellidos.Text  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Nombre Apellidos  If TextBox\_NumeroSocio.Text = "" And TextBox\_Nombre.Text <> "" And TextBox\_Apellidos.Text <> "" And TextBox\_Telefono.Text = "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  If resultado3 = False Then  'MsgBox(" El dato nombre, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado4 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE Nombre LIKE? and Apellidos LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 50).Value = TextBox\_Nombre.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_Apellidos.Text  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Apellidos Telefono  If TextBox\_NumeroSocio.Text = "" And TextBox\_Nombre.Text = "" And TextBox\_Apellidos.Text <> "" And TextBox\_Telefono.Text <> "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado2 As Boolean = validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 2)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  If resultado2 = False Then  'MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 9.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado4 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE Apellidos LIKE? and Telefono LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 50).Value = TextBox\_Apellidos.Text  comando.Parameters.Add("@var2", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_Telefono.Text)  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Numero telefono  If TextBox\_NumeroSocio.Text <> "" And TextBox\_Nombre.Text = "" And TextBox\_Apellidos.Text = "" And TextBox\_Telefono.Text <> "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 2)  Dim resultado2 As Boolean = validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 2)  If resultado1 = False Then  'MsgBox(" El dato numero socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 8.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado2 = False Then  'MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 9.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE NumeroDeSocio LIKE? and Telefono LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_NumeroSocio.Text)  comando.Parameters.Add("@var2", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_Telefono.Text)  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Nombre Telefono  If TextBox\_NumeroSocio.Text = "" And TextBox\_Nombre.Text <> "" And TextBox\_Apellidos.Text = "" And TextBox\_Telefono.Text <> "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado2 As Boolean = validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 2)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  If resultado2 = False Then  'MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 9.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado3 = False Then  'MsgBox(" El dato nombre, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE Nombre LIKE? and Telefono LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 50).Value = TextBox\_Nombre.Text  comando.Parameters.Add("@var2", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_Telefono.Text)  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Numero  If TextBox\_NumeroSocio.Text <> "" And TextBox\_Nombre.Text = "" And TextBox\_Apellidos.Text = "" And TextBox\_Telefono.Text = "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 2)  If resultado1 = False Then  'MsgBox(" El dato numero socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 8.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE NumeroDeSocio LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_NumeroSocio.Text)  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Nombre  If TextBox\_NumeroSocio.Text = "" And TextBox\_Nombre.Text <> "" And TextBox\_Apellidos.Text = "" And TextBox\_Telefono.Text = "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  If resultado3 = False Then  'MsgBox(" El dato nombre, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE Nombre LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 50).Value = TextBox\_Nombre.Text  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Apellido  If TextBox\_NumeroSocio.Text = "" And TextBox\_Nombre.Text = "" And TextBox\_Apellidos.Text <> "" And TextBox\_Telefono.Text = "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  If resultado4 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE Apellidos LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 50).Value = TextBox\_Apellidos.Text  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  ' Telefono  If TextBox\_NumeroSocio.Text = "" And TextBox\_Nombre.Text = "" And TextBox\_Apellidos.Text = "" And TextBox\_Telefono.Text <> "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado2 As Boolean = validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 2)  If resultado2 = False Then  'MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 9.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Socios WHERE Telefono LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.Integer, 15).Value = Convert.ToInt64(TextBox\_Telefono.Text)  midataset.Clear()  cb.Fill(midataset, "Socios")  DataGridView\_Socios.DataSource = midataset  DataGridView\_Socios.DataMember = "Socios"  End If  End If  Catch ex As System.InvalidOperationException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex2 As System.FormatException  ' Avisamos del error por mensaje  MsgBox("El formato de los datos introducidos es incorrecto, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex3 As System.Data.OleDb.OleDbException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, es la sintaxis correcta?, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex4 As System.NullReferenceException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, intentalo de nuevo. Referencia a objeto no establecida como instancia de un objeto.", MsgBoxStyle.OkOnly, "Operación invalida")  End Try  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 8 caracteres.  Private Sub TextBox\_NumeroSocio\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_NumeroSocio.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 9 caracteres.  Private Sub TextBox\_Telefono\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_Telefono.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 50 caracteres.  Private Sub TextBox\_Nombre\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_Nombre.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  End Sub  ' Validamos los apellidos  Private Sub TextBox\_Apellidos\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_Apellidos.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  End Sub  ' Esté método es ejecutado cuando el usuario presiona la tecla "F1"  Private Sub GestionSocios\_HelpRequested(sender As Object, hlpevent As HelpEventArgs) Handles Me.HelpRequested  Help.ShowHelp(Me, "CHM\LaCasaDelLibro.chm", "")  End Sub  ' Esté método es ejecutado cuando el usuario presiona el botón de ayuda situado en la barra superior.  ' Muestra un documento de ayuda.  Private Sub VerLaAyudaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles VerLaAyudaToolStripMenuItem.Click  Help.ShowHelp(Me, "CHM\LaCasaDelLibro.chm", "")  End Sub  Private Sub GenerarInformeToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GenerarInformeToolStripMenuItem.Click  GestionSociosInforme.ShowDialog()  End Sub  End Class |

### 8.10 Formulario GestionSociosAltas

#### Explicación de formulario

Este formulario permite al usuario añadir un nuevo socio a la base de datos.

Para ello, el usuario debe rellenar todas las cajas con datos y después presionar el botón “Alta”.

Tras completarse la operación, el formulario se cierra de forma automática.



|  |
| --- |
| ' Necesitamos importar el módelo de base de datos que vamos a utilizar, este es de access.  Imports System.Data.OleDb  ' No hace falta hacer imports libValidaciones para instanciar sus clases porque está incluida en el proyecto.  Imports System.IO  Public Class GestionSociosAltas  ' Especificamos la base de datos a la que nos vamos a conectar.  Public conexion As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=cas\_lib\_dib.accdb")  ' Al adaptador le asignamos la conexion que acabamos de realizar y una consulta  Public adaptador As New OleDbDataAdapter("Select \* from Socios", conexion)  ' Aquí alojaremos los datos de la DB  Public midataset As New DataSet  ' Para evitar la apariciónd de una excepción cuando no le toca  Dim validacionControlCorreo As Integer = 0  ' Método que se ejecuta cuando el botón "Salir..." del ToolStrip es pulsado y que nos lleva al formulario "GestionSocios".  Private Sub SalirToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles SalirToolStripMenuItem.Click  ' Mostramos el formulario "GestionSocios".  GestionSocios.Show()  ' Reiniciamos el valor para la próxima vez  controlCalculadora = 0  ' Cerramos este formulario  Me.Close()  End Sub  Dim controlCalculadora As Integer  ' Método que se ejecuta cuando es pulsado el botón "Calculadora" del menuStrip  Private Sub CalculadoraToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles CalculadoraToolStripMenuItem.Click  If controlCalculadora < 2 Then  ' Try catch para atrapar el error en caso de que el ordenador del usuario  Try  Dim program As String  program = "calc.exe"  Process.Start(program)  Catch ex As System.ComponentModel.Win32Exception '  MsgBox("Ha ocurrido un error, no se pudo iniciar la calculadora.", MsgBoxStyle.OkOnly, "Error (proceso calculadora)")  End Try  controlCalculadora = controlCalculadora + 1  Else  MsgBox("Solo se pueden iniciar dos calculadoras por sesión.", MsgBoxStyle.OkOnly, "Aviso")  End If  End Sub  ' Método que se ejecuta cuando se pulsa el botón "Limpiar".  ' "Limpia" todos los textBox, dejandolos vacios.  Private Sub Button\_Limpiar\_Click(sender As Object, e As EventArgs) Handles Button\_Limpiar.Click  ' Vaciamos cada textBox de forma individual  TextBox\_NumeroSocio.Clear()  TextBox\_Nombre.Clear()  TextBox\_Apellidos.Clear()  TextBox\_Telefono.Clear()  TextBox\_Correo.Clear()  End Sub  ' Método que se ejecuta cuando el botón "Alta" es pulsado.  ' Introduce los datos escritos por el usuario en los textBox en la DB.  Private Sub Button\_Alta\_Click(sender As Object, e As EventArgs) Handles Button\_Alta.Click  If TextBox\_NumeroSocio.Text = "" Or TextBox\_Nombre.Text = "" Or TextBox\_Apellidos.Text = "" Or TextBox\_Telefono.Text = "" Or TextBox\_Correo.Text = "" Then  MsgBox("No se puede dar de alta , debe rellenar todos los datos.", MsgBoxStyle.OkOnly, "Error al dar de alta.")  Else  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 2)  Dim resultado2 As Boolean = validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 2)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  Dim resultado5 As Boolean = validarNumeroSocio.validarCorreo(TextBox\_Correo.Text, 1)  If resultado1 = False Then  'MsgBox(" El dato numero socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 8.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado2 = False Then  'MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 9.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado3 = False Then  'MsgBox(" El dato nombre, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado4 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado5 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim valor As String  Dim control As Integer = 0  ' Comprobamos que la clave primaria no se encuentra ya registrada.  For contador As Integer = 0 To GestionSocios.DataGridView\_Socios.RowCount - 1  valor = GestionSocios.DataGridView\_Socios.Item(0, contador).Value  If valor = TextBox\_NumeroSocio.Text Then  MsgBox("No puedes introducir un número de socio que ya existe en la base de datos.", MsgBoxStyle.OkOnly, "Error, clave duplicada")  control = 1  End If  Next  If control = 0 Then  Try  ' #################### 1º Preparamos a la base de datos para recibir los datos. ##############################  Dim cb As New OleDbCommandBuilder(adaptador)  adaptador.InsertCommand = cb.GetInsertCommand  ' #################### 2º Recogemos los datos y los introducimos ##############################  Dim drc As DataRowCollection = midataset.Tables("Socios").Rows  drc.Add(TextBox\_NumeroSocio.Text, TextBox\_Nombre.Text, TextBox\_Apellidos.Text, TextBox\_Telefono.Text, TextBox\_Correo.Text)  adaptador.Update(midataset.Tables("Socios"))  Catch ex As System.InvalidOperationException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex2 As System.FormatException  ' Avisamos del error por mensaje  MsgBox("El formato de los datos introducidos es incorrecto, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex3 As System.Data.OleDb.OleDbException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, es la sintaxis correcta?, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex4 As System.NullReferenceException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, intentalo de nuevo. Referencia a objeto no establecida como instancia de un objeto.", MsgBoxStyle.OkOnly, "Operación invalida")  End Try  ' #################### 3º Actualizamos el middataset ##############################  ' Actualizamos el dataGridView del formulario de gestión principal  GestionSocios.midataset.Clear()  GestionSocios.adaptador.Fill(GestionSocios.midataset, "Socios")  ' Reiniciamos su valor para la próxima vez  controlCalculadora = 0  ' Cerramos la ventana  Me.Close()  ' System.NullReferenceException: 'Referencia a objeto no establecida como instancia de un objeto.'  ' System.Data.OleDb.OleDbException: 'Error de sintaxis en la instrucción INSERT INTO.'  End If  End If  End Sub  ' Método que se ejecuta al iniciarse el formulario.  Private Sub GestionSociosAltas\_Load(sender As Object, e As EventArgs) Handles MyBase.Load  ' Cargar la memoria del cache con datos.  adaptador.Fill(midataset, "Socios")  If GestionSocios.numeroDeControlBindingAltaSocios = 0 Then  Me.TextBox\_NumeroSocio.DataBindings.Add("text", midataset, "Socios.NumeroDeSocio")  Me.TextBox\_Nombre.DataBindings.Add("text", midataset, "Socios.Nombre")  Me.TextBox\_Apellidos.DataBindings.Add("text", midataset, "Socios.Apellidos")  Me.TextBox\_Telefono.DataBindings.Add("text", midataset, "Socios.Telefono")  Me.TextBox\_Correo.DataBindings.Add("text", midataset, "Socios.Correo")  GestionSocios.numeroDeControlBindingAltaSocios = 1  End If  ' Relacionar los campos de la tabla con los textbox  controlCalculadora = 0  ' Vaciamos cada textBox de forma individual  TextBox\_NumeroSocio.Clear()  TextBox\_Nombre.Clear()  TextBox\_Apellidos.Clear()  TextBox\_Telefono.Clear()  TextBox\_Correo.Clear()  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 8 caracteres.  Private Sub TextBox\_NumeroSocio\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_NumeroSocio.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 9 caracteres.  Private Sub TextBox\_Telefono\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_Telefono.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de no más de 50 caracteres.  Private Sub TextBox\_Nombre\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_Nombre.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de no más de 50 caracteres.  Private Sub TextBox\_Apellidos\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_Apellidos.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  End Sub  ' Validamos este campo evitando que tenga caracteres prohibidos en un e-mail y que tenga una longitud de no más de 120 caracteres.  Private Sub TextBox\_Correo\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_Correo.TextChanged  If validacionControlCorreo = 1 Then  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarCorreo(TextBox\_Correo.Text, 1)  Else  validacionControlCorreo = 1  End If  End Sub  ' Esté método es ejecutado cuando el usuario presiona la tecla "F1"  Private Sub GestionSocios\_HelpRequested(sender As Object, hlpevent As HelpEventArgs) Handles Me.HelpRequested  Help.ShowHelp(Me, "CHM\LaCasaDelLibro.chm", "")  End Sub  ' Esté método es ejecutado cuando el usuario presiona el botón de ayuda situado en la barra superior.  ' Muestra un documento de ayuda.  Private Sub VerLaAyudaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles VerLaAyudaToolStripMenuItem.Click  Help.ShowHelp(Me, "CHM\LaCasaDelLibro.chm", "")  End Sub  End Class |

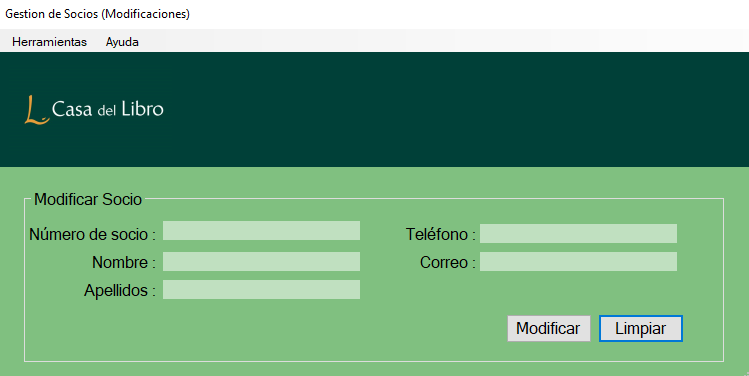
### 8.11 Formulario GestionSociosModificaciones

#### Explicación de formulario

Este formulario permite al usuario realizar la modificación del registro de un socio en la base de datos.

