



## Preguntas detonadoras



- ¿Qué es el framework .NET? ¿Cómo está organizado? ¿Para qué sirve?
- ¿Cuál es la herramienta de software necesaria para implementar aplicaciones en la plataforma .NET?
- ¿Representa lo mismo la programación visual que la programación orientada a objetos?
- ¿Se puede programar orientado a objetos en modo consola?
- ¿Cuáles son los controles visuales más comunes en una aplicación visual? ¿Cómo se utilizan?
- ¿Por qué es importante utilizar nomenclaturas estándar para identificar los componentes de una aplicación?
- ¿Cuáles son las recomendaciones de nomenclatura para identificar los componentes de una aplicación?

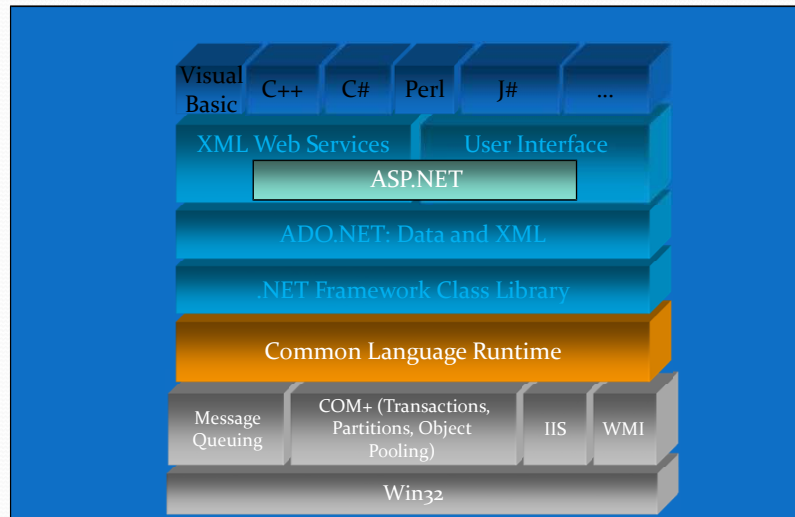
3

## Introducción a la programación en C# .NET

- ◆ **Introducción a la plataforma Microsoft .NET**
  - ◆ El framework .NET y sus componentes
  - ◆ Principales clases del framework .NET
- ◆ **Microsoft Visual Studio .NET**
  - ◆ Compilación y ejecución de programas
  - ◆ El debugger
  - ◆ Breakpoints
  - ◆ Watches
  - ◆ Ejecutar paso a paso
  - ◆ Examinar variables en tiempo de ejecución

