

TÍTULO

DESARROLLO DE APLICACIONES I

Sesión Nro. 02

Objetivos de la sesión:

Al culminar la presente podrás:

- Conocer el manejo de los formularios y de los controles básicos así como pautas para una mejor presentación de los mismos
- Manejar los recursos gráficos de un proyecto.
- Manejar eventos y delegarlos a otros controles.
- Manejar controles Listbox y ComboBox
- Manejar controles PictureBox, DateTimePicker y MaskedTextBox
- Manejar los errores en tiempo de ejecución (excepciones)



Temario

1. Manejo de Formularios Windows
2. Controles Básicos
3. Los procedimientos de evento
4. Manejo de MaskedTextBox, PictureBox y DateTimePicker
5. Manejo de Excepciones



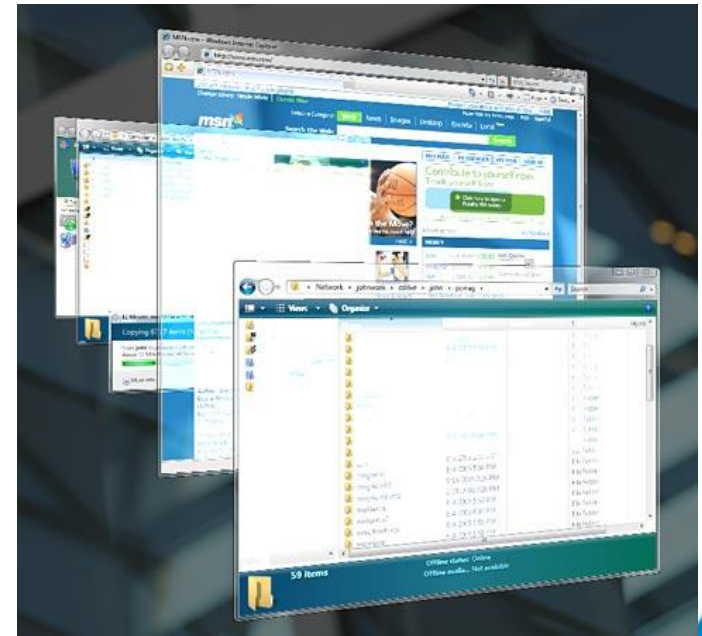
Tema Nro. 1:

Manejo de Formularios Windows



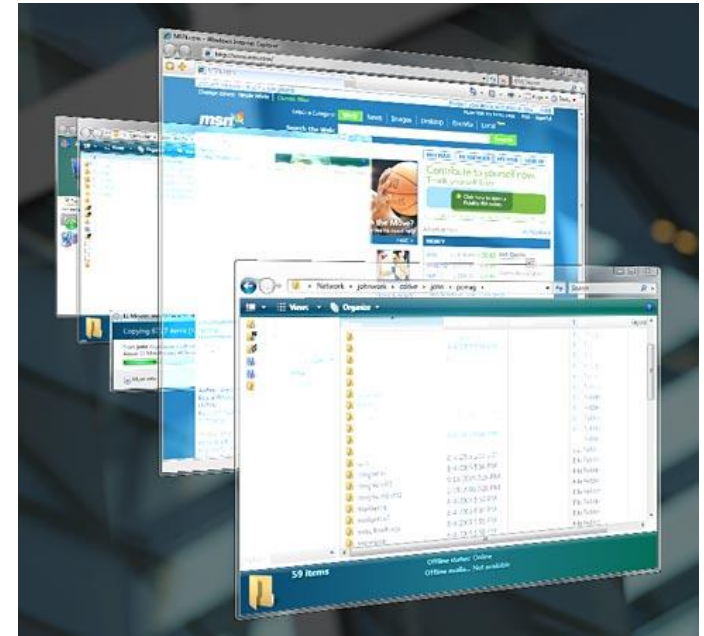
El espacio de nombres System.Windows.Form

Este espacio de nombres contiene todos los tipos del entorno, a través de los cuales podremos desarrollar aplicaciones compuestas por formularios Windows, junto a los correspondientes controles que permiten al usuario la interacción con el programa.



La clase Form

Esta clase contiene todos los miembros para la creación y manipulación de formularios. Tras instanciar un objeto de la clase Form, mediante la configuración de las adecuadas propiedades, podemos crear formularios estándar, de diálogo, de interfaz múltiple o MDI, con diferentes bordes, etc.





Principales propiedades

BackColor	Color de fondo del formulario.
Text	Texto en la barra de título del formulario.
ControlBox	True/False. Determina si tiene o no el cuadro de control.
Enabled	True/False. Determina si está habilitado para responder a las acciones del usuario.
FormBorderStyle	Define el estilo de los bordes del formulario
Icon	Icono que se muestra cuando el formulario está minimizado.
IsMdiContainer	True/False Que establece si el formulario será o no contenedor.
KeyPreview	True/False Que indica si el formulario será “sensible” a eventos de teclado de su colección de controles
Location	Ubicación del formulario.
MaximizeBox	True/False. Determina si tiene o no el botón <i>Maximizar</i> .
MinimizeBox	True/False. Determina si tiene o no el botón <i>Minimizar</i> .
Name	Nombre del formulario.
Text	Título del formulario
StartPosition	Establece la posición inicial del formulario
WindowState	Estado inicial del formulario (normal, maximizado o minimizado)



Principales Eventos

Activated	Ocurre cuando el formulario se convierte en la ventana activa.
Click	Ocurre cuando hace clic sobre el formulario.
Deactivated	Ocurre cuando el formulario deja de ser la ventana activa.
FormClosing	Ocurre cuando se esta cerrando el formulario.
FormClosed	Ocurre cuando el formulario ya se cerró.
Load	Ocurre cuando se carga un formulario.
KeyUp/ KeyDown /KeyPress	Ocurre cuando se pulsa una tecla sobre cualquiera de los controles de edición de un formulario, siempre y cuando su propiedad KeyPreview este en True





Principales Métodos

Show : Carga y muestra el formulario (no modal)

Ejemplo: `frm01.Show();`

ShowDialog :Carga y muestra el formulario (modal)