Para ello, debe modificar aquellos datos que desee y después presionar el botón “Modificar”.

El formulario se cerrara de forma automática tras completarse la operación.

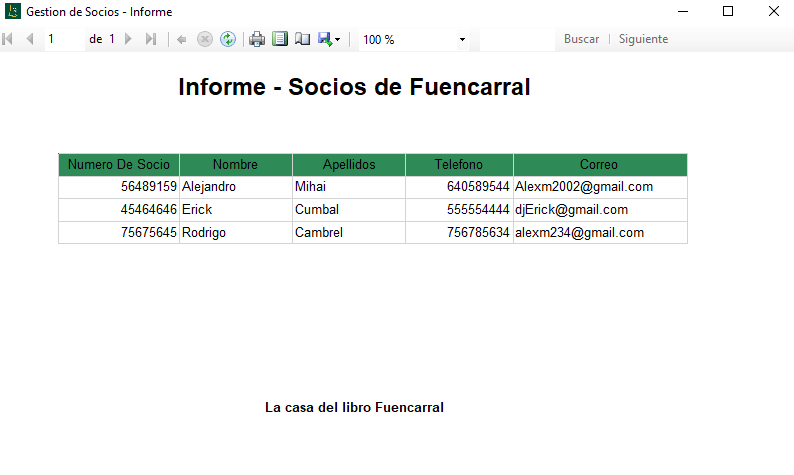


|  |
| --- |
| ' Necesitamos importar el módelo de base de datos que vamos a utilizar, este es de access.  Imports System.Data.OleDb  ' No hace falta hacer imports libValidaciones para instanciar sus clases porque está incluida en el proyecto.  Imports System.IO  Public Class GestionSociosModificaciones  ' Variable para almacenar el número de socio inicial con el que se identificará el registro a modificar.  Dim numSocioInicial As String  ' Variable de control que controla que no se recoja más de una vez el valor de la primary key  Dim numControlPK As Single = 0  ' Para evitar la apariciónd de una excepción cuando no le toca  Dim validacionControlCorreo As Integer = 0  Dim validacionNumeroControl As Integer = 0  ' Especificamos la base de datos a la que nos vamos a conectar.  Public conexion As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=cas\_lib\_dib.accdb")  ' Al adaptador le asignamos la conexion que acabamos de realizar y una consulta  Public adaptador As New OleDbDataAdapter("Select \* from Socios", conexion)  ' Aquí alojaremos los datos de la DB  Public midataset As New DataSet  ' Método que se ejecuta cuando se pulsa el botón "Limpiar".  ' "Limpia" todos los textBox, dejandolos vacios.  Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click  ' Vaciamos cada textBox de forma individual  TextBox\_NumeroSocio.Clear()  TextBox\_Nombre.Clear()  TextBox\_Apellidos.Clear()  TextBox\_Telefono.Clear()  TextBox\_Correo.Clear()  End Sub  ' Método que se ejecuta al iniciarse el formulario. El cual nos mostrará los datos de la Seleccion que hemos hecho en el DataGridView\_Socios del formulario GestionSocios  ' en los textBox correspondientes. Para esto crearemos un nuevo comando, y lo asociaremos al midataset para obtener los datos que queremos  Private Sub GestionSociosModificaciones\_Load(sender As Object, e As EventArgs) Handles MyBase.Load  'Creamos un comando nuevo con una Query nueva la cual vamos a usar para poder buscar un socio en concreto  Dim comando As New OleDbCommand(("select \* from Socios where NumeroDeSocio = @numSocio"), conexion)  'Al adaptador le asignamos este nuevo comando "comando"  adaptador.SelectCommand = comando  'Al comando le vamos a pasar por parametro @numSocio (valor de tipo Numeric, 8) el cual va a ser cogido de la Fila(currentRow) seleccionada de la Columna 0 (NumeroSocio)  'del DataGridView\_Socios del formulario GestionSocios. Es decir al seleccionar un elemento detectará cual es la fila y el valor de la columna 0 y lo pasará por parametro.  comando.Parameters.Add("@numSocio", OleDbType.Numeric, 8).Value = GestionSocios.DataGridView\_Socios.Item(0, GestionSocios.DataGridView\_Socios.CurrentRow.Index).Value  'Limpiamos el midataset para que no haya información residual  midataset.Clear() 'de este form  'Asociamos el nuevo adaptador con el nuevo comando al midataset de la tabla Socios  adaptador.Fill(midataset, "Socios")  If GestionSocios.numeroDeControlBindingModificaciones = 0 Then  'Se relacionan los campos de la tabla con los textbox y se muestran los datos del registro que queremos modificar.  Me.TextBox\_NumeroSocio.DataBindings.Add("text", midataset, "Socios.NumeroDeSocio")  Me.TextBox\_Nombre.DataBindings.Add("text", midataset, "Socios.Nombre")  Me.TextBox\_Apellidos.DataBindings.Add("text", midataset, "Socios.Apellidos")  Me.TextBox\_Telefono.DataBindings.Add("text", midataset, "Socios.Telefono")  Me.TextBox\_Correo.DataBindings.Add("text", midataset, "Socios.Correo")  GestionSocios.numeroDeControlBindingModificaciones = 1  End If  ' Inicializamos la variable de control  controlCalculadora = 0  ' Inicializamos la variable asignandole el número de socio inicial  numSocioInicial = GestionSocios.DataGridView\_Socios.Item(0, GestionSocios.DataGridView\_Socios.CurrentRow.Index).Value  End Sub  ' Método que se ejecuta al pulsarse el botón "Modificar"  ' Se encarga de modificar los datos ya existentes en la DB  Private Sub Button\_Modificar\_Click(sender As Object, e As EventArgs) Handles Button\_Modificar.Click  If TextBox\_NumeroSocio.Text = "" Or TextBox\_Nombre.Text = "" Or TextBox\_Apellidos.Text = "" Or TextBox\_Telefono.Text = "" Or TextBox\_Correo.Text = "" Then  MsgBox("Debes seleccionar un registro para actualizarlo y si lo has seleccionado, no debe quedar ningún campo en blanco", MsgBoxStyle.OkOnly, "Error al dar de alta.")  Else  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 2)  Dim resultado2 As Boolean = validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 2)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  Dim resultado5 As Boolean = validarNumeroSocio.validarCorreo(TextBox\_Correo.Text, 1)  If resultado1 = False Then  'MsgBox(" El dato numero socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 8.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado2 = False Then  'MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 9.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado3 = False Then  'MsgBox(" El dato nombre, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado4 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado5 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  ' Variable auxiliar  Dim valor As String  ' Variable de control para controlar las primary keys duplicadas  Dim control As Integer = 0  ' Comprobamos que la clave primaria no se encuentra ya registrada.  For contador As Integer = 0 To GestionSocios.DataGridView\_Socios.RowCount - 1  valor = GestionSocios.DataGridView\_Socios.Item(0, contador).Value  If valor = TextBox\_NumeroSocio.Text And valor <> numSocioInicial Then  MsgBox("No puedes introducir un número de socio que ya existe en la base de datos.", MsgBoxStyle.OkOnly, "Error, clave duplicada")  ' Dado que la primary key que pretendemos introducir ya se encuentra en los registros, cambiamos la variable de control  control = 1  End If  Next  If control = 0 Then  Try  ' Montamos una query parametrizada.  Dim queryParametrizada As String = "UPDATE Socios SET NumeroDeSocio=?, Nombre=?, Apellidos=?, Telefono=?, Correo=? WHERE NumeroDeSocio=?"  Using cmd = New OleDbCommand(queryParametrizada, conexion)  ' Abrimos la conexión  conexion.Open()  cmd.Parameters.AddWithValue("@p1", Convert.ToInt64(TextBox\_NumeroSocio.Text))  cmd.Parameters.AddWithValue("@p2", TextBox\_Nombre.Text)  cmd.Parameters.AddWithValue("@p3", TextBox\_Apellidos.Text)  cmd.Parameters.AddWithValue("@p4", Convert.ToInt64(TextBox\_Telefono.Text))  cmd.Parameters.AddWithValue("@p5", TextBox\_Correo.Text)  cmd.Parameters.AddWithValue("@p6", numSocioInicial)  cmd.ExecuteNonQuery()  ' Cerramos la conexión  conexion.Close()  End Using  Catch ex As System.InvalidOperationException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex2 As System.Data.OleDb.OleDbException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex3 As System.FormatException  ' Avisamos del error por mensaje  MsgBox("Uno de los datos tiene un formato incorrecto, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  End Try  ' Actualizamos el dataGridView del formulario de gestión principal  GestionSocios.midataset.Clear()  GestionSocios.adaptador.Fill(GestionSocios.midataset, "Socios")  ' Reiniciamos su valor para la próxima vez  controlCalculadora = 0  ' Cerramos la ventana  Me.Close()  End If  End If  End Sub  ' ##################################################### Métodos varios ##########################################################################  ' Método que se ejecuta cuando el botón "Salir..." del ToolStrip es pulsado y que nos lleva al formulario "GestionSocios".  Private Sub SalirToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles SalirToolStripMenuItem.Click  ' Mostramos el formulario "GestionSocios".  GestionSocios.Show()  ' Reiniciamos el valor para la próxima vez  controlCalculadora = 0  ' Cerramos este formulario  Me.Close()  End Sub  Dim controlCalculadora As Integer  ' Método que se ejecuta cuando es pulsado el botón "Calculadora" del menuStrip  Private Sub CalculadoraToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles CalculadoraToolStripMenuItem.Click  If controlCalculadora < 2 Then  ' Try catch para atrapar el error en caso de que el ordenador del usuario  Try  Dim program As String  program = "calc.exe"  Process.Start(program)  Catch ex As System.ComponentModel.Win32Exception '  MsgBox("Ha ocurrido un error, no se pudo iniciar la calculadora.", MsgBoxStyle.OkOnly, "Error (proceso calculadora)")  End Try  controlCalculadora = controlCalculadora + 1  Else  MsgBox("Solo se pueden iniciar dos calculadoras por sesión.", MsgBoxStyle.OkOnly, "Aviso")  End If  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de 8 caracteres.  Private Sub TextBox\_NumeroSocio\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_NumeroSocio.TextChanged  If validacionNumeroControl = 1 Then  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNumeroSocio(TextBox\_NumeroSocio.Text, 1)  Else  validacionNumeroControl = 1  End If  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 9 caracteres.  Private Sub TextBox\_Telefono\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_Telefono.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarTelefono(TextBox\_Telefono.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de no más de 50 caracteres.  Private Sub TextBox\_Nombre\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_Nombre.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNombre(TextBox\_Nombre.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de no más de 50 caracteres.  Private Sub TextBox\_Apellidos\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_Apellidos.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNombre(TextBox\_Apellidos.Text, 2)  End Sub  ' Validamos este campo evitando que tenga caracteres prohibidos en un e-mail y que tenga una longitud de no más de 120 caracteres.  Private Sub TextBox\_Correo\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_Correo.TextChanged  If validacionControlCorreo = 1 Then  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarCorreo(TextBox\_Correo.Text, 1)  Else  validacionControlCorreo = 1  End If  End Sub  ' Esté método es ejecutado cuando el usuario presiona la tecla "F1"  Private Sub GestionSocios\_HelpRequested(sender As Object, hlpevent As HelpEventArgs) Handles Me.HelpRequested  Help.ShowHelp(Me, "CHM\LaCasaDelLibro.chm", "")  End Sub  ' Esté método es ejecutado cuando el usuario presiona el botón de ayuda situado en la barra superior.  ' Muestra un documento de ayuda.  Private Sub VerLaAyudaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles VerLaAyudaToolStripMenuItem.Click  Help.ShowHelp(Me, "CHM\LaCasaDelLibro.chm", "")  End Sub  End Class |

### 8.12 Formulario GestionSociosInforme

#### Explicación de formulario

Este formulario genera un informe que el usuario puede visionar e imprimir en el formato que desee.