4

## Introducción a la plataforma Microsoft .Net

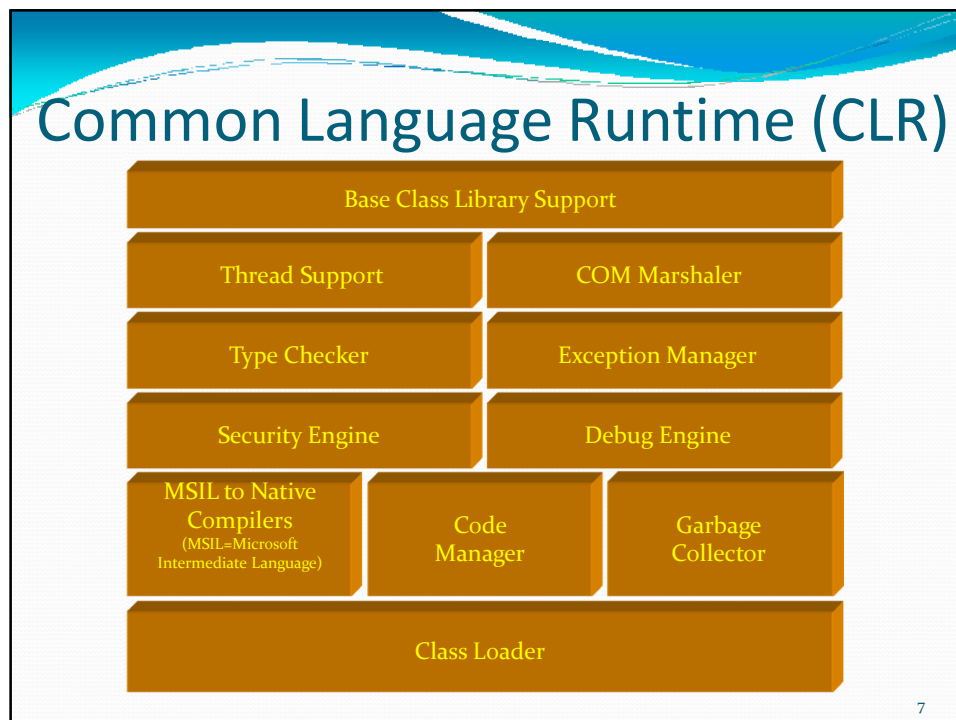


5

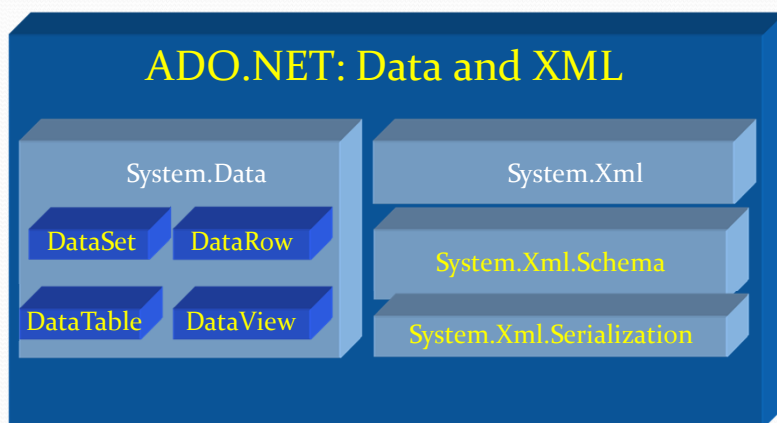
## Componentes del framework .Net

- ◆ **Common Language Runtime**
- ◆ Librerías de clases del Framework .NET
- ◆ **ADO.NET: Datos and XML**
- ◆ Formas Web y Servicios Web XML
- ◆ **Interfaces para Windows**