Ejemplo: `frm02.ShowDialog();`

Hide : Oculta el formulario

Ejemplo: `frm01.Hide();`

Close: Cierra un formulario

Ejemplo: `frm02.Close ();`

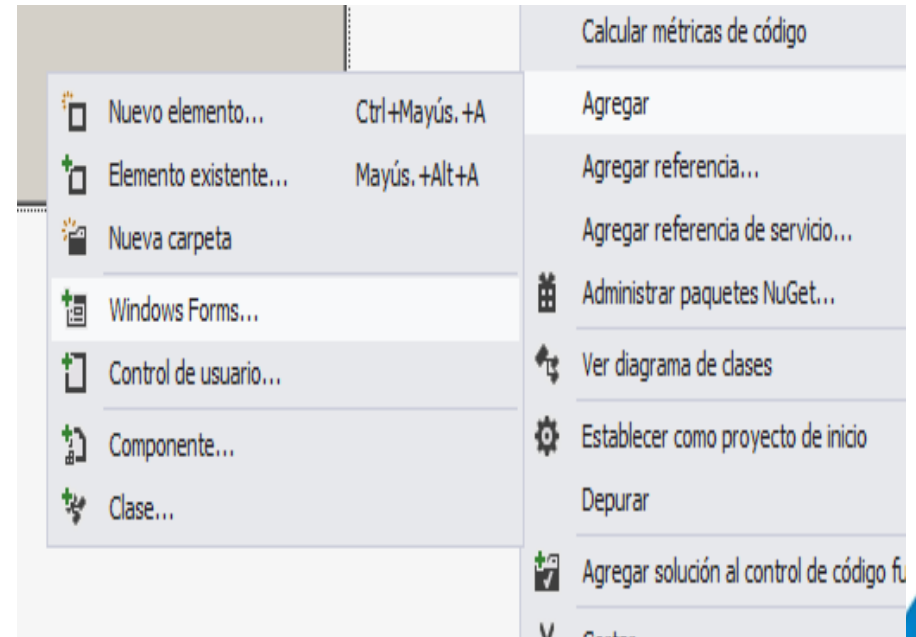
Recuerde que puede hacer mención al formulario activo con la palabra “this”. Por ejemplo la siguiente instrucción cerraría el formulario actual -> `this.Close();`

Los formularios Windows

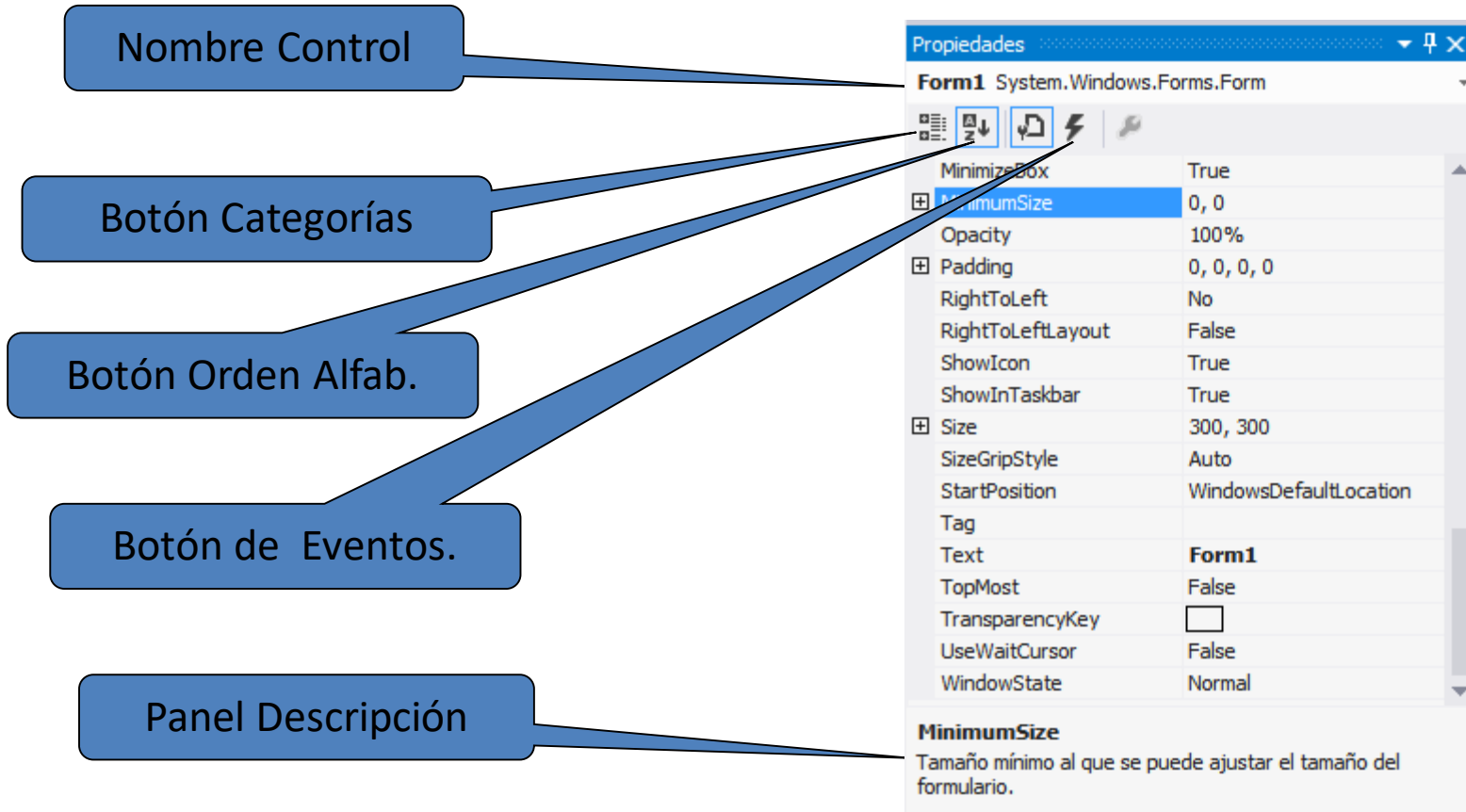


Manejo de formularios con el Explorador de proyectos

Se pueden adicionar, eliminar , excluir formularios (así como otros objetos como módulos, clases, informes, conjuntos de datos, etc.) desde la ventana del explorador de proyectos.



La Ventana de Propiedades



The diagram illustrates the 'Propiedades' (Properties) window in a development environment. Callouts point to the following components:

- Nombre Control:** Points to the title bar of the 'Propiedades' window.
- Botón Categorías:** Points to the 'Categorías' (Categories) button in the toolbar.
- Botón Orden Alfab.:** Points to the 'Orden Alfab.' (Alphabetical Order) button in the toolbar.
- Botón de Eventos:** Points to the 'Eventos' (Events) button in the toolbar.
- Panel Descripción:** Points to the description panel at the bottom of the window.

The 'Propiedades' window displays the properties for 'Form1' (System.Windows.Forms.Form). The properties are organized into expandable sections:

- Form1 System.Windows.Forms.Form**
 - MinimizeBox: True
 - MaximumSize: 0, 0
 - Opacity: 100%
 - Padding: 0, 0, 0, 0
 - RightToLeft: No
 - RightToLeftLayout: False
 - ShowIcon: True
 - ShowInTaskbar: True
 - Size: 300, 300
 - SizeGripStyle: Auto
 - StartPosition: WindowsDefaultLocation
 - Tag:
 - Text: **Form1**
 - TopMost: False
 - TransparencyKey: ☐
 - UseWaitCursor: False
 - WindowState: Normal
- MinimumSize**
 - Tamaño mínimo al que se puede ajustar el tamaño del formulario.