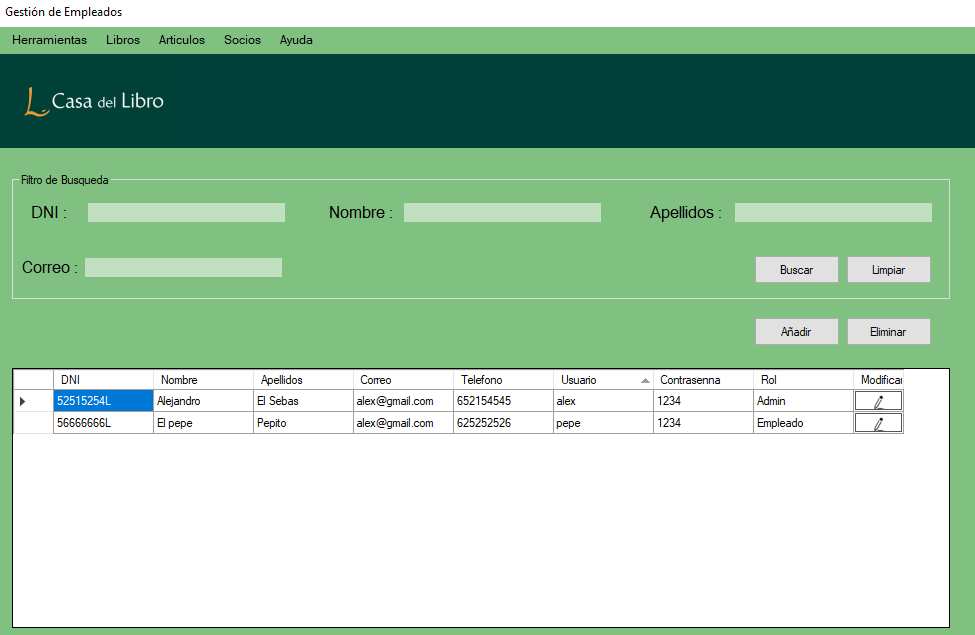


|  |
| --- |
| ' Necesitamos importar el módelo de base de datos que vamos a utilizar, este es de access.  Imports System.Data.OleDb  Public Class GestionSociosInforme  Private Sub GestionSociosInforme\_Load(sender As Object, e As EventArgs) Handles MyBase.Load  Dim dt As New DataTable  Using con As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=cas\_lib\_dib.accdb")  con.Open()  Using cmd As New OleDbCommand("Select \* from Socios", con)  Dim da As New OleDbDataAdapter  da.SelectCommand = cmd  da.Fill(dt)  End Using  End Using  With Me.ReportViewer1.LocalReport  .DataSources.Clear()  .ReportPath = "GestionSociosInforme.rdlc"  .DataSources.Add(New Microsoft.Reporting.WinForms.ReportDataSource("DataSetSocios", dt))  End With  Me.ReportViewer1.RefreshReport()  End Sub  End Class |

### 8.13 Formulario GestionEmpleados

#### Explicación de formulario

Este formulario permite al usuario buscar empleados en la base de datos en base a cuatro datos (DNI, Nombre, Apellidos, Correo) y además, permite su visualización, añadir nuevos empleados, eliminarlos y modificarlos.