6

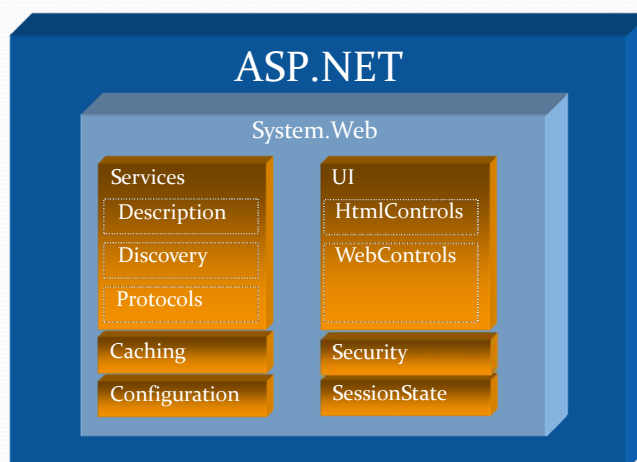


## ADO .NET: Datos y XML



9

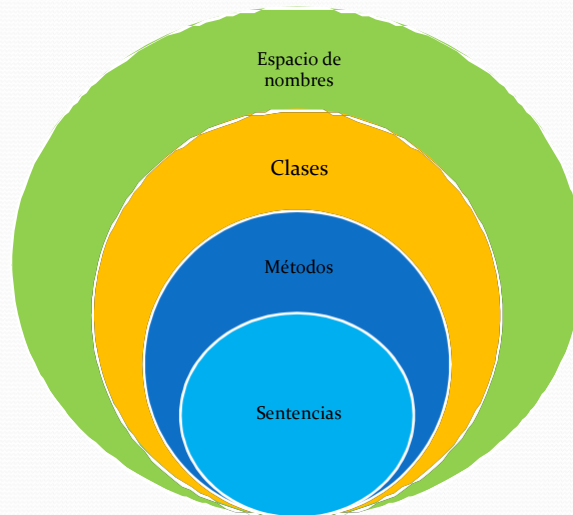
## Formas web y servicios web XML



10



## Estructura jerárquica de una aplicación en C#



11

## Proyecto de consola en C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Ejercicio1 // Espacio de nombres (nombre del proyecto)
{
    class Program // Clase del programa
    {
        static void Main(string[] args) // Método principal
        {
        }
    }
}
```

12