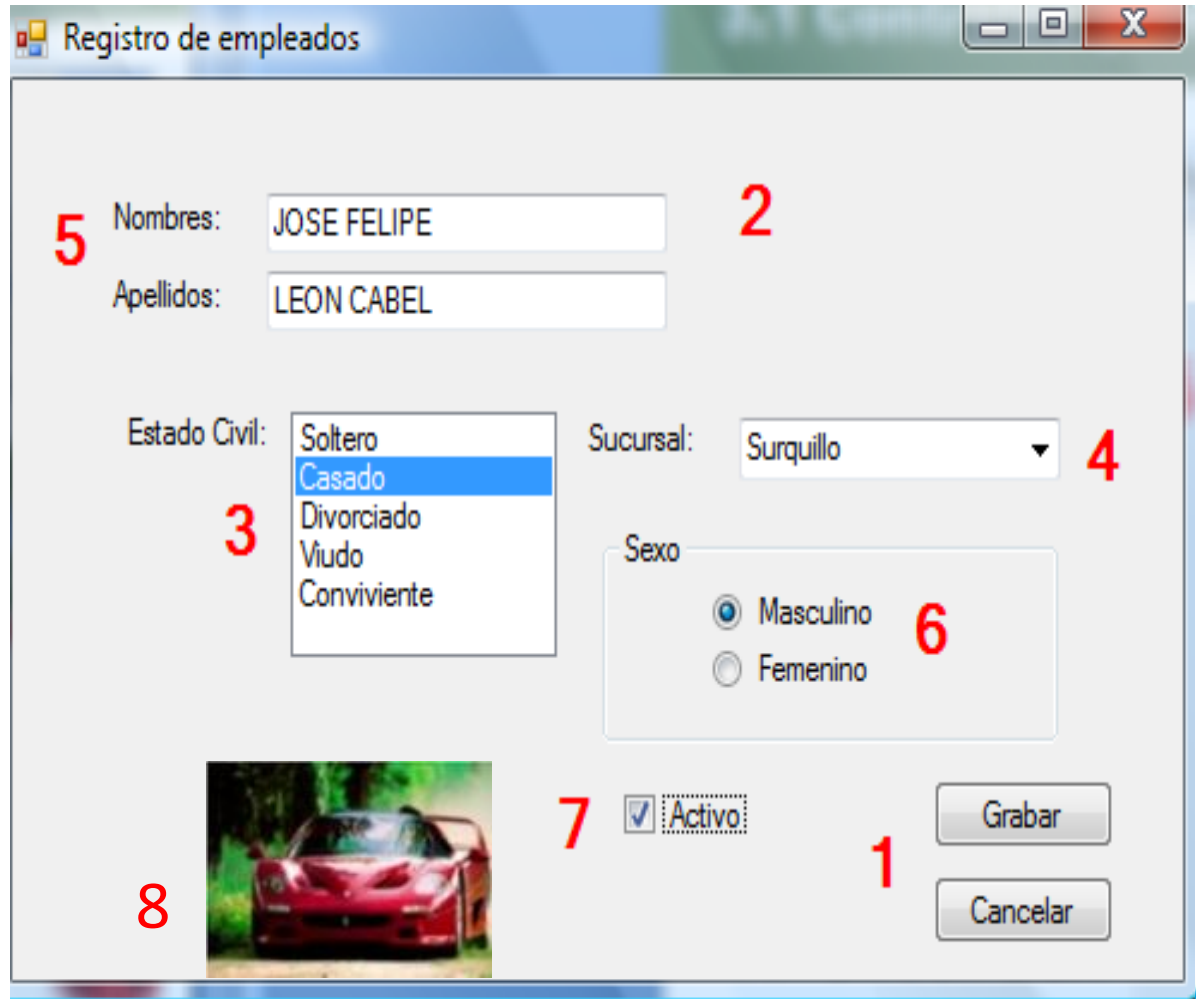


Tema Nro. 2: Controles de Formularios Windows



Controles Básicos

1. Button
2. TextBox
3. ListBox
4. ComboBox
5. Label
6. Radio Button
7. Check Box
8. PictureBox

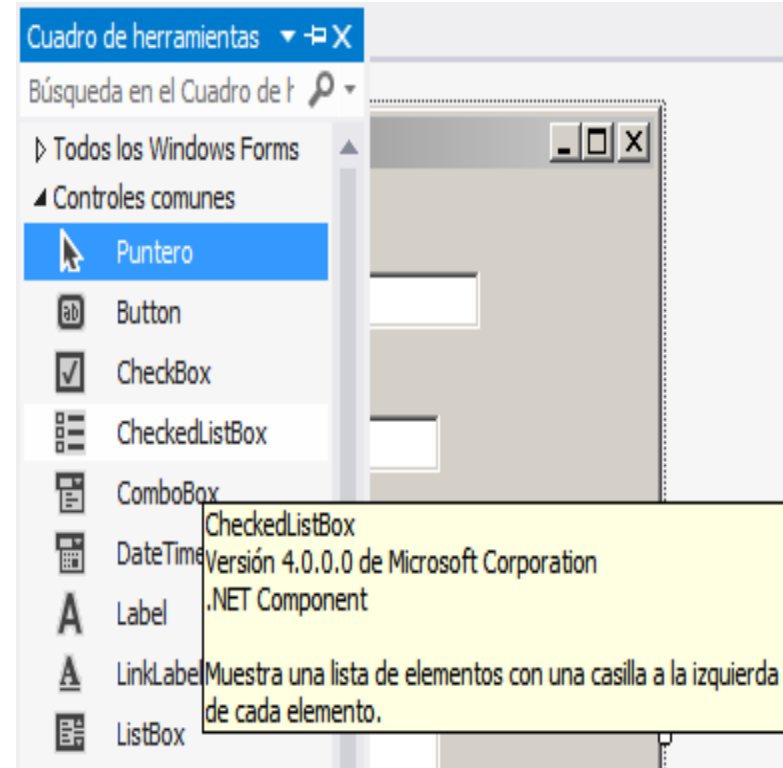
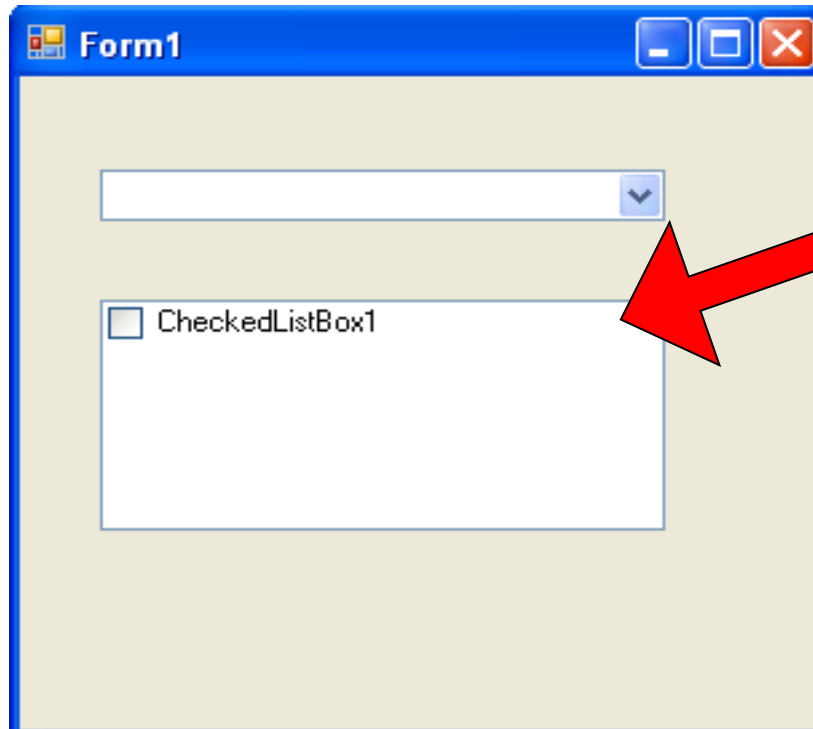