|  |
| --- |
| ' Necesitamos importar el módelo de base de datos que vamos a utilizar, este es de access.  Imports System.Data.OleDb  ' No hace falta hacer imports libValidaciones para instanciar sus clases porque está incluida en el proyecto.  Imports System.IO  Public Class GestionEmpleados  ' Número de control para controlar el dataBinding de los text boxes del formulario modificaciones, evitando que se relacionen dos veces.  Public numeroDeControlBindingModificacionesEmpleados As Long  Public numeroDeControlBindingAltaEmpleados As Long  Public posicionDataGridSeleccionada As Integer  ' Especificamos la base de datos a la que nos vamos a conectar.  Public conexion As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=cas\_lib\_dib.accdb")  ' Al adaptador le asignamos la conexion que acabamos de realizar y una consulta  Public adaptador As New OleDbDataAdapter("Select \* from Empleados", conexion)  'Declaramos el botón Modificar que será creado dinamicamente junto a los resgistros del DataGridView.  Public button\_din\_modificar As New DataGridViewButtonColumn  'Creamos in estilo que aplicaremos a las caracteristicas del botón Modificar.  Public style As New DataGridViewCellStyle  ' Aquí alojaremos los datos de la DB  Public midataset As New DataSet  Public posicionDataGridSeleccionadaEmpleados As Integer  ' Método que se ejecuta cuando el formulario es iniciado por primera vez  Private Sub GestionEmpleados\_Load(sender As Object, e As EventArgs) Handles MyBase.Load  Try  ' Cargar la memoria del cache con datos.  adaptador.Fill(midataset, "Empleados")  Me.TextBox\_DNIOCULTO.DataBindings.Add("text", midataset, "Empleados.DNI")  ' cargar en el datagridview, le decimos de donde sacamos los datos  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  ' Inicializamos el número de control  numeroDeControlBindingModificacionesEmpleados = 0  numeroDeControlBindingAltaEmpleados = 0  ' Inicializamos la variable de control  controlCalculadora = 0  'Creación en la ultima columna del DataGridView el botón de modificar en cada registro.  crearButtonDataGridViewEmpleados()  Catch ex As System.Data.OleDb.OleDbException  MsgBox("Parece que algo ha salido mal. Revise que la base de datos no esté abierta durante la ejecución.", MsgBoxStyle.OkOnly, "Error - Base de datos")  ' Mostramos el menú principal.  MenuPrincipal.Show()  ' Cerramos este formulario  Me.Close()  End Try  End Sub  ' Método que permite posicionar la ventana en la posición especificada del formulario "GestionEmpleadosAltas".  ' En este caso para evitar que quede encima del formulario anterior.  Private Shared Sub posicionarGestionAltas()  GestionEmpleadosAltas.StartPosition = FormStartPosition.Manual  Dim a As Integer  a = My.Computer.Screen.Bounds.Size.Width - (My.Computer.Screen.Bounds.Size.Width \* 0.55)  Dim b As Integer  b = My.Computer.Screen.Bounds.Size.Height - (My.Computer.Screen.Bounds.Size.Height \* 0.8)  GestionEmpleadosAltas.Location = New Point(a, b)  End Sub  ' Método que permite posicionar la ventana en la posición especificada del formulario "GestionEmpleadosModificaciones".  ' En este caso para evitar que quede encima del formulario anterior.  Private Shared Sub posicionarGestionModificaciones()  GestionEmpleadosModificaciones.StartPosition = FormStartPosition.Manual  Dim a As Integer  a = My.Computer.Screen.Bounds.Size.Width - (My.Computer.Screen.Bounds.Size.Width \* 0.55)  Dim b As Integer  b = My.Computer.Screen.Bounds.Size.Height - (My.Computer.Screen.Bounds.Size.Height \* 0.8)  GestionEmpleadosModificaciones.Location = New Point(a, b)  End Sub  ' Método que se ejecuta cuando el botón "Añadir" es pulsado.  Private Sub Button\_Annadir\_Click(sender As Object, e As EventArgs) Handles Button\_Annadir.Click  ' Posicionamos el formulario que vamos a mostrar.  posicionarGestionAltas()  ' Mostramos el formulario  GestionEmpleadosAltas.ShowDialog()  End Sub  'Método que crea y da formato al botón de modificar, en cada una de los registros del DataGridView.  'Este bóton abrirá el formulario de GestionEmpleadosModificaciones.  Public Sub crearButtonDataGridViewEmpleados()  'style.BackColor when the button Is Not selected  style.BackColor = Color.White  'Le asignamos un nombre  button\_din\_modificar.Name = "Modificar"  'Le asignamos un tamaño  button\_din\_modificar.Width = 50  'declare a DataGridViewCellStyle  button\_din\_modificar.FlatStyle = FlatStyle.Popup  'Cargar en la ultima columna del DataGridView, la creación y el estilo del botón interno.  DataGridView\_Empleados.Columns.Add(button\_din\_modificar)  End Sub  'Metodo que pinta el Icono asociado al botón dinámico Modificar, el la ultima Columna del DataGridView  Private Sub DataGridView1\_CellPainting(ByVal sender As Object, ByVal e As DataGridViewCellPaintingEventArgs) Handles DataGridView\_Empleados.CellPainting  If e.ColumnIndex = 8 AndAlso e.RowIndex >= 0 Then  e.Paint(e.CellBounds, DataGridViewPaintParts.All)  'Liena de código que calcula la posicion de el dibujo en concreto.  e.Graphics.DrawImage(My.Resources.modificar\_ico, CInt((e.CellBounds.Width / 2) - (My.Resources.modificar\_ico.Width / 2)) + e.CellBounds.X, CInt((e.CellBounds.Height / 2) - (My.Resources.modificar\_ico.Height / 2)) + e.CellBounds.Y)  e.Handled = True  End If  End Sub  'Metodo que detecta que botón se ha pulsado y envia la orden de posicionar a:  'GestionLibros/GestionEmpleados/GestionSocios  Private Sub posicionarFormulariosGestiones(ByVal btn\_Name As String)  'Especifica que el tipo de posicion de arranque será manual para Gestión Libros  GestionLibros.StartPosition = FormStartPosition.Manual  'Especifica que el tipo de posicion de arranque será manual para Gestión Articulos  GestionArticulos.StartPosition = FormStartPosition.Manual  'Especifica que el tipo de posicion de arranque será manual para Gestión Socios  GestionSocios.StartPosition = FormStartPosition.Manual  'Varaible tipo Integer que calculará la posición vertical en la que ubicará el nuevo formulario  Dim a As Integer  a = My.Computer.Screen.Bounds.Size.Width - (My.Computer.Screen.Bounds.Size.Width \* 0.97)  'Varaible tipo Integer que calculará la posición horizontal en la que ubicará el nuevo formulario  Dim b As Integer  b = My.Computer.Screen.Bounds.Size.Width - (My.Computer.Screen.Bounds.Size.Width \* 0.97)  'Condición que compará el nombre del botón recibido como parametro con la cadena correspondiente a dicho botón.  If btn\_Name.Equals("gestiónlibrostoolstripmenuitem1") Then  GestionLibros.Location = New Point(a, b) 'Posiciona en las coordenadas a y b  ElseIf btn\_Name.Equals("gestiónarticulostoolstripmenuitem1") Then  GestionArticulos.Location = New Point(a, b) 'Posiciona en las coordenadas a y b  ElseIf btn\_Name.Equals("gestiónsociostoolstripmenuitem1") Then  GestionSocios.Location = New Point(a, b) 'Posiciona en las coordenadas a y b  End If  End Sub  ' Método que se ejecuta cuando es pulsado el botón "Gestion de Libros" del menuStrip  Private Sub GestiónLibrosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestiónLibrosToolStripMenuItem1.Click  'Variable que nos dice cual es el Nombre del ToolStripMenuItem pulsado  Dim item\_Name As String = DirectCast(sender, ToolStripMenuItem).Name.ToLower.Replace("clear", "")  'Confirmación sobre cambio de formulario  If MsgBox("¿Está seguro de que desea cambiar de formulario?", 36, "Salir") = 6 Then  Me.Hide() 'Ocultamos este formulario  'Invocamos al método que posiciona el Formulario de Gestion  posicionarFormulariosGestiones(item\_Name)  GestionLibros.Show()  End If  End Sub  ' Método que se ejecuta cuando es pulsado el botón "Gestion de Socios" del menuStrip  Private Sub GestiónDeSociosToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles GestiónSociosToolStripMenuItem1.Click  'Variable que nos dice cual es el Nombre del ToolStripMenuItem pulsado  Dim item\_Name As String = DirectCast(sender, ToolStripMenuItem).Name.ToLower.Replace("clear", "")  'Confirmación sobre cambio de formulario  If MsgBox("¿Está seguro de que desea cambiar de formulario?", 36, "Salir") = 6 Then  Me.Hide() 'Ocultamos este formulario  'Invocamos al método que posiciona el Formulario de Gestion  posicionarFormulariosGestiones(item\_Name)  GestionSocios.Show()  End If  End Sub  ' Método que se ejecuta cuando es pulsado el botón "Gestion de Empleados" del menuStrip  Private Sub GestiónEmpleadosToolStripMenuItem1\_Click(sender As Object, e As EventArgs) Handles GestiónArticulosToolStripMenuItem1.Click  'Variable que nos dice cual es el Nombre del ToolStripMenuItem pulsado  Dim item\_Name As String = DirectCast(sender, ToolStripMenuItem).Name.ToLower.Replace("clear", "")  'Confirmación sobre cambio de formulario  If MsgBox("¿Está seguro de que desea cambiar de formulario?", 36, "Salir") = 6 Then  Me.Hide() 'Ocultamos este formulario  'Invocamos al método que posiciona el Formulario de Gestion  posicionarFormulariosGestiones(item\_Name)  GestionArticulos.Show()  End If  End Sub  Dim controlCalculadora As Integer  ' Método que se ejecuta cuando es pulsado el botón "Calculadora" del menuStrip  Private Sub CalculadoraToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles CalculadoraToolStripMenuItem.Click  If controlCalculadora < 2 Then  ' Try catch para atrapar el error en caso de que el ordenador del usuario  Try  Dim program As String  program = "calc.exe"  Process.Start(program)  Catch ex As System.ComponentModel.Win32Exception '  MsgBox("Ha ocurrido un error, no se pudo iniciar la calculadora.", MsgBoxStyle.OkOnly, "Error (proceso calculadora)")  End Try  controlCalculadora = controlCalculadora + 1  Else  MsgBox("Solo se pueden iniciar dos calculadoras por sesión.", MsgBoxStyle.OkOnly, "Aviso")  End If  End Sub  ' Método que se ejecuta al pulsar en una de las cajas del DataGridView  Private Sub DataGridView\_Socios\_CellClick(sender As Object, e As DataGridViewCellEventArgs) Handles DataGridView\_Empleados.CellClick  posicionDataGridSeleccionadaEmpleados = BindingContext(midataset, "Empleados").Position  Button\_Eliminar.Enabled = True  End Sub  'Metodo evento que capta la puslación en la celda relativa al botón.  Private Sub DataGridView1\_CellContentClick(sender As System.Object, e As DataGridViewCellEventArgs) Handles DataGridView\_Empleados.CellContentClick  Try  'Convierte el objeto en sender  Dim senderGrid = DirectCast(sender, DataGridView)  'Comprueba que es una columna del data gridview que tiene el evento y que tiene indice mayor que 0  'Si es correcto se ejecutará el comando de abrir GestionSociosModificaciones  If TypeOf senderGrid.Columns(e.ColumnIndex) Is DataGridViewButtonColumn AndAlso e.RowIndex >= 0 Then  ' Posicionamos el formulario que vamos a mostrar.  posicionarGestionModificaciones()  ' Mostramos el formulario  GestionEmpleadosModificaciones.ShowDialog()  End If  Catch ex As System.ArgumentException  ' MsgBox("Se ha producido un error al realizar la acción solicitada, intentelo de nuevo.", MsgBoxStyle.OkOnly, "Error del sistema")  End Try  End Sub  'Método que crea y da formato al botón de modificar, en cada una de los registros del DataGridView.  'Este bóton abrirá el formulario de GestionSociosModificaciones.  Public Sub crearButtonDataGridView()  'style.BackColor when the button Is Not selected  style.BackColor = Color.White  'Le asignamos un nombre  button\_din\_modificar.Name = "Modificar"  'Le asignamos un tamaño  button\_din\_modificar.Width = 50  'declare a DataGridViewCellStyle  button\_din\_modificar.FlatStyle = FlatStyle.Popup  'Cargar en la ultima columna del DataGridView, la creación y el estilo del botón interno.  DataGridView\_Empleados.Columns.Add(button\_din\_modificar)  End Sub  ' Método que se ejecuta cuando el botón "Salir..." del ToolStrip es pulsado y que nos lleva al formulario "MenuPrincipal"  Private Sub SalirToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles SalirToolStripMenuItem.Click  ' Mostramos el menú principal.  MenuPrincipal.Show()  ' Reiniciamos su valor para que la calculadora pueda mostrarse de nuevo  controlCalculadora = 0  ' Cerramos este formulario  Me.Hide()  End Sub  ' Método que se ejecuta al ser pulsado el botón "Eliminar"  Private Sub Button\_Eliminar\_Click(sender As Object, e As EventArgs) Handles Button\_Eliminar.Click  Try  If TextBox\_DNIOCULTO.Text = "" Then  MsgBox("Debes seleccionar un registro para eliminarlo")  Else  ' #################### 1º Informaros a la DB de que vamos a eliminar un registro ##############################  Dim cb As New OleDbCommandBuilder(adaptador)  adaptador.DeleteCommand = cb.GetDeleteCommand  ' #################### 2º Cambiamos el estado de los botones del menuStrip ##############################  Dim res As Integer  res = MsgBox("¿Estás seguro de que quieres eliminar?", MsgBoxStyle.YesNo)  If res = vbYes Then  If BindingContext(midataset, "Empleados").Count > 0 Then  BindingContext(midataset, "Empleados").RemoveAt(BindingContext(midataset, "Empleados").Position)  End If  If BindingContext(midataset, "Empleados").Count = 0 Then  Button\_Eliminar.Enabled = False  End If  End If  ' #################### 3º Cambiamos el estado de los botones del menuStrip ##############################  adaptador.Update(midataset.Tables("Empleados"))  midataset.Clear()  adaptador.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  ' registro()  ' #################### 4º Cambiamos el estado de los botones del menuStrip ##############################  Button\_Eliminar.Enabled = False  End If  Catch ex As Exception  MsgBox(ex.Message)  End Try  End Sub  ' Método que se ejecuta cuando el botón limpiar es presionado.  ' Limpia todos los textBoxes del formulario  Private Sub Button\_Limpiar\_Click(sender As Object, e As EventArgs) Handles Button\_Limpiar.Click  TextBox\_DNI.Clear()  TextBox\_NOMBRE.Clear()  TextBox\_APELLIDOS.Clear()  TextBox\_CORREO.Clear()  ' Actualizamos el datagridview reiniciandolo  midataset.Clear()  ' Cargar la memoria del cache con datos.  adaptador.Fill(midataset, "Empleados")  ' cargar en el datagridview, le decimos de donde sacamos los datos  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End Sub  ' Método que se ejecuta cuando el botón "Buscar" es pulsado.  ' Se encarga de buscar en base a los contenidos de los textBoxes  Private Sub Button\_Buscar\_Click(sender As Object, e As EventArgs) Handles Button\_Buscar.Click  Try  ' Comprobamos que haya datos en los textBoxes (por lo menos en uno de ellos)  If TextBox\_DNI.Text = "" And TextBox\_NOMBRE.Text = "" And TextBox\_APELLIDOS.Text = "" And TextBox\_CORREO.Text = "" Then  MsgBox("No se puede buscar , debe rellenar al menos una caja con datos.", MsgBoxStyle.OkOnly, "Error al buscar.")  Else  ' Si se ha introducido todo  If TextBox\_DNI.Text <> "" And TextBox\_NOMBRE.Text <> "" And TextBox\_APELLIDOS.Text <> "" And TextBox\_CORREO.Text <> "" Then  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 2)  Dim resultado2 As Boolean = validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  If resultado1 = False Then  'MsgBox(" El dato numero socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 8.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado2 = False Then  'MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 9.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado3 = False Then  'MsgBox(" El dato nombre, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado4 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE DNI LIKE? and Nombre LIKE? and Apellidos LIKE? and Correo LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 9).Value = TextBox\_DNI.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_NOMBRE.Text  comando.Parameters.Add("@var3", OleDbType.VarChar, 50).Value = TextBox\_APELLIDOS.Text  comando.Parameters.Add("@var4", OleDbType.VarChar, 120).Value = TextBox\_CORREO.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' DNI Nombre Apellido  If TextBox\_DNI.Text <> "" And TextBox\_NOMBRE.Text <> "" And TextBox\_APELLIDOS.Text <> "" And TextBox\_CORREO.Text = "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 2)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  If resultado1 = False Then  'MsgBox(" El dato numero socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 8.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado3 = False Then  'MsgBox(" El dato nombre, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado4 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE DNI LIKE? and Nombre LIKE? and Apellidos LIKE? ", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 9).Value = TextBox\_DNI.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_NOMBRE.Text  comando.Parameters.Add("@var3", OleDbType.VarChar, 50).Value = TextBox\_APELLIDOS.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' Nombre Apellidos Correo  If TextBox\_DNI.Text = "" And TextBox\_NOMBRE.Text <> "" And TextBox\_APELLIDOS.Text <> "" And TextBox\_CORREO.Text <> "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado2 As Boolean = validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  If resultado2 = False Then  Exit Sub  ElseIf resultado3 = False Then  Exit Sub  ElseIf resultado4 = False Then  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE Nombre LIKE? and Apellidos LIKE? and Correo LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 50).Value = TextBox\_NOMBRE.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_APELLIDOS.Text  comando.Parameters.Add("@var3", OleDbType.VarChar, 120).Value = TextBox\_CORREO.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' DNI Apellidos Correo  If TextBox\_DNI.Text <> "" And TextBox\_NOMBRE.Text = "" And TextBox\_APELLIDOS.Text <> "" And TextBox\_CORREO.Text <> "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 2)  Dim resultado2 As Boolean = validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  If resultado1 = False Then  Exit Sub  ElseIf resultado2 = False Then  Exit Sub  ElseIf resultado4 = False Then  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE DNI LIKE? and Apellidos LIKE? and Correo LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 9).Value = TextBox\_DNI.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_APELLIDOS.Text  comando.Parameters.Add("@var3", OleDbType.VarChar, 120).Value = TextBox\_CORREO.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' DNI Nombre Correo  If TextBox\_DNI.Text <> "" And TextBox\_NOMBRE.Text <> "" And TextBox\_APELLIDOS.Text = "" And TextBox\_CORREO.Text <> "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 2)  Dim resultado2 As Boolean = validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  If resultado1 = False Then  Exit Sub  ElseIf resultado2 = False Then  Exit Sub  ElseIf resultado3 = False Then  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE DNI LIKE? and Nombre LIKE? and Correo LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 9).Value = TextBox\_DNI.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_NOMBRE.Text  comando.Parameters.Add("@var3", OleDbType.VarChar, 120).Value = TextBox\_CORREO.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' DNI Nombre  If TextBox\_DNI.Text <> "" And TextBox\_NOMBRE.Text <> "" And TextBox\_APELLIDOS.Text = "" And TextBox\_CORREO.Text = "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 2)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  If resultado1 = False Then  Exit Sub  ElseIf resultado3 = False Then  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE DNI LIKE? and Nombre LIKE? ", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 9).Value = TextBox\_DNI.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_NOMBRE.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' DNI Apellidos  If TextBox\_DNI.Text <> "" And TextBox\_NOMBRE.Text = "" And TextBox\_APELLIDOS.Text <> "" And TextBox\_CORREO.Text = "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 2)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  If resultado1 = False Then  Exit Sub  ElseIf resultado4 = False Then  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE DNI LIKE? and Apellidos LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 9).Value = TextBox\_DNI.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_APELLIDOS.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' Nombre Apellidos  If TextBox\_DNI.Text = "" And TextBox\_NOMBRE.Text <> "" And TextBox\_APELLIDOS.Text <> "" And TextBox\_CORREO.Text = "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  If resultado3 = False Then  Exit Sub  ElseIf resultado4 = False Then  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE Nombre LIKE? and Apellidos LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 50).Value = TextBox\_NOMBRE.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 50).Value = TextBox\_APELLIDOS.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' Apellidos Correo  If TextBox\_DNI.Text = "" And TextBox\_NOMBRE.Text = "" And TextBox\_APELLIDOS.Text <> "" And TextBox\_CORREO.Text <> "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado2 As Boolean = validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  If resultado2 = False Then  'MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 9.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado4 = False Then  'MsgBox(" El dato apellidos, no puede contener caracteres que sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 50.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE Apellidos LIKE? and Correo LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 50).Value = TextBox\_APELLIDOS.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 120).Value = TextBox\_CORREO.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' DNI Correo  If TextBox\_DNI.Text <> "" And TextBox\_NOMBRE.Text = "" And TextBox\_APELLIDOS.Text = "" And TextBox\_CORREO.Text <> "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 2)  Dim resultado2 As Boolean = validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  If resultado1 = False Then  'MsgBox(" El dato numero socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 8.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  ElseIf resultado2 = False Then  'MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, tampoco puede tener una longitud mayor a 9.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE DNI LIKE? and Correo LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 9).Value = TextBox\_DNI.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 120).Value = TextBox\_CORREO.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' Nombre Correo  If TextBox\_DNI.Text = "" And TextBox\_NOMBRE.Text <> "" And TextBox\_APELLIDOS.Text = "" And TextBox\_CORREO.Text <> "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado2 As Boolean = validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  If resultado2 = False Then  Exit Sub  ElseIf resultado3 = False Then  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE Nombre LIKE? and Correo LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 50).Value = TextBox\_NOMBRE.Text  comando.Parameters.Add("@var2", OleDbType.VarChar, 120).Value = TextBox\_CORREO.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' DNI  If TextBox\_DNI.Text <> "" And TextBox\_NOMBRE.Text = "" And TextBox\_APELLIDOS.Text = "" And TextBox\_CORREO.Text = "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 2)  If resultado1 = False Then  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE DNI LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 9).Value = TextBox\_DNI.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' Nombre  If TextBox\_DNI.Text = "" And TextBox\_NOMBRE.Text <> "" And TextBox\_APELLIDOS.Text = "" And TextBox\_CORREO.Text = "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  If resultado3 = False Then  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE Nombre LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 50).Value = TextBox\_NOMBRE.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' Apellido  If TextBox\_DNI.Text = "" And TextBox\_NOMBRE.Text = "" And TextBox\_APELLIDOS.Text <> "" And TextBox\_CORREO.Text = "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  If resultado4 = False Then  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE Apellidos LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 50).Value = TextBox\_APELLIDOS.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  ' Correo  If TextBox\_DNI.Text = "" And TextBox\_NOMBRE.Text = "" And TextBox\_APELLIDOS.Text = "" And TextBox\_CORREO.Text <> "" Then  ' Validamos.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado2 As Boolean = validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  If resultado2 = False Then  Exit Sub  End If  Dim ds As New DataSet  Dim cb As New OleDbDataAdapter  Dim comando As New OleDbCommand("Select \* from Empleados WHERE Correo LIKE?", conexion)  cb.SelectCommand = comando  comando.Parameters.Add("@var1", OleDbType.VarChar, 120).Value = TextBox\_CORREO.Text  midataset.Clear()  cb.Fill(midataset, "Empleados")  DataGridView\_Empleados.DataSource = midataset  DataGridView\_Empleados.DataMember = "Empleados"  End If  End If ' IF 1  Catch ex As System.InvalidOperationException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex2 As System.FormatException  ' Avisamos del error por mensaje  MsgBox("El formato de los datos introducidos es incorrecto, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex3 As System.Data.OleDb.OleDbException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, es la sintaxis correcta?, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex4 As System.NullReferenceException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, intentalo de nuevo. Referencia a objeto no establecida como instancia de un objeto.", MsgBoxStyle.OkOnly, "Operación invalida")  End Try  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 8 caracteres.  Private Sub TextBox\_NumeroSocio\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_DNI.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 9 caracteres.  Private Sub TextBox\_Telefono\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_CORREO.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 50 caracteres.  Private Sub TextBox\_Nombre\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_NOMBRE.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  End Sub  Private Sub TextBox\_Apellidos\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_APELLIDOS.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  End Sub  ' Esté método es ejecutado cuando el usuario presiona la tecla "F1"  Private Sub GestionSocios\_HelpRequested(sender As Object, hlpevent As HelpEventArgs) Handles Me.HelpRequested  Help.ShowHelp(Me, "CHM\LaCasaDelLibro.chm", "")  End Sub  ' Esté método es ejecutado cuando el usuario presiona el botón de ayuda situado en la barra superior.  ' Muestra un documento de ayuda.  Private Sub VerLaAyudaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles VerLaAyudaToolStripMenuItem.Click  Help.ShowHelp(Me, "CHM\LaCasaDelLibro.chm", "")  End Sub  End Class |