## Proyecto de formas en C#

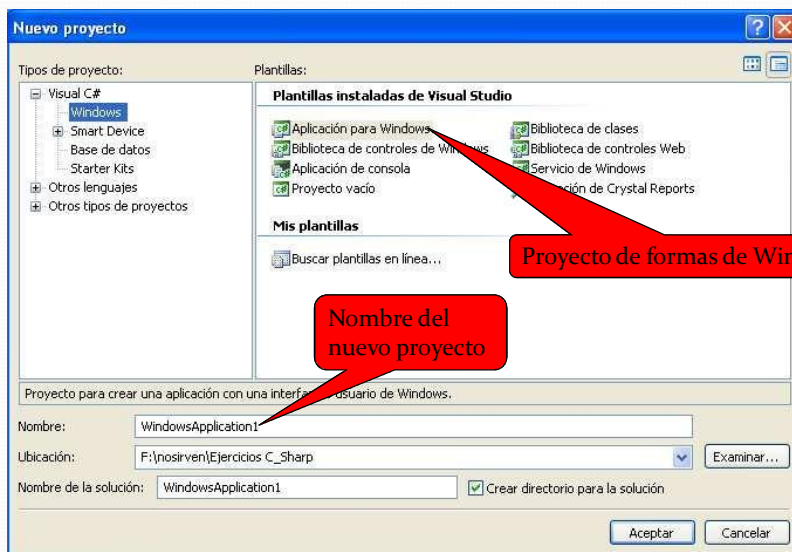
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Ejercicio2 // Espacio de nombres (nombre del proyecto)
{
    public partial class Form1 : Form // Clase
    {
        public Form1()
        {
            InitializeComponent();
        }

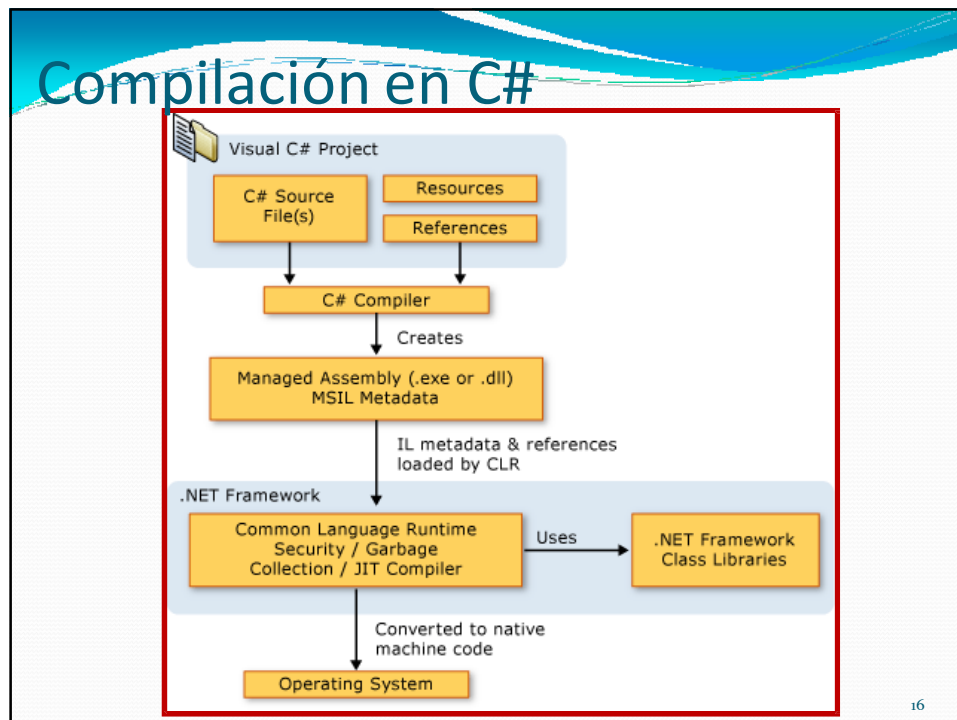
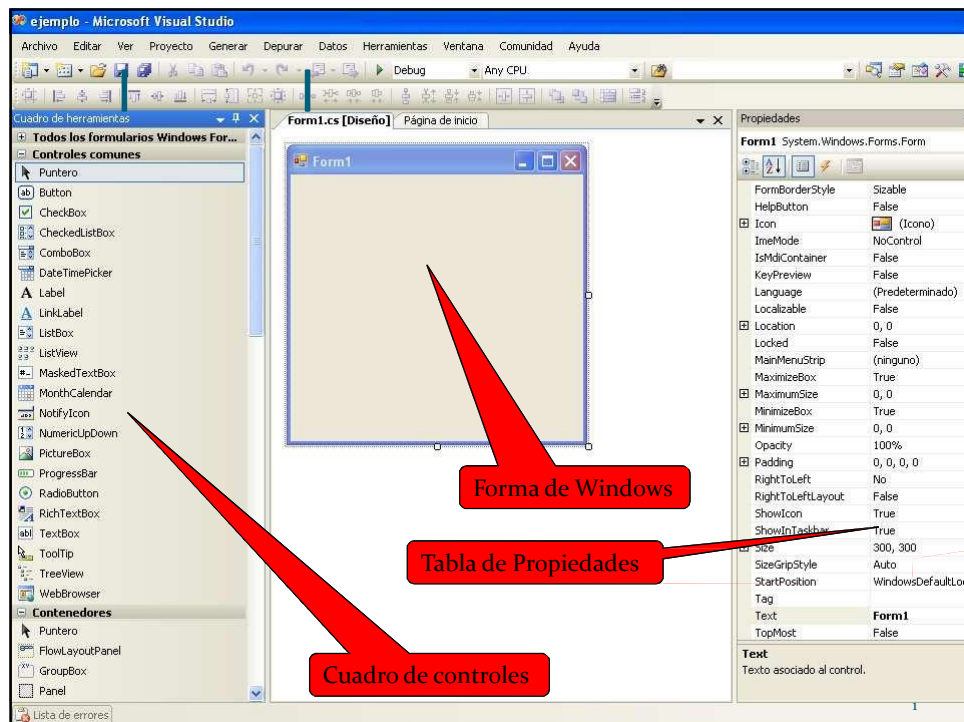
        // Método principal que se ejecuta al cargar el proyecto
        private void Form1_Load(object sender, EventArgs e)
        {
        }
    }
}
```