The screenshot shows a Windows application window titled "Registro de empleados". The window contains the following controls and annotations:

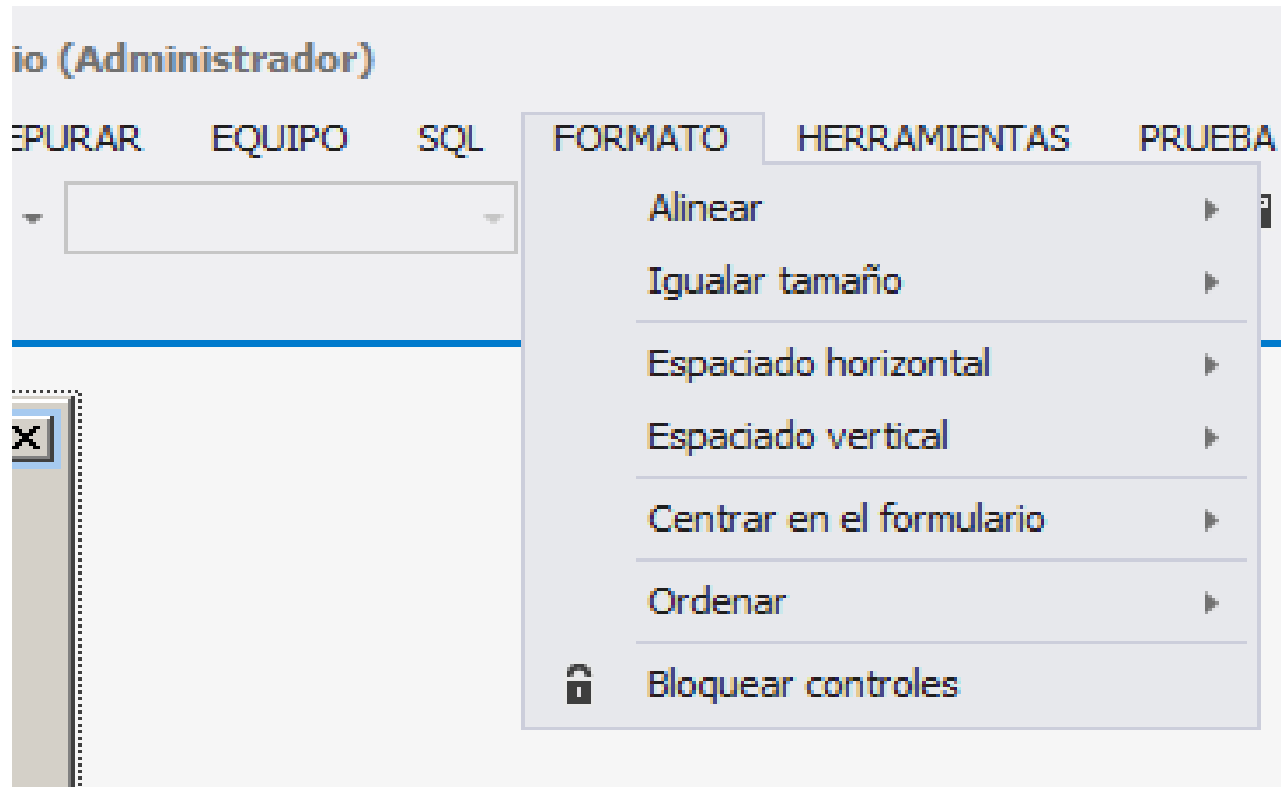
- 5**: A label "Nombres:" followed by a text box containing "JOSE FELIPE".
- 2**: A label "Apellidos:" followed by a text box containing "LEON CABEL".
- 3**: A label "Estado Civil:" followed by a list box containing "Soltero", "Casado" (selected), "Divorciado", "Viudo", and "Conviviente".
- 4**: A label "Sucursal:" followed by a dropdown menu showing "Surquillo".
- 6**: A label "Sexo:" followed by two radio buttons: "Masculino" (selected) and "Femenino".
- 7**: A label "Activo:" followed by a checked checkbox.
- 8**: A picture box showing a red sports car.
- 1**: Two buttons at the bottom right: "Grabar" and "Cancelar".



Agregando controles al formulario:



Organizar controles en el formulario



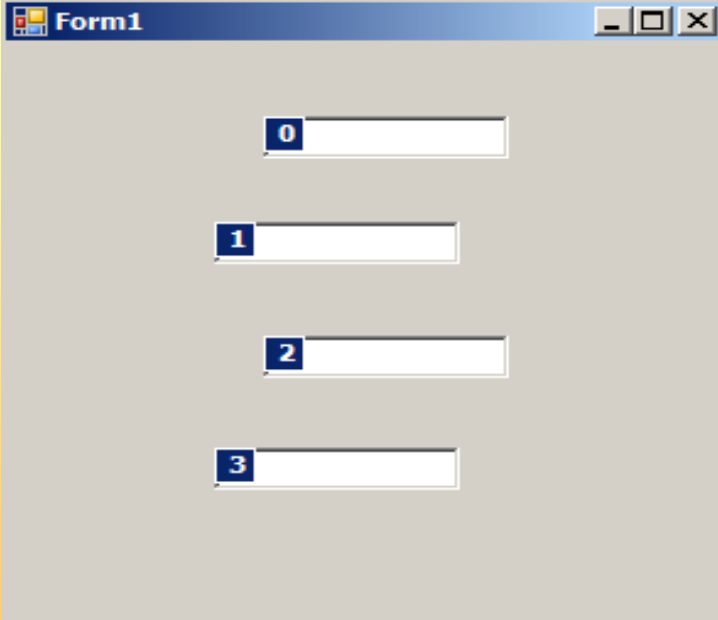
Estableciendo el Orden de Tabulación

- **Para establecer el orden de tabulación de los controles**

- En el menú **Ver**, seleccionar **Orden de tabulación**
- Hacer clic en un control para cambiar su orden de tabulación

-- O --

- Establecer la propiedad **TabIndex**
- Configurar la propiedad **TabStop** como **True**



The screenshot shows a Windows application window titled "Form1". Inside the window, there are four text input boxes stacked vertically. To the left of each text box is a small blue square containing a white number, representing the TabIndex property. The numbers are 0, 1, 2, and 3, respectively, from top to bottom. This visualizes the tab order of the controls in the form.