### 8.14 Formulario GestionEmpleadosAltas

#### Explicación de formulario

Este formulario permite al usuario introducir nuevos empleados en la base de datos.

Para ello el usuario debe rellenar todos los campos y presionar el botón “Alta”.

El formulario se cierra tras realizar la operación.



|  |
| --- |
| ' Necesitamos importar el módelo de base de datos que vamos a utilizar, este es de access.  Imports System.Data.OleDb  ' No hace falta hacer imports libValidaciones para instanciar sus clases porque está incluida en el proyecto.  Imports System.IO  Public Class GestionEmpleadosAltas  ' Especificamos la base de datos a la que nos vamos a conectar.  Public conexion As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=cas\_lib\_dib.accdb")  ' Al adaptador le asignamos la conexion que acabamos de realizar y una consulta  Public adaptador As New OleDbDataAdapter("Select \* from Empleados", conexion)  ' Aquí alojaremos los datos de la DB  Public midataset As New DataSet  ' Método que se ejecuta cuando el botón "Salir..." del ToolStrip es pulsado y que nos lleva al formulario "GestionEmpleados".  Private Sub SalirToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles SalirToolStripMenuItem.Click  ' Mostramos el formulario "GestionEmpleados".  GestionEmpleados.Show()  ' Reiniciamos el valor para la próxima vez  controlCalculadora = 0  ' Cerramos este formulario  Me.Close()  End Sub  Dim controlCalculadora As Integer  ' Método que se ejecuta cuando es pulsado el botón "Calculadora" del menuStrip  Private Sub CalculadoraToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles CalculadoraToolStripMenuItem.Click  If controlCalculadora < 2 Then  ' Try catch para atrapar el error en caso de que el ordenador del usuario  Try  Dim program As String  program = "calc.exe"  Process.Start(program)  Catch ex As System.ComponentModel.Win32Exception '  MsgBox("Ha ocurrido un error, no se pudo iniciar la calculadora.", MsgBoxStyle.OkOnly, "Error (proceso calculadora)")  End Try  controlCalculadora = controlCalculadora + 1  Else  MsgBox("Solo se pueden iniciar dos calculadoras por sesión.", MsgBoxStyle.OkOnly, "Aviso")  End If  End Sub  ' Método que se ejecuta al pulsar el botón "Limpiar"  ' Limpia todos los textBoxes  Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click  TextBox\_DNI.Clear()  TextBox\_NOMBRE.Clear()  TextBox\_APELLIDOS.Clear()  TextBox\_CORREO.Clear()  TextBox\_TELEFONO.Clear()  TextBox\_USUARIO.Clear()  TextBox\_CONTRASENNA.Clear()  TextBox\_ROL.Clear()  End Sub  ' Método que se ejecuta al iniciarse el formulario.  Private Sub GestionEmpleadosAltas\_Load(sender As Object, e As EventArgs) Handles MyBase.Load  ' Cargar la memoria del cache con datos.  adaptador.Fill(midataset, "Empleados")  If GestionEmpleados.numeroDeControlBindingAltaEmpleados = 0 Then  ' Relacionar los campos de la tabla con los textbox  Me.TextBox\_DNI.DataBindings.Add("text", midataset, "Empleados.DNI")  Me.TextBox\_NOMBRE.DataBindings.Add("text", midataset, "Empleados.Nombre")  Me.TextBox\_APELLIDOS.DataBindings.Add("text", midataset, "Empleados.Apellidos")  Me.TextBox\_CORREO.DataBindings.Add("text", midataset, "Empleados.Correo")  Me.TextBox\_TELEFONO.DataBindings.Add("text", midataset, "Empleados.Telefono")  Me.TextBox\_USUARIO.DataBindings.Add("text", midataset, "Empleados.Usuario")  Me.TextBox\_CONTRASENNA.DataBindings.Add("text", midataset, "Empleados.Contrasenna")  Me.TextBox\_ROL.DataBindings.Add("text", midataset, "Empleados.Rol")  GestionEmpleados.numeroDeControlBindingAltaEmpleados = 1  End If  ' Inicializamos la variable de control  controlCalculadora = 0  ' Vaciamos cada textBox de forma individual  TextBox\_DNI.Clear()  TextBox\_NOMBRE.Clear()  TextBox\_APELLIDOS.Clear()  TextBox\_CORREO.Clear()  TextBox\_TELEFONO.Clear()  TextBox\_USUARIO.Clear()  TextBox\_CONTRASENNA.Clear()  TextBox\_ROL.Clear()  End Sub  ' Método que se ejecuta cuando el botón "Alta" es pulsado.  ' Introduce los datos escritos por el usuario en los textBox en la DB.  Private Sub Button\_Guardar\_Alta\_Click(sender As Object, e As EventArgs) Handles Button\_Guardar\_Alta.Click  If TextBox\_DNI.Text = "" Or TextBox\_NOMBRE.Text = "" Or TextBox\_APELLIDOS.Text = "" Or TextBox\_TELEFONO.Text = "" Or TextBox\_CORREO.Text = "" Or TextBox\_USUARIO.Text = "" Or TextBox\_CONTRASENNA.Text = "" Or TextBox\_ROL.Text = "" Then  MsgBox("No se puede dar de alta , debe rellenar todos los datos.", MsgBoxStyle.OkOnly, "Error al dar de alta.")  Else  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 2)  Dim resultado2 As Boolean = validarNumeroSocio.validarTelefono(TextBox\_TELEFONO.Text, 2)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  Dim resultado5 As Boolean = validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  Dim resultado6 As Boolean = validarNumeroSocio.validarUsuario(TextBox\_USUARIO.Text)  Dim resultado7 As Boolean = validarNumeroSocio.validarContra(TextBox\_CONTRASENNA.Text, 2)  Dim resultado8 As Boolean = validarNumeroSocio.validarROL(TextBox\_ROL.Text, 1)  If resultado1 = False Then  Exit Sub  ElseIf resultado2 = False Then  Exit Sub  ElseIf resultado3 = False Then  Exit Sub  ElseIf resultado4 = False Then  Exit Sub  ElseIf resultado5 = False Then  Exit Sub  ElseIf resultado6 = False Then  Exit Sub  ElseIf resultado7 = False Then  Exit Sub  ElseIf resultado8 = False Then  Exit Sub  End If  Dim testComp As Integer = StrComp(TextBox\_ROL.Text, "Admin", CompareMethod.Text)  Dim testComp2 As Integer = StrComp(TextBox\_ROL.Text, "Encargado", CompareMethod.Text)  Dim testComp3 As Integer = StrComp(TextBox\_ROL.Text, "Empleado", CompareMethod.Text)  If testComp <> 0 Then  If testComp2 <> 0 Then  If testComp3 <> 0 Then  MsgBox(" El rol debe ser uno de los siguientes: Admin, Encargado o Empleado.", MsgBoxStyle.OkOnly, "Error - Rol incorrecto")  Exit Sub  End If  End If  End If  Dim valor As String  Dim control As Integer = 0  ' Comprobamos que la clave primaria no se encuentra ya registrada.  For contador As Integer = 0 To GestionEmpleados.DataGridView\_Empleados.RowCount - 1  valor = GestionEmpleados.DataGridView\_Empleados.Item(0, contador).Value  If valor = TextBox\_DNI.Text Then  MsgBox("No puedes introducir un DNI que ya existe en la base de datos.", MsgBoxStyle.OkOnly, "Error, clave duplicada")  control = 1  End If  Next  If control = 0 Then  Try  ' #################### 1º Preparamos a la base de datos para recibir los datos. ##############################  Dim cb As New OleDbCommandBuilder(adaptador)  adaptador.InsertCommand = cb.GetInsertCommand  ' #################### 2º Recogemos los datos y los introducimos ##############################  Dim drc As DataRowCollection = midataset.Tables("Empleados").Rows  drc.Add(TextBox\_DNI.Text, TextBox\_NOMBRE.Text, TextBox\_APELLIDOS.Text, TextBox\_CORREO.Text, TextBox\_TELEFONO.Text, TextBox\_USUARIO.Text, TextBox\_CONTRASENNA.Text, TextBox\_ROL.Text)  adaptador.Update(midataset.Tables("Empleados"))  ' #################### 3º Actualizamos el middataset ##############################  ' Actualizamos el dataGridView del formulario de gestión principal  GestionEmpleados.midataset.Clear()  GestionEmpleados.adaptador.Fill(GestionEmpleados.midataset, "Empleados")  ' Reiniciamos su valor para la proxima vez  controlCalculadora = 0  ' Cerramos la ventana  Me.Close()  Catch ex As System.InvalidOperationException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex2 As System.FormatException  ' Avisamos del error por mensaje  MsgBox("El formato de los datos introducidos es incorrecto, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex3 As System.Data.OleDb.OleDbException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, es la sintaxis correcta?, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex4 As System.NullReferenceException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, intentalo de nuevo. Referencia a objeto no establecida como instancia de un objeto.", MsgBoxStyle.OkOnly, "Operación invalida")  End Try  End If  End If  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 8 caracteres.  Private Sub TextBox\_DNI\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_DNI.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 9 caracteres.  Private Sub TextBox\_Telefono\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_TELEFONO.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarTelefono(TextBox\_TELEFONO.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de no más de 50 caracteres.  Private Sub TextBox\_Nombre\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_NOMBRE.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de no más de 50 caracteres.  Private Sub TextBox\_Apellidos\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_APELLIDOS.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  End Sub  ' Validamos este campo evitando que tenga caracteres prohibidos en un e-mail y que tenga una longitud de no más de 120 caracteres.  Private Sub TextBox\_Correo\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_CORREO.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  End Sub  ' Validación del usuario  Private Sub TextBox\_USUARIO\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_USUARIO.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarUsuario(TextBox\_USUARIO.Text)  End Sub  ' Validación de la contraseña  Private Sub TextBox\_CONTRASENNA\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_CONTRASENNA.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarContra(TextBox\_CONTRASENNA.Text, 1)  End Sub  ' Validación del rol  Private Sub TextBox\_ROL\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_ROL.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarROL(TextBox\_ROL.Text, 1)  End Sub  ' Esté método es ejecutado cuando el usuario presiona la tecla "F1"  Private Sub GestionSocios\_HelpRequested(sender As Object, hlpevent As HelpEventArgs) Handles Me.HelpRequested  Help.ShowHelp(Me, "CHM\LaCasaDelLibro.chm", "")  End Sub  ' Esté método es ejecutado cuando el usuario presiona el botón de ayuda situado en la barra superior.  ' Muestra un documento de ayuda.  Private Sub VerLaAyudaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles VerLaAyudaToolStripMenuItem.Click  Help.ShowHelp(Me, "CHM\LaCasaDelLibro.chm", "")  End Sub  End Class |

### 8.15 Formulario GestionEmpleadosModificaciones

#### Explicación de formulario

Este formulario permite al usuario realizar la modificación de un registro.

Para ello debe modificar los datos ya existentes y sustituirlos por los que desee. Tras pulsar el botón “Modificar”, el registro es modificado y el formulario se cierra.



|  |
| --- |
| ' Necesitamos importar el módelo de base de datos que vamos a utilizar, este es de access.  Imports System.Data.OleDb  ' No hace falta hacer imports libValidaciones para instanciar sus clases porque está incluida en el proyecto.  Imports System.IO  Public Class GestionEmpleadosModificaciones  ' Especificamos la base de datos a la que nos vamos a conectar.  Public conexion As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=cas\_lib\_dib.accdb")  ' Al adaptador le asignamos la conexion que acabamos de realizar y una consulta  Public adaptador As New OleDbDataAdapter("Select \* from Empleados", conexion)  ' Aquí alojaremos los datos de la DB  Public midataset As New DataSet  Dim numDniInicial As String  ' Método que se ejecuta al iniciarse el formulario  Private Sub GestionEmpleadosModificaciones\_Load(sender As Object, e As EventArgs) Handles MyBase.Load  'Creamos un comando nuevo con una Query nueva la cual vamos a usar para poder buscar un socio en concreto  Dim comando As New OleDbCommand(("select \* from Empleados where DNI = @numDNI"), conexion)  'Al adaptador le asignamos este nuevo comando "comando"  adaptador.SelectCommand = comando  'Al comando le vamos a pasar por parametro @numSocio (valor de tipo Numeric, 8) el cual va a ser cogido de la Fila(currentRow) seleccionada de la Columna 0 (NumeroSocio)  'del DataGridView\_Socios del formulario GestionSocios. Es decir al seleccionar un elemento detectará cual es la fila y el valor de la columna 0 y lo pasará por parametro.  comando.Parameters.Add("@numDNI", OleDbType.VarChar, 9).Value = GestionEmpleados.DataGridView\_Empleados.Item(0, GestionEmpleados.DataGridView\_Empleados.CurrentRow.Index).Value  'Limpiamos el midataset para que no haya información residual posicionDataGridSeleccionadaEmpleados  midataset.Clear() 'de este form  'Asociamos el nuevo adaptador con el nuevo comando al midataset de la tabla Socios  adaptador.Fill(midataset, "Empleados")  If GestionEmpleados.numeroDeControlBindingModificacionesEmpleados = 0 Then  'Se relacionan los campos de la tabla con los textbox y se muestran los datos del registro que queremos modificar.  Me.TextBox\_DNI.DataBindings.Add("text", midataset, "Empleados.DNI")  Me.TextBox\_NOMBRE.DataBindings.Add("text", midataset, "Empleados.Nombre")  Me.TextBox\_APELLIDOS.DataBindings.Add("text", midataset, "Empleados.Apellidos")  Me.TextBox\_CORREO.DataBindings.Add("text", midataset, "Empleados.Correo")  Me.TextBox\_TELEFONO.DataBindings.Add("text", midataset, "Empleados.Telefono")  Me.TextBox\_USUARIO.DataBindings.Add("text", midataset, "Empleados.Usuario")  Me.TextBox\_CONTRASENNA.DataBindings.Add("text", midataset, "Empleados.Contrasenna")  Me.TextBox\_ROL.DataBindings.Add("text", midataset, "Empleados.Rol")  GestionEmpleados.numeroDeControlBindingModificacionesEmpleados = 1  End If  ' Inicializamos la variable de control  controlCalculadora = 0  ' Inicializamos la variable asignandole el número de socio inicial  numDniInicial = GestionEmpleados.DataGridView\_Empleados.Item(0, GestionEmpleados.DataGridView\_Empleados.CurrentRow.Index).Value  Try  Catch ex As System.FormatException  MsgBox("Se ha producido un error, intentelo de nuevo", MsgBoxStyle.OkOnly, "Error - Operación abortada")  End Try  End Sub  ' Método que se ejecuta cuando se pulsa el botón "Limpiar".  ' "Limpia" todos los textBox, dejandolos vacios.  Private Sub Button1\_Click(sender As Object, e As EventArgs) Handles Button1.Click  TextBox\_DNI.Clear()  TextBox\_NOMBRE.Clear()  TextBox\_APELLIDOS.Clear()  TextBox\_CORREO.Clear()  TextBox\_TELEFONO.Clear()  TextBox\_USUARIO.Clear()  TextBox\_CONTRASENNA.Clear()  TextBox\_ROL.Clear()  End Sub  ' Método que se ejecuta cuando el botón "Salir..." del ToolStrip es pulsado y que nos lleva al formulario "GestionEmpleados".  Private Sub SalirToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles SalirToolStripMenuItem.Click  ' Mostramos el formulario "GestionEmpleados".  GestionEmpleados.Show()  ' Reiniciamos el valor para la próxima vez  controlCalculadora = 0  ' Cerramos este formulario  Me.Close()  End Sub  Dim controlCalculadora As Integer  ' Método que se ejecuta cuando es pulsado el botón "Calculadora" del menuStrip  Private Sub CalculadoraToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles CalculadoraToolStripMenuItem.Click  If controlCalculadora < 2 Then  ' Try catch para atrapar el error en caso de que el ordenador del usuario  Try  Dim program As String  program = "calc.exe"  Process.Start(program)  Catch ex As System.ComponentModel.Win32Exception '  MsgBox("Ha ocurrido un error, no se pudo iniciar la calculadora.", MsgBoxStyle.OkOnly, "Error (proceso calculadora)")  End Try  controlCalculadora = controlCalculadora + 1  Else  MsgBox("Solo se pueden iniciar dos calculadoras por sesión.", MsgBoxStyle.OkOnly, "Aviso")  End If  End Sub  ' Método que se ejecuta al pulsarse el botón "Modificar"  ' Se encarga de modificar los datos ya existentes en la DB  Private Sub Button\_Guardar\_Alta\_Click(sender As Object, e As EventArgs) Handles Button\_Guardar\_Alta.Click  If TextBox\_DNI.Text = "" Or TextBox\_NOMBRE.Text = "" Or TextBox\_APELLIDOS.Text = "" Or TextBox\_TELEFONO.Text = "" Or TextBox\_CORREO.Text = "" Or TextBox\_USUARIO.Text = "" Or TextBox\_CONTRASENNA.Text = "" Or TextBox\_ROL.Text = "" Then  MsgBox("Debes seleccionar un registro para actualizarlo y si lo has seleccionado, no debe quedar ningún campo en blanco", MsgBoxStyle.OkOnly, "Error al modificar.")  Else  ' Validamos todas las cajas y si alguna es incorrecta... salimos del metodo.  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  Dim resultado1 As Boolean = validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 2)  Dim resultado2 As Boolean = validarNumeroSocio.validarTelefono(TextBox\_TELEFONO.Text, 2)  Dim resultado3 As Boolean = validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  Dim resultado4 As Boolean = validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  Dim resultado5 As Boolean = validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  Dim resultado6 As Boolean = validarNumeroSocio.validarUsuario(TextBox\_USUARIO.Text)  Dim resultado7 As Boolean = validarNumeroSocio.validarContra(TextBox\_CONTRASENNA.Text, 2)  Dim resultado8 As Boolean = validarNumeroSocio.validarROL(TextBox\_ROL.Text, 1)  If resultado1 = False Then  Exit Sub  ElseIf resultado2 = False Then  Exit Sub  ElseIf resultado3 = False Then  Exit Sub  ElseIf resultado4 = False Then  Exit Sub  ElseIf resultado5 = False Then  Exit Sub  ElseIf resultado6 = False Then  Exit Sub  ElseIf resultado7 = False Then  Exit Sub  ElseIf resultado8 = False Then  Exit Sub  End If  Dim testComp As Integer = StrComp(TextBox\_ROL.Text, "Admin", CompareMethod.Text)  Dim testComp2 As Integer = StrComp(TextBox\_ROL.Text, "Encargado", CompareMethod.Text)  Dim testComp3 As Integer = StrComp(TextBox\_ROL.Text, "Empleado", CompareMethod.Text)  If testComp <> 0 Then  If testComp2 <> 0 Then  If testComp3 <> 0 Then  MsgBox(" El rol debe ser uno de los siguientes: Admin, Encargado o Empleado.", MsgBoxStyle.OkOnly, "Error - Rol incorrecto")  Exit Sub  End If  End If  End If  Dim valor As String  Dim control As Integer = 0  ' Comprobamos que la clave primaria no se encuentra ya registrada.  For contador As Integer = 0 To GestionEmpleados.DataGridView\_Empleados.RowCount - 1  valor = GestionEmpleados.DataGridView\_Empleados.Item(0, contador).Value  If valor = TextBox\_DNI.Text And valor <> numDniInicial Then  MsgBox("No puedes introducir un DNI que ya existe en la base de datos.", MsgBoxStyle.OkOnly, "Error, clave duplicada")  control = 1  End If  Next  If control = 0 Then  Dim cb As New OleDbCommandBuilder(GestionEmpleados.adaptador)  GestionEmpleados.adaptador.UpdateCommand = cb.GetUpdateCommand  Dim a As Integer = BindingContext(GestionEmpleados.midataset, "Empleados").Position  Dim fila As DataRow = GestionEmpleados.midataset.Tables("Empleados").Rows(a)  Try  'Montamos una query parametrizada.  Dim queryParametrizada As String = "UPDATE Empleados SET DNI=?, Nombre=?, Apellidos=?, Correo=?, Telefono=?, Usuario=?, Contrasenna=?, Rol=? WHERE DNI=?"  Using cmd = New OleDbCommand(queryParametrizada, conexion)  conexion.Open()  cmd.Parameters.AddWithValue("@p1", TextBox\_DNI.Text)  cmd.Parameters.AddWithValue("@p2", TextBox\_NOMBRE.Text)  cmd.Parameters.AddWithValue("@p3", TextBox\_APELLIDOS.Text)  cmd.Parameters.AddWithValue("@p4", TextBox\_CORREO.Text)  cmd.Parameters.AddWithValue("@p5", Convert.ToSingle(TextBox\_TELEFONO.Text))  cmd.Parameters.AddWithValue("@p6", TextBox\_USUARIO.Text)  cmd.Parameters.AddWithValue("@p7", Convert.ToInt16(TextBox\_CONTRASENNA.Text))  cmd.Parameters.AddWithValue("@p8", TextBox\_ROL.Text)  cmd.Parameters.AddWithValue("@p9", numDniInicial)  cmd.ExecuteNonQuery()  conexion.Close()  End Using  'Dim cb As New OleDbCommandBuilder(adaptador)  'adaptador.UpdateCommand = cb.GetUpdateCommand  Catch ex As System.InvalidOperationException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex2 As System.FormatException  ' Avisamos del error por mensaje  MsgBox("El formato de los datos introducidos es incorrecto, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  Catch ex3 As System.Data.OleDb.OleDbException  ' Avisamos del error por mensaje  MsgBox("Algo no ha ido bien, intentalo de nuevo", MsgBoxStyle.OkOnly, "Operación invalida")  End Try  ' Actualizamos el dataGridView del formulario de gestión principal  GestionEmpleados.midataset.Clear()  GestionEmpleados.adaptador.Fill(GestionEmpleados.midataset, "Empleados")  ' Reiniciamos su valor para la próxima vez  controlCalculadora = 0  ' Cerramos la ventana  Me.Close()  ' Hacer try-catch  End If  End If  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 8 caracteres.  Private Sub TextBox\_DNI\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_DNI.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarDNI(TextBox\_DNI.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de más de 9 caracteres.  Private Sub TextBox\_Telefono\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_TELEFONO.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarTelefono(TextBox\_TELEFONO.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de no más de 50 caracteres.  Private Sub TextBox\_Nombre\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_NOMBRE.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNombre(TextBox\_NOMBRE.Text, 1)  End Sub  ' Validamos este campo evitando que tenga caracteres que no sean númericos y que tenga una longitud de no más de 50 caracteres.  Private Sub TextBox\_Apellidos\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_APELLIDOS.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarNombre(TextBox\_APELLIDOS.Text, 2)  End Sub  ' Validamos este campo evitando que tenga caracteres prohibidos en un e-mail y que tenga una longitud de no más de 120 caracteres.  Private Sub TextBox\_Correo\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_CORREO.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarCorreo(TextBox\_CORREO.Text, 1)  End Sub  ' Validación del usuario  Private Sub TextBox\_USUARIO\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_USUARIO.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarUsuario(TextBox\_USUARIO.Text)  End Sub  ' Validación de la contraseña  Private Sub TextBox\_CONTRASENNA\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_CONTRASENNA.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarContra(TextBox\_CONTRASENNA.Text, 1)  End Sub  ' Validación del rol  Private Sub TextBox\_ROL\_TextChanged(sender As Object, e As EventArgs) Handles TextBox\_ROL.TextChanged  ' Instanciamos la clase  Dim validarNumeroSocio As New libreriaValidacion.Validacion  validarNumeroSocio.validarROL(TextBox\_ROL.Text, 1)  End Sub  ' Esté método es ejecutado cuando el usuario presiona la tecla "F1"  Private Sub GestionSocios\_HelpRequested(sender As Object, hlpevent As HelpEventArgs) Handles Me.HelpRequested  Help.ShowHelp(Me, "CHM\LaCasaDelLibro.chm", "")  End Sub  ' Esté método es ejecutado cuando el usuario presiona el botón de ayuda situado en la barra superior.  ' Muestra un documento de ayuda.  Private Sub VerLaAyudaToolStripMenuItem\_Click(sender As Object, e As EventArgs) Handles VerLaAyudaToolStripMenuItem.Click  Help.ShowHelp(Me, "CHM\LaCasaDelLibro.chm", "")  End Sub  End Class |

### 8.16 Clase Validacion

#### Explicación de clase

Esta clase contiene los métodos utilizados para la validación de todos los formularios.