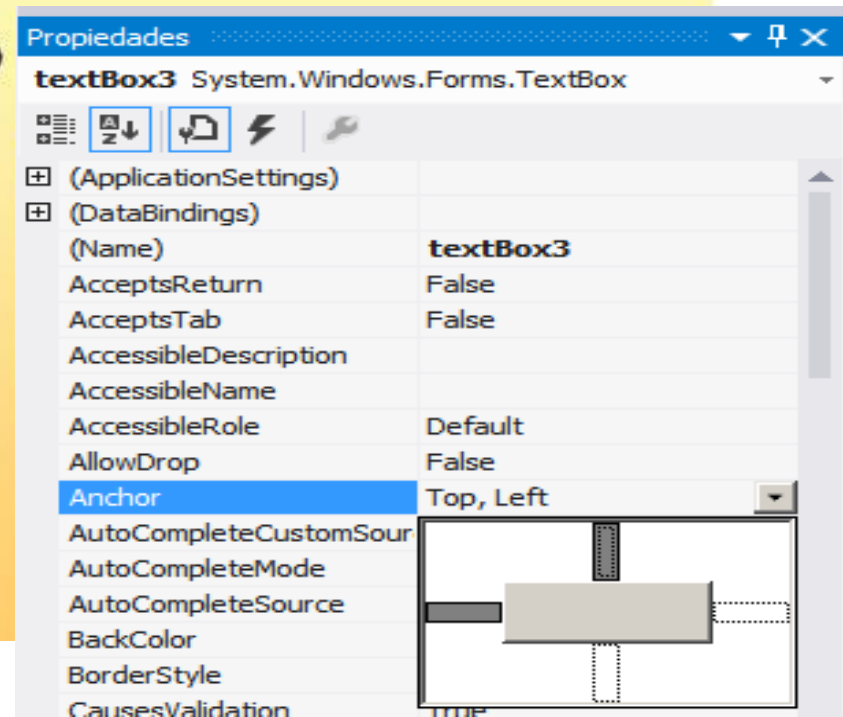
Delimitando un control

■ Delimitar

- Garantiza que los bordes del control permanecen en la misma posición respecto al contenedor principal

■ Delimitar un control al formulario

- Establecer su propiedad **Anchor**
- Valor predeterminado: **Superior, Izquierda**



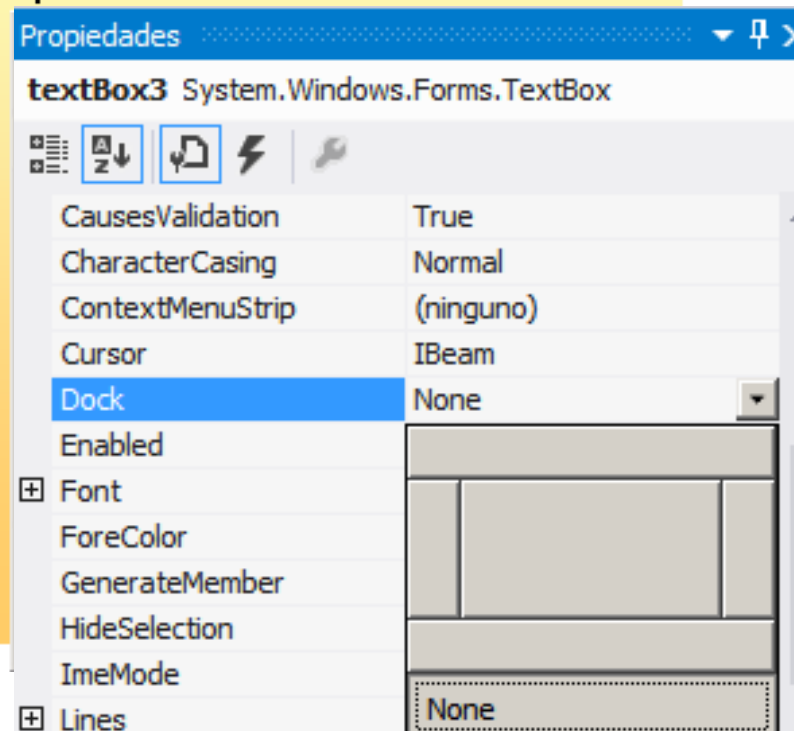
Acoplando un control

■ Acoplar

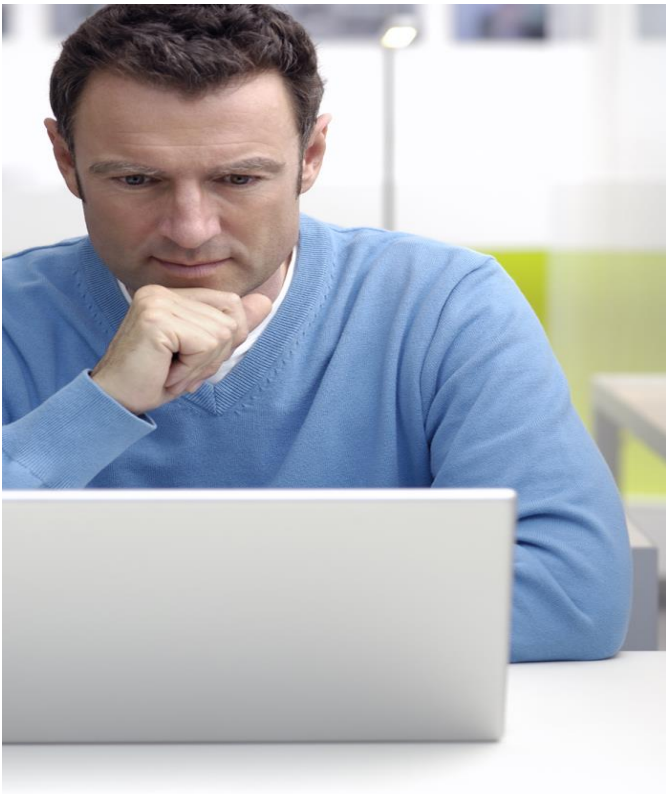
- Permite pegar los bordes de un control a los bordes de su control principal

■ Acoplar un control

- Establecer la propiedad **Dock**



Demostración 1: Manejo de propiedades Anchor, Dock y TabIndex.



- En esta demostración, aprenderemos a manejar las propiedades de formato y ubicación mas importantes de los controles.



Tema Nro. 3: Los procedimientos de evento



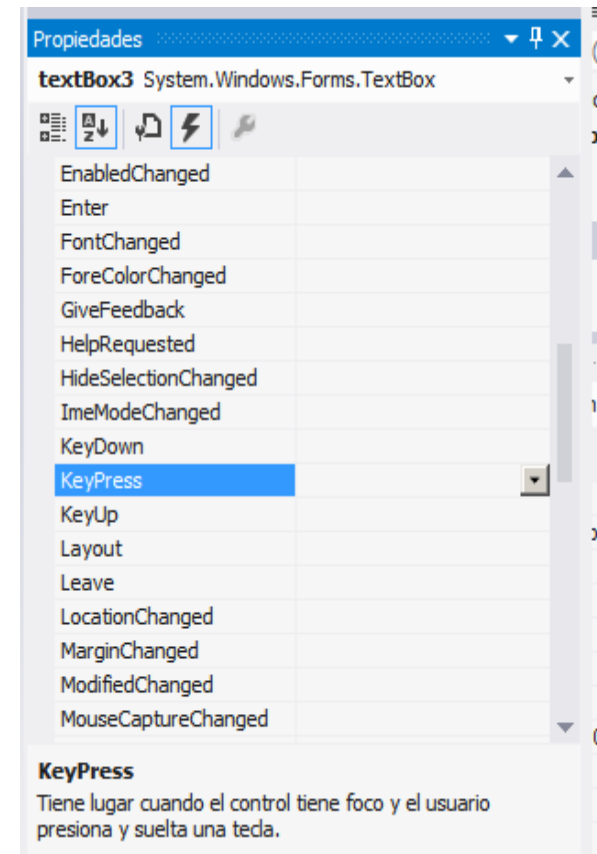
¿Qué son los procedimientos de evento?

- Cuando se trabaja con interfaces graficas , la programación en la capa de presentación obedece o se gestiona en base a los eventos que se dan por la interacción del usuario y la interface grafica.
- Por ejemplo el hacer un clic en un botón , o pulsar una tecla o cargar un formulario, son ejemplos de estas interacciones llamadas también eventos .
- Es importante establecer que debe realizar el programa ante el suceso de cada evento. Y eso se logra codificar en los llamados procedimientos de evento



¿Qué son los procedimientos de evento?

- Cada control tiene su evento por defecto . Por ejemplo un botón tiene como evento por defecto el clic (Click), o una caja de texto el evento de escritura (TextChanged).
- Si queremos codificar el evento por defecto del control bastara con hacer doble clic sobre el y ya habremos generado en la ventana de código dicho procedimiento.
- Si se desea programar otro evento podremos hacerlo seleccionando el evento desde la ventana de propiedades del control (vista Eventos) y haciendo doble clic sobre el evento a programar



¿Qué son los procedimientos de evento?

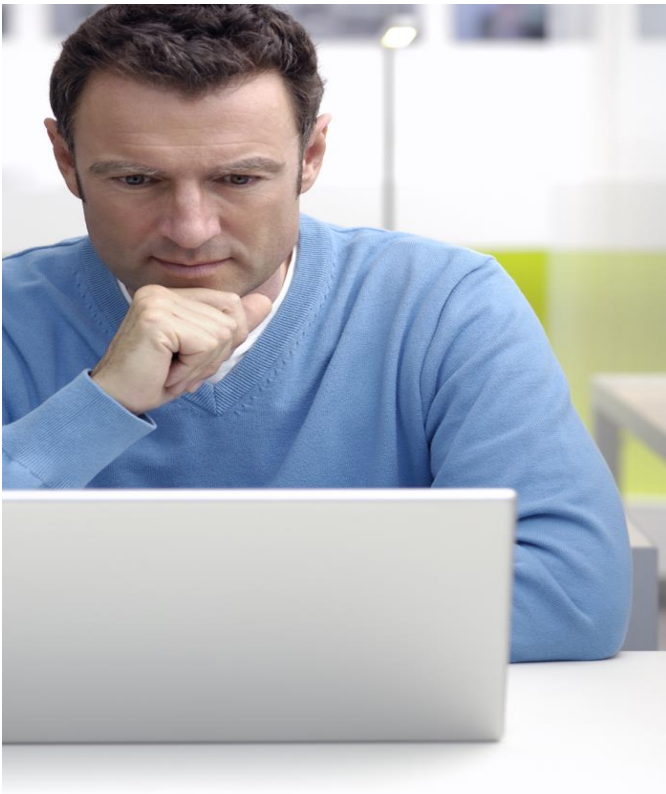
- Cada procedimiento de evento recibe un nombre basado en el control y el evento a programar. Así mismo se manejan 2 parámetros :
 - Sender : Que es el control que genera el evento
 - e : Que es un parámetro que permite manipular el evento dado de ser necesario.

```
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show("Bienvenidos al C#.....fui enviado por :" + sender, "Mensaje",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
}

private void textBox1_KeyPress(object sender, KeyPressEventArgs e)
{
    if( e.KeyChar == '\r' )
    {
        MessageBox.Show("El texto tipeado es : " + textBox1.Text);
    }
}
```



Demostración 2: Validación de Cajas de Texto




- En esta demostración, aprenderemos a validar cajas de texto para evitar se ingresen datos no deseados en base a las propiedades `MaxLength` y `CharacterCasing` así como el evento `KeyPress`. Así mismo conoceremos como podemos delegar eventos entre controles.



Tema 4: Otros Controles de Importancia

1. [DateTimePicker](#)
2. [MaskedTextBox](#)
3. [PictureBox](#)



The screenshot shows a Windows application window titled "Uso de Formularios". It contains two tabs: "Datos Generales" (selected) and "Datos Opcionales". The "Datos Generales" tab is divided into two sections: "Datos del Usuario" and "Datos del Perfil".

Datos del Usuario:

- Nombre:** Text box containing "Liliana Nataly".
- Apellido P:** Text box containing "Arcila".
- Apellido M:** Text box containing "Díaz".
- Sexo:** Radio buttons for "Masculino" and "Femenino". "Femenino" is selected.
- Nacionalidad:** Dropdown menu showing "Perú".