Estos métodos son:

* validarDNI, que se asegura de que el DNI tiene el formato correcto.
* validarNumeroSocio, que se asegura de que solo se pueden escribir números.
* validarTelefono, que se asegura de que solo se puedan escribir números.
* validarNombre, evita que se puedan introducir números o otros simbolos.
* validarUsuario, evita que se puedan introducir ciertos simbolos.
* validarContra, evita que se puedan introducir caracteres que no sean números.
* validarRol, evita caracteres no permitidos.
* validarCorreo, evita que se puedan introducir ciertos simbolos.
* ValidarISBN, evita caracteres no númericos.
* validarTitulolibro, evita simbolos especiales.
* validarAutor, evita números.
* validar4digitos, valida una longitud de 4.
* validarEditorial, evita números y simbolos.
* validarIdioma, evita números y simbolos.
* validarPlazaEdicion, evita números y simbolos.
* validarTraductor, evita números y simbolos.
* validarPrecio, evita carateres no númericos.
* validarNombreProducto, evita simbolos.
* validarCategoria, evita números y simbolos.

|  |
| --- |
| Imports System.IO  Public Class Validacion  ' Validación de DNI  Public Function validarDNI(codigo As String, tipo As Integer) As Boolean  ' Si el tipo es 1, solo se comprueba que no se pase de 8 caracteres y que estos sean solo númericos  ' Si el tipo es 2, comprueba todo, incluyendo que la minima longitud sea de 8 caracteres.  ' Se opta de nuevo por la solución de poner en un String los valores permitidos.  Dim charsPosibles As String = "ABCDEFGHIJKLMNÑOPQRSTUVWXYZ0123456789"  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 9  If tipo = 1 Then  For i = 1 To codigo.Length  For j = 1 To charsPosibles.Length  If (GetChar(codigo, i) <> GetChar(charsPosibles, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length <= valorMaximo Then  valido = True  Else  MsgBox(" El dato DNI, no puede contener caracteres que no sean numéricos o símbolos no permitidos, debe tener una longitud de 9 digitos. La letra debe estar en mayúscula.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  'MsgBox(" El número de socio no puede contener otros caracteres que no sean números.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End If  If tipo = 2 Then  For i = 1 To codigo.Length  For j = 1 To charsPosibles.Length  If (GetChar(codigo, i) <> GetChar(charsPosibles, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length = valorMaximo Then  valido = True  Else  MsgBox(" El dato DNI, no puede contener caracteres que no sean numéricos o símbolos no permitidos, debe tener una longitud de 9 digitos. La letra debe estar en mayúscula.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  'MsgBox(" El número de socio no puede contener otros caracteres que no sean números.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End If  End Function  ' Validación de número de socio  Public Function validarNumeroSocio(codigo As String, tipo As Integer) As Boolean  ' Si el tipo es 1, solo se comprueba que no se pase de 8 caracteres y que estos sean solo númericos  ' Si el tipo es 2, comprueba todo, incluyendo que la minima longitud sea de 8 caracteres.  ' Se opta de nuevo por la solución de poner en un String los valores permitidos.  Dim charsPosibles As String = "0123456789"  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 8  If tipo = 1 Then  For i = 1 To codigo.Length  For j = 1 To charsPosibles.Length  If (GetChar(codigo, i) <> GetChar(charsPosibles, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length <= valorMaximo Then  valido = True  Else  MsgBox(" El dato número socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, debe tener una longitud de 8 digitos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  'MsgBox(" El número de socio no puede contener otros caracteres que no sean números.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End If  If tipo = 2 Then  For i = 1 To codigo.Length  For j = 1 To charsPosibles.Length  If (GetChar(codigo, i) <> GetChar(charsPosibles, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length = valorMaximo Then  valido = True  Else  MsgBox(" El dato número socio, no puede contener caracteres que no sean numéricos o símbolos no permitidos, debe tener una longitud de 8 digitos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  'MsgBox(" El número de socio no puede contener otros caracteres que no sean números.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End If  'Solo se permiten códigos de hasta 8 números.  'If codigo.Length > valorMaximo Then  'MsgBox("Por favor, introduzca un número de socio que sea de 8 dígitos", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")  'End If  End Function  ' Validación de número de teléfono  Public Function validarTelefono(codigo As String, tipo As Integer) As Boolean  ' Se opta de nuevo por la solución de poner en un String los valores permitidos.  Dim charsPosibles As String = "0123456789"  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 9  ' Solo se permiten códigos de hasta 9 números.  'If codigo.Length > valorMaximo Then  'MsgBox("Por favor, introduzca un número de teléfono que sea menor de 8 dígitos", MsgBoxStyle.OkOnly, "Longitud incorrecta.")  'End If  If tipo = 1 Then  For i = 1 To codigo.Length  For j = 1 To charsPosibles.Length  If (GetChar(codigo, i) <> GetChar(charsPosibles, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length <= valorMaximo Then  valido = True  Else  MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, también debe tener una longitud de 9 digitos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End If  If tipo = 2 Then  For i = 1 To codigo.Length  For j = 1 To charsPosibles.Length  If (GetChar(codigo, i) <> GetChar(charsPosibles, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length = valorMaximo Then  valido = True  Else  MsgBox(" El dato telefono, no puede contener caracteres que no sean numéricos o símbolos no permitidos, también debe tener una longitud de 9 digitos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End If  End Function  ' Validación de nombre  Public Function validarNombre(codigo As String, tipo As Integer) As Boolean  ' Si tipo es 1, se realizara la validación de un nombre.  ' Si tipo es 2, se realizara la validación de un apellido.  ' Para el nombre.  Dim charsPosiblesNombre As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZáéíóúÁÉÍÓÚ "  Dim charsPosiblesApellido As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZáéíóúÁÉÍÓÚ- "  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 50  ' Solo se permiten nombres de hasta 50 números.  If codigo.Length > valorMaximo Then  MsgBox("Por favor, introduzca un nombre que sea menor de 50 caracteres.", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")  End If  ' Validación nombre  If tipo = 1 Then  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  End If  ' Validación apellido  If tipo = 2 Then  For i = 1 To codigo.Length  For j = 1 To charsPosiblesApellido.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesApellido, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  End If  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length Then  valido = True  Else  MsgBox(" El nombre/apellido no puede contener caracteres numéricos o símbolos no permitidos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  ' Validación de usuario  Public Function validarUsuario(codigo As String) As Boolean  Dim charsPosiblesUsuario As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZ1234567890-\_"  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 30  ' Solo se permiten usuarios de hasta 30 caracteres.  If codigo.Length > valorMaximo Then  MsgBox("Por favor, introduzca un usuario que sea menor de 31 caracteres.", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")  End If  For i = 1 To codigo.Length  For j = 1 To charsPosiblesUsuario.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesUsuario, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length Then  valido = True  Else  MsgBox(" El usuario no puede contener símbolos no permitidos. Debe tener una longitud máxima de 30 caracteres", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  ' Validación de contraseña  Public Function validarContra(codigo As String, tipo As Integer) As Boolean  Dim charsPosiblesNombre As String = "1234567890"  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 4  If tipo = 1 Then  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length <= valorMaximo Then  valido = True  Else  MsgBox(" La contraseña no puede contener caracteres no permitidos. Debe tener una longitud de 4 digitos", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End If  If tipo = 2 Then  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length = valorMaximo Then  valido = True  Else  MsgBox(" La contraseña no puede contener caracteres no permitidos. Debe tener una longitud de 4 digitos", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End If  End Function  ' Validación del ROL  Public Function validarROL(codigo As String, tipo As Integer) As Boolean  Dim charsPosiblesNombre As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZ"  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 20  If tipo = 1 Then  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length <= valorMaximo Then  valido = True  Else  MsgBox(" El rol no puede contener caracteres no permitidos. Debe tener una longitud menor de 21 caracteres", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End If  If tipo = 2 Then  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length = valorMaximo Then  valido = True  Else  MsgBox(" El rol no puede contener caracteres no permitidos. Debe tener una longitud menor de 21 caracteres", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End If  End Function  ' Validación de correo electrónico  Public Function validarCorreo(codigo As String, tipo As Integer) As Boolean  ' Si el tipo es 1, solo se comprueba que no se pase de 120 caracteres  ' Si el tipo es 2, comprueba todo, incluyendo que la minima longitud sea de 120 caracteres.  ' Para el correo.  Dim charsPosiblesCorreo As String = "1234567890abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZ\_-.@!#"  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 120  If tipo = 1 Then  For i = 1 To codigo.Length  For j = 1 To charsPosiblesCorreo.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesCorreo, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length <= valorMaximo Then  valido = True  Else  MsgBox(" El correo electrónico no puede contener símbolos no permitidos o ser mayor de 120 caracteres.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  'MsgBox(" El número de socio no puede contener otros caracteres que no sean números.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End If  If tipo = 2 Then  For i = 1 To codigo.Length  For j = 1 To charsPosiblesCorreo.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesCorreo, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length = valorMaximo Then  valido = True  Else  MsgBox(" El correo electrónico no puede contener símbolos no permitidos o ser mayor de 120 caracteres.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  'MsgBox(" El número de socio no puede contener otros caracteres que no sean números.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End If  End Function  ' Método usado en la pantalla de inicio de sesión para ver que se introducen solo números en el campo de contraseña.  Public Function validarCodigo(codigo As String, tipo As Integer) As Boolean  ' Si se recibe 0 se está validando contraseña en inicio de sesión. (se permiten 4 dígitos)  ' Si se recibe 1 se está validando teléfono. (se permite 9 dígitos).  ' Si se recibe 2 se está validando precios en productos  ' Se opta de nuevo por la solución de poner en un String los valores permitidos.  Dim charsPosibles As String = "0123456789"  ' Para el campo de teléfono se permite usar el carácter especial +.  Dim charsPosiblesTelefono As String = "0123456789+"  Dim charsPosiblesPrecio As String = "0123456789. "  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  If tipo = 0 Then  valorMaximo = 4  ElseIf tipo = 1 Then  valorMaximo = 9  ElseIf tipo = 2 Then  valorMaximo = 5  End If  ' Solo se permiten códigos de hasta 4 números.  ' La validación de que sean 4 se hace en otro método, porque eso depende de la creación de usuarios.  If codigo.Length > valorMaximo Then  MsgBox("Por favor, introduzca una contraseña/código de 4 dígitos", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")  Else  For i = 1 To codigo.Length  If tipo = 0 Then  For j = 1 To charsPosibles.Length  If (GetChar(codigo, i) <> GetChar(charsPosibles, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  ElseIf tipo = 1 Then  For j = 1 To charsPosiblesTelefono.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesTelefono, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  ElseIf tipo = 2 Then  For j = 1 To charsPosiblesPrecio.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesPrecio, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  End If  Next i  End If  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length Then  valido = True  End If  Return valido  End Function  ' Método para comprobar que usuario y contraseña introducidos en la pantalla de inicio  ' coinciden con datos guardados en el fichero secuencial usuarios.txt  Public Function comprobarDatosSecuencial(file As String, usuario As String, contrasena As Integer) As Integer  'si devuelve 0 los datos están mal, si devuelve 1 están bien, y si 2 es el admin el que se está conectando.  Dim coincidenciaNombre As String = ""  Dim coincidenciaContrasena As Integer = 0  Dim correcto As Integer = 0  Dim coincidencia As String  Dim admin As Boolean  Try  ' Este método es como el de coprobar datos para archivos de texto, pero para archivos secuenciales.  ' Devuelve un booleano, true si los datos coinciden, false si los datos no son correctos.  FileOpen(1, file, OpenMode.Input)  While Not EOF(1)  Input(1, coincidenciaNombre)  ' MsgBox("coincidenciaNombre:" & coincidenciaNombre & " y parámetro: " & usuario)  If coincidenciaNombre.Equals(usuario) Then  ' Si mientras se lee el archivo secuencial se encuentra un nombre que coincida con el  ' pasado por parámetro, se pasa a leer el siguiente campo(la contraseña).  Input(1, coincidenciaContrasena)  'comentado, razón: esta linea era para testeo: MsgBox("coincidenciaContrasena:" & coincidenciaContrasena & " y parámetro: " & contrasena)  ' Si la contraseña coincide con la pasada por parámetro, entonces se valida el acceso a la aplicación.  If coincidenciaContrasena = contrasena Then  ' FileOpen(1, "usuarios.txt", OpenMode.Input)  Input(1, coincidencia)  Input(1, coincidencia)  Input(1, coincidencia)  Input(1, coincidencia)  Input(1, admin)  If admin = True Then  correcto = 2  ' Salimos del while porque ya encontramos al usuario.  Exit While  Else  ' Salimos del while porque ya encontramos al usuario.  correcto = 1  Exit While  End If  Else  End If  End If  End While  FileClose(1)  Catch ex As Exception  MsgBox(Err.Description)  End Try  Return correcto  End Function  Public Function errorLogWrite()  ' Escribir en errorLog  Dim fichero = "Errorlog.txt"  Dim fich As New FileStream(fichero, FileMode.Append, FileAccess.Write)  Dim rs As New StreamWriter(fich)  rs.WriteLine(Now & "---> " & "Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number)  rs.Close()  fich.Close()  End Function  Public Function mensajeErrorDatos()  MsgBox("Error en la comprobación de datos." & Chr(13) & "Detalle: " & Err.Description & Chr(13) & "Número de error: " & Err.Number)  errorLogWrite()  End Function  ' Método usado para validar diversos campos  ' Si se recibe el integer 0 = se valida el campo categoría  ' si se recibe el integer 1 = se valida el campo nombre  ' si se recibe el integer 2 = se valida el campo tamaño  Public Function validarCampoProducto(stringRecibido As String, tipo As Integer) As Boolean  ' Entorno:  ' Creamos una lista de caracteres permitidos.  Dim charsPosibles As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZáéíóúÁÉÍÓÚ "  Dim charsPosiblesCategoria As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZáéíóúÁÉÍÓÚ "  Dim charsposiblesTamano As String = "0123456789'. "  ' Las variables donde guardaremos los caracteres para compararlos.  Dim chDeLista As Char  Dim chRecibido As Char  ' El buchón/chivato para ver si el número de aciertos (caracteres equivalentes) es el mismo al número  ' de caracteres del string pasado por parámetro.  Dim coincidencias As Integer = 0  ' Finalmente la variable boolean valido para que en el formulario se pueda saber si  ' el string es válido o no.  Dim valido As Boolean = False  Dim valorMaximo As Integer  ' Asignamos el valor a la variable valorMaximo según el tipo de campo que hemos recibido por parámetro.  If tipo = 0 Or tipo = 1 Then  valorMaximo = 15  ElseIf tipo = 2 Then  valorMaximo = 8  End If  ' Inicio  ' Primero comprobamos el tamaño del string  If stringRecibido.Length > valorMaximo Then  MsgBox("No se permiten más de" & valorMaximo & " caracteres.", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta")  Else  ' Iteramos un número de veces igual al tamaño del string pasado por parámetro.  For i = 1 To stringRecibido.Length  ' Guardamos el carácter en chRecibido.  chRecibido = GetChar(stringRecibido, i)  ' Comprobamos si el carácter que acabamos de guardar es igual a los caracteres del string(charsPosibles).  ' Iteramos un número de veces igual al tamaño del string usado para comparar los caracteres en el campo.  ' Con esta estructura vemos qué estamos validando  ' Solo si es el tamaño (parámetro con valor 2) se usa la lista charsPosiblesTamano  ' Que solo permite números y punto  If tipo = 2 Then  For j = 1 To charsposiblesTamano.Length  If chRecibido = GetChar(charsposiblesTamano, j) Then  ' Comparamos los caracteres. Si son iguales, se suma 1 al buchón  coincidencias = coincidencias + 1  End If  Next j  ElseIf tipo = 1 Then  ' Si estamos validando nombre:  For j = 1 To charsPosibles.Length  If chRecibido = GetChar(charsPosibles, j) Then  ' Comparamos los caracteres. Si son iguales, se suma 1 al buchón  coincidencias = coincidencias + 1  End If  Next j  ' Si estamos validando categoría:  ElseIf tipo = 0 Then  For j = 1 To charsPosiblesCategoria.Length  If chRecibido = GetChar(charsPosiblesCategoria, j) Then  ' Comparamos los caracteres. Si son iguales, se suma 1 al buchón  coincidencias = coincidencias + 1  End If  Next j  End If  Next i  ' Si el buchón tiene un valor igual al tamaño del string pasado, significa que todos los caracteres  ' son válidos, pues fueron comparados con los caracteres del String y fueron dados por buenos.  If coincidencias = stringRecibido.Length Then  valido = True  End If  End If  Return valido  End Function  'Validacion de ISBN  Public Function ValidarISBN(codigo As String) As Boolean  ' Se opta de nuevo por la solución de poner en un String los valores permitidos.  Dim charsPosibles As String = "0123456789"  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  'Máxima longitud del valor.  valorMaximo = 13  'Solo se permiten códigos de hasta 13 números.  If codigo.Length > valorMaximo Then  MsgBox("Por favor, introduzca un ISBN que sea menor de 13 dígitos", MsgBoxStyle.OkOnly, "Longitud incorrecta.")  End If  For i = 1 To codigo.Length  For j = 1 To charsPosibles.Length  If (GetChar(codigo, i) <> GetChar(charsPosibles, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length <= valorMaximo Then  valido = True  Else  MsgBox("El dato ISBN, solo acepta caracteres numéricos, también debe tener una longitud de 13 digitos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  'Validacion de Titulos libros  Public Function validarTitulolibro(codigo As String) As Boolean  Dim charsPosiblesNombre As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZ1234567890-\_éúíóáÁÉÍÓÚ,()¿?¡!: "  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 100  ' Solo se permiten usuarios de hasta 50 caracteres.  If codigo.Length > valorMaximo Then  MsgBox("Por favor, introduzca un Titulo que sea menor de 100 caracteres.", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")  End If  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length Then  valido = True  Else  MsgBox(" El Titulo no puede contener símbolos no permitidos. Debe tener una longitud máxima de 100 caracteres", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  'Validacion Nombre Autor/es  Public Function validarAutor(codigo As String) As Boolean  ' Si tipo es 1, se realizara la validación de un nombre.  ' Para el nombre.  Dim charsPosiblesNombre As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZáéíóúÁÉÍÓÚ.- "  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 50  ' Solo se permiten nombres de hasta 50 números.  If codigo.Length > valorMaximo Then  MsgBox("Por favor, introduzca un nombre que sea menor de 50 caracteres.", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")  End If  ' Validación nombre  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length Then  valido = True  Else  MsgBox(" El Autor no puede contener caracteres numéricos o símbolos no permitidos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  'Validación Numero de paginas/ Año de edicion /Sotck  Public Function validar4digitos(codigo As String) As Boolean  Dim charsPosiblesNombre As String = "1234567890"  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 4  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length <= valorMaximo Then  valido = True  Else  MsgBox(" Solo se permiten Números. Debe tener una longitud de 4 digitos", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  'Validacion Editorial  Public Function validarEditorial(codigo As String) As Boolean  ' Para el nombre.  Dim charsPosiblesNombre As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZáéíóúÁÉÍÓÚ.123456789 "  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 50  ' Solo se permiten nombres de hasta 50 números.  If codigo.Length > valorMaximo Then  MsgBox("Por favor, introduzca un nombre de editorial que sea menor de 50 caracteres.", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")  End If  ' Validación nombre  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length Then  valido = True  Else  MsgBox(" La editorial no puede contener caracteres numéricos o símbolos no permitidos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  'Validacion Idioma  Public Function validarIdioma(codigo As String) As Boolean  ' Para el nombre.  Dim charsPosiblesNombre As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZáéíóúÁÉÍÓÚ "  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 50  ' Solo se permiten nombres de hasta 50 números.  If codigo.Length > valorMaximo Then  MsgBox("Por favor, introduzca un Idioma que sea menor de 50 caracteres.", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")  End If  ' Validación nombre  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length Then  valido = True  Else  MsgBox(" El Idioma no puede contener caracteres numéricos o símbolos no permitidos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  'Validacion Plaza de Edicion  Public Function validarPlazaEdicion(codigo As String, tipo As Integer) As Boolean  ' Si tipo es 1, se realizara la validación de un nombre.  ' Si tipo es 2, se realizara la validación de un apellido.  ' Para el nombre.  Dim charsPosiblesNombre As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZáéíóúÁÉÍÓÚ- "  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 20  ' Solo se permiten nombres de hasta 50 números.  If codigo.Length > valorMaximo Then  MsgBox("Por favor, introduzca un nombre que sea menor de 20 caracteres.", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")  End If  ' Validación nombre  If tipo = 1 Then  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  End If  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length Then  valido = True  Else  MsgBox(" La plaza de edicion no puede contener caracteres numéricos o símbolos no permitidos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  'Validacion Traductor  Public Function validarTraductor(codigo As String) As Boolean  ' Para el nombre.  Dim charsPosiblesNombre As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZáéíóúÁÉÍÓÚ.- "  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 50  ' Solo se permiten nombres de hasta 50 números.  If codigo.Length > valorMaximo Then  MsgBox("Por favor, introduzca un nombre que sea menor de 50 caracteres.", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")  End If  ' Validación nombre  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length Then  valido = True  Else  MsgBox(" El Tradcutor no puede contener caracteres numéricos o símbolos no permitidos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  'Validar Precio  Public Function validarPrecio(codigo As String) As Boolean  ' Se opta de nuevo por la solución de poner en un String los valores permitidos.  Dim charsPosibles As String = "0123456789,"  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 6  ' Solo se permiten códigos de hasta 6 números.  'If codigo.Length > valorMaximo Then  'MsgBox("Por favor, introduzca un número de teléfono que sea menor de 6 dígitos", MsgBoxStyle.OkOnly, "Longitud incorrecta.")  'End If  For i = 1 To codigo.Length  For j = 1 To charsPosibles.Length  If (GetChar(codigo, i) <> GetChar(charsPosibles, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length And codigo.Length <= valorMaximo Then  valido = True  Else  MsgBox(" El precio, solo admite caracteres numéricos y coma ',' .También debe tener una longitud de 6 digitos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  'Validar Nombre productos  Public Function validarNombreProducto(codigo As String) As Boolean  Dim charsPosiblesNombre As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZ1234567890-\_éúíóá¿?¡!();,.& "  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 50  ' Solo se permiten nombres de hasta 50 números.  If codigo.Length > valorMaximo Then  MsgBox("Por favor, introduzca una Nombre que sea menor de 50 caracteres.", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")  End If  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length Then  valido = True  Else  MsgBox(" El Nombre del producto no puede contener símbolos no permitidos. Debe tener una longitud máxima de 50 caracteres", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  'Validar Categoría Y Genero  Public Function validarCategoria(codigo As String) As Boolean  ' Para el nombre.  Dim charsPosiblesNombre As String = "abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZáéíóúÁÉÍÓÚ- "  Dim valido As Boolean = False  Dim coincidencias As Integer = 0  Dim valorMaximo As Integer  ' Máxima longitud del valor.  valorMaximo = 50  ' Solo se permiten nombres de hasta 50 números.  If codigo.Length > valorMaximo Then  MsgBox("Por favor, introduzca una Categoria que sea menor de 50 caracteres.", MsgBoxStyle.OkOnly + MsgBoxStyle.Exclamation, "Longitud incorrecta.")  End If  ' Validación nombre  For i = 1 To codigo.Length  For j = 1 To charsPosiblesNombre.Length  If (GetChar(codigo, i) <> GetChar(charsPosiblesNombre, j)) Then  Else  ' Si se encuentran el mismo número de coincidencias que caracteres contrastados hay  ' entonces los datos se validan.  coincidencias = coincidencias + 1  End If  Next j  Next i  ' Devolvemos si los datos son válidos si se han encontrado tantas coincidencias como caracteres tenga la cadena.  If coincidencias = codigo.Length Then  valido = True  Else  MsgBox(" La Categoria no puede contener caracteres numéricos o símbolos no permitidos.", MsgBoxStyle.OkOnly, "Error - Caracteres incorrectos")  End If  Return valido  End Function  End Class |