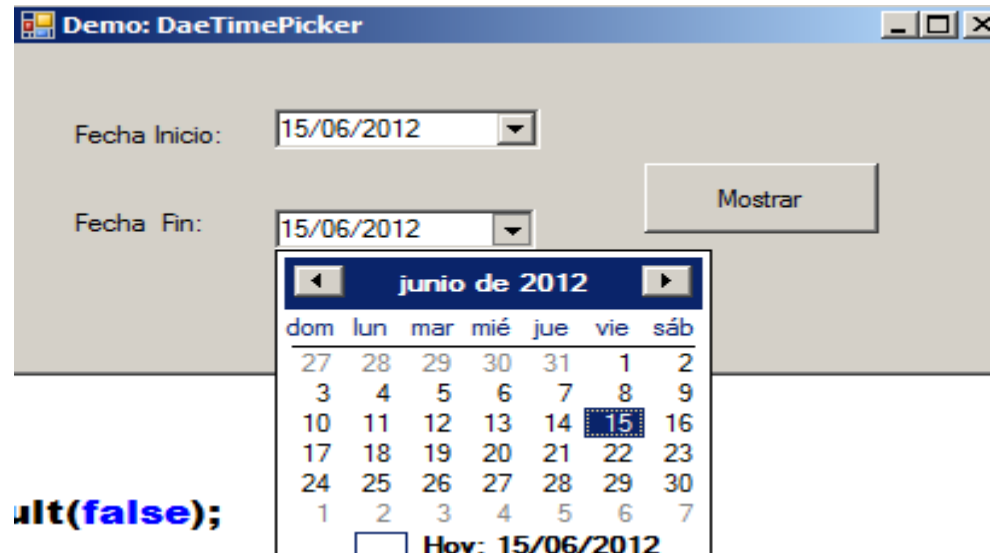
Datos del Perfil:

- A placeholder image for a user profile picture.
- Fecha Nacimiento:** Date picker showing "23/02/1993".
- Desea Mostrar estos datos:** Check box, which is checked.



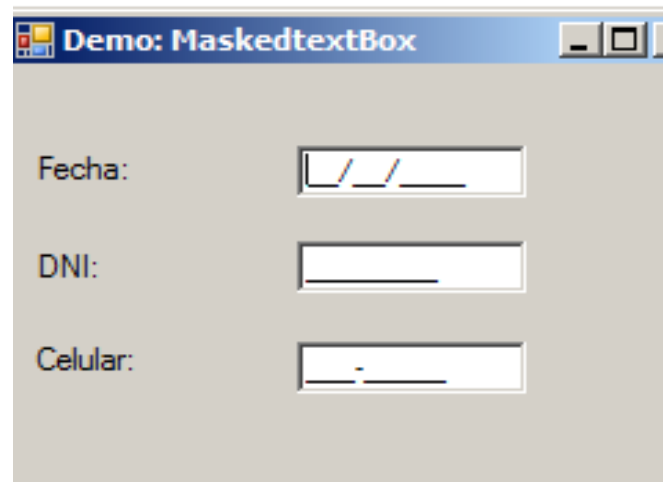
Control DateTimePicker

- Este control permite el ingreso de fechas a través de una especie de combobox donde se despliega un calendario y se selecciona la fecha deseada. Su ventaja radica en que ya no es necesario hacer ninguna validación sobre la fecha ingresada.
- Para determinar su formato , se establece su propiedad Format , que en la mayoría de casos se establece como ShortDate (fecha corta)



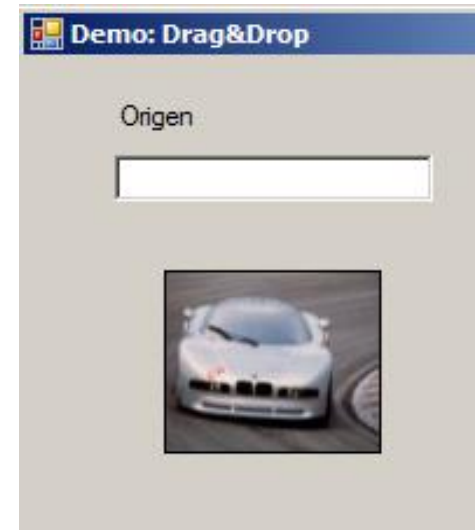
Control MaskedTextBox

- Este control permite ingresar textos con una mascara o formato a un TextBox , de ahí su denominación de MaskedTextBox.
- Sirve para ingresar datos que requieren de un formato ya conocido (fechas, teléfonos, etc.) como también de formatos personalizados.
- La propiedad Mask permite establecer la mascara de ingreso de datos.

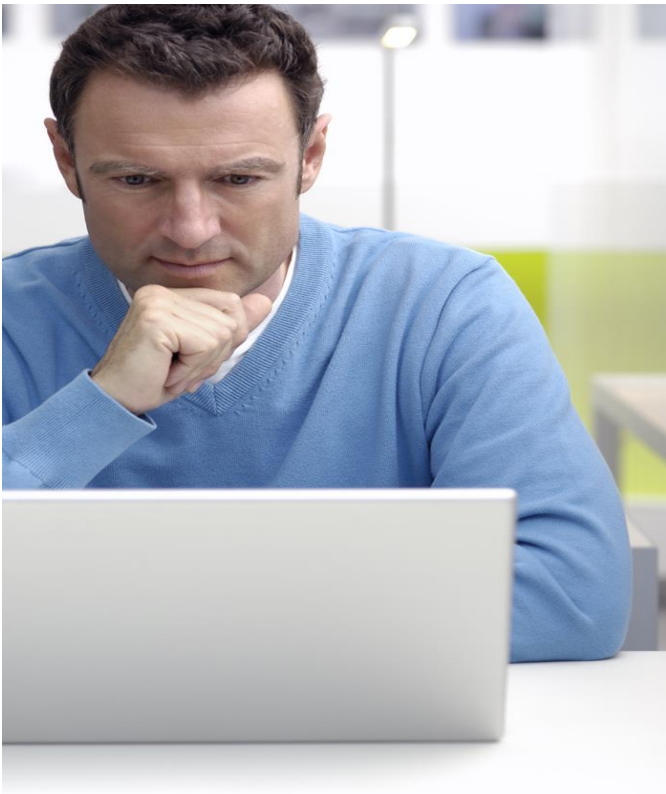


Control PictureBox

- Este control permite almacenar imágenes (jpg,ico,avi,etc) y se emplea mucho para el tema de mostrar fotos o imágenes con animación dentro de un formulario.
- Para asignar la imagen establezca la propiedad Image y de preferencia importe una imagen como recurso del proyecto y asignarla al control PictureBox
- Para mejorar su presentación puede emplear las propiedades BorderStyle (formato del borde) y SizeMode (asignándole el valor StretchImage , si desea que la imagen cope el tamaño del control, independiente del tamaño de dicha imagen).



Demostración 3: Manejo de controles DateTimePicker, MaskedTextBox y PictureBox



- En esta demostración, aprenderemos a manejar fechas y formatos de entrada en función a los controles DateTimePicker y MaskedTextBox, así como el incrustar imágenes en PictureBox.

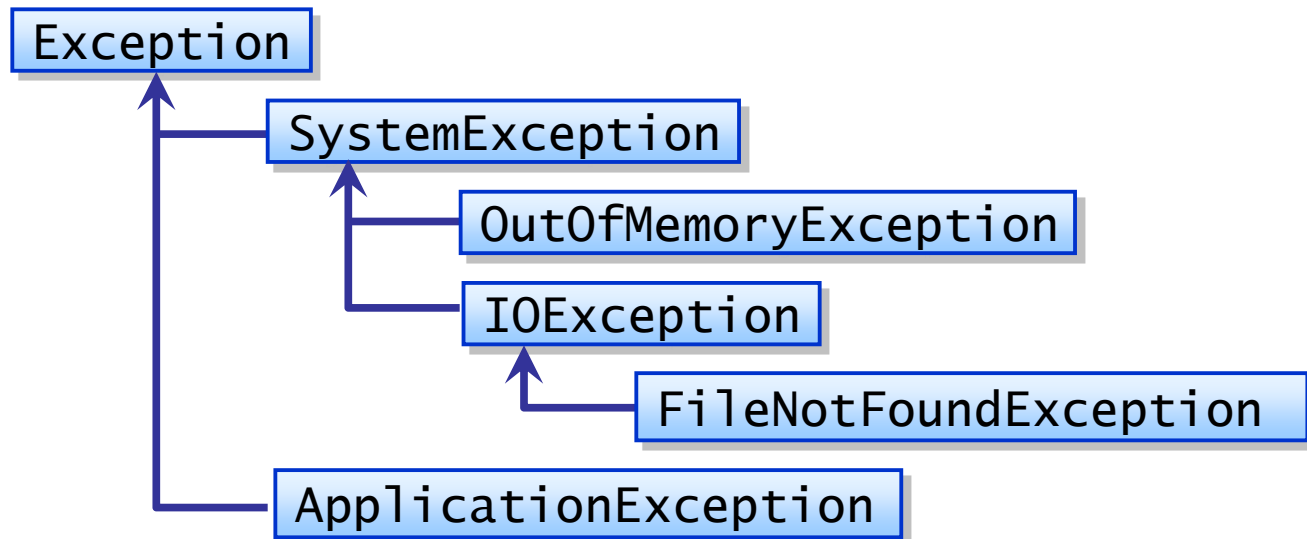


Tema Nro. 5: Manejo estructurado de excepciones



La clase Exception

- El .NET Framework proporciona el siguiente modelo de objeto de excepciones:



- Las clases Exception permiten recuperar información sobre cualquier excepción que encontremos
- Las propiedades de la clase base Exception permiten analizar excepciones
 - Principales propiedades: StackTrace, Message, HelpLink, Source



¿Qué es la gestión estructurada de excepciones?

- Detecta y responde a errores mientras se ejecuta una aplicación
- Utilice try...catch...finally para encapsular y proteger bloques de código que podrían provocar errores
 - Cada bloque tiene uno o más controladores asociados
 - Cada controlador especifica alguna forma de condición de filtro en el tipo de excepción que controla
- Ventajas:
 - Permite la separación entre la lógica y el código de gestión de errores
 - Facilita la lectura, depuración y mantenimiento del código



Cómo utilizar la instrucción try...catch

- Poner el código que podría lanzar excepciones en un bloque Try
- Gestionar las excepciones en otro bloque Catch

```
try
{
    cmd.ExecuteNonQuery();
}
catch (SQLException ex1)
{
    MessageBox.Show("Error :" + ex1.Message);
}
catch (Exception ex2)
{
    MessageBox.Show("Error :" + ex2.Message);
}
```

Lógica de programa

Gestión de excepciones



Cómo utilizar el bloque Finally

- Sección opcional; si se incluye, se ejecuta siempre
- Colocar código de limpieza, como el utilizado para cerrar archivos, en el bloque finally

```
try
{
    cmd.ExecuteNonQuery();
}
catch (SQLException ex1)
{
    MessageBox.Show("Error :“ + ex1.Message);
}
finally
{
    cmd.Parameters.Clear();
}
```



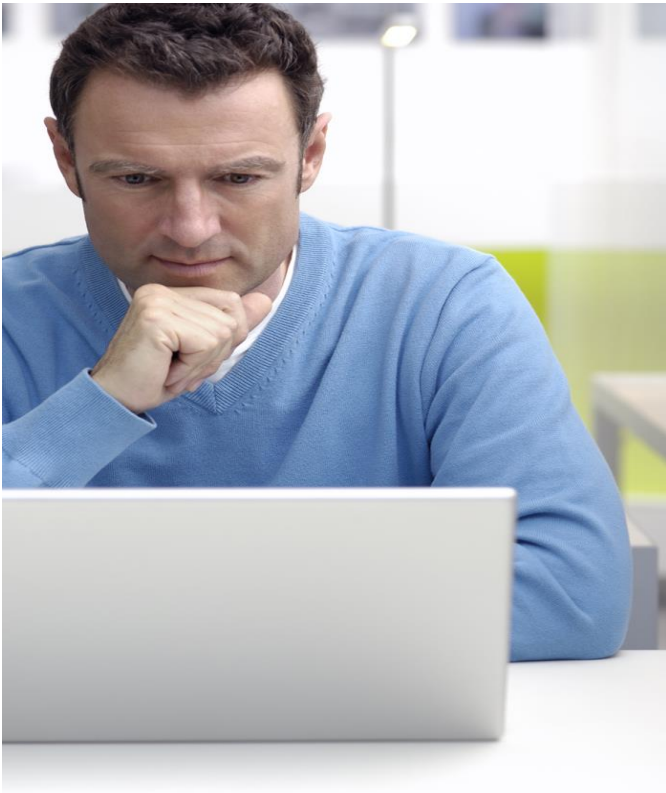
Cómo lanzar excepciones

- Utilizar la instrucción `throw` para crear una excepción que podamos controlar con el código de la gestión estructurada de excepciones.
- Se emplea en los casos donde se rompa una regla de negocio y queramos manipularlo como una excepción .

```
if (.....)
{
    throw new Exception(«Upsss error!!!» );
}
```



Demostración 3: Manejo de ListBox ComboBox y Manejo de Excepciones



- En esta demostración, aprenderemos a manejar la colección Items de un ListBox y de ComboBox , reforzando también el manejo estructurado de excepciones



Conclusiones de la sesión:

- El empleo de los controles correctos para cada caso garantiza un fácil manejo de la aplicación, obteniendo una capa de presentación amigable.
- Al saber manejar las excepciones podremos evitar que las aplicaciones de corten de manera abrupta y sin un sustento de cara al usuario..
- Manejar correctamente las reglas de negocio , aplicando los procesos correspondientes dentro de una especificación funcional nos ayuda como desarrolladores en la implementación de una solución informática.



Visual
Studio