13

## Proyecto de formas de Windows (aplicación visual)

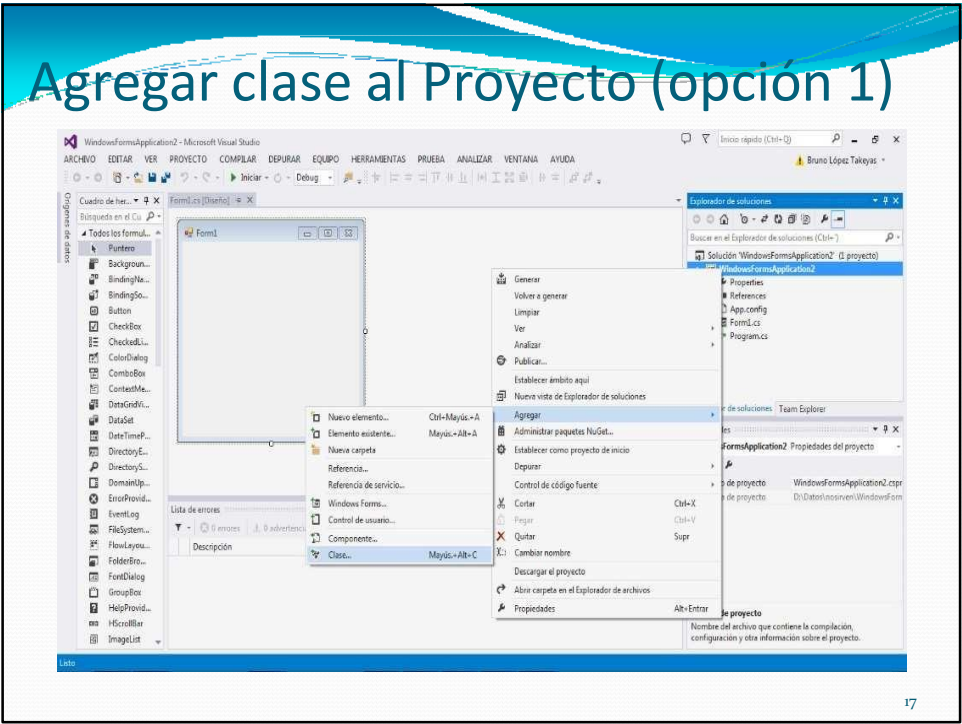


14



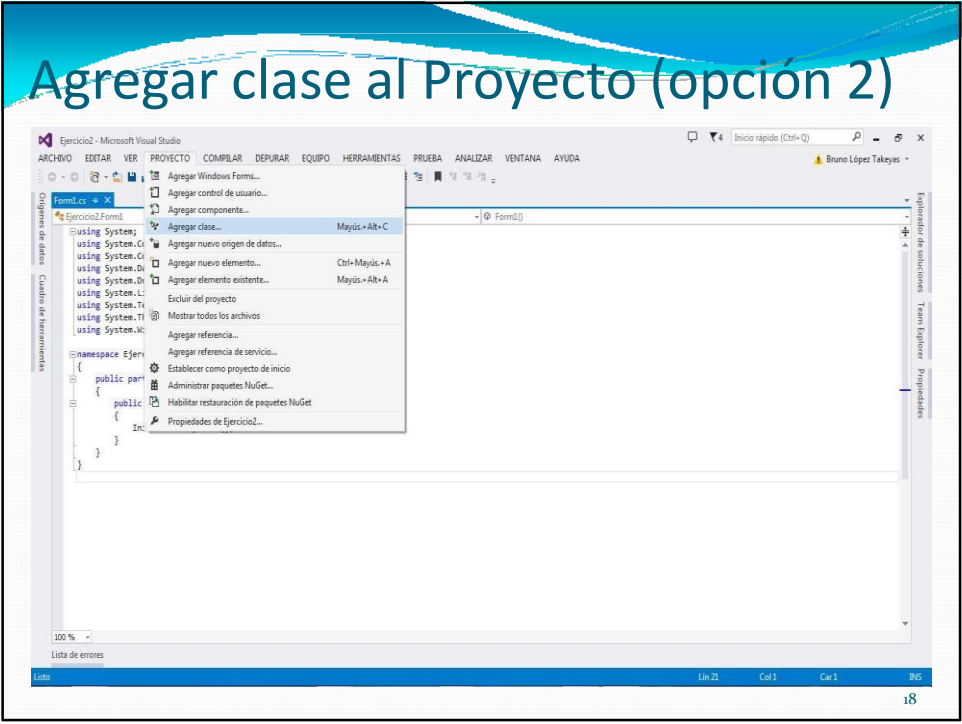


## Agregar clase al Proyecto (opción 1)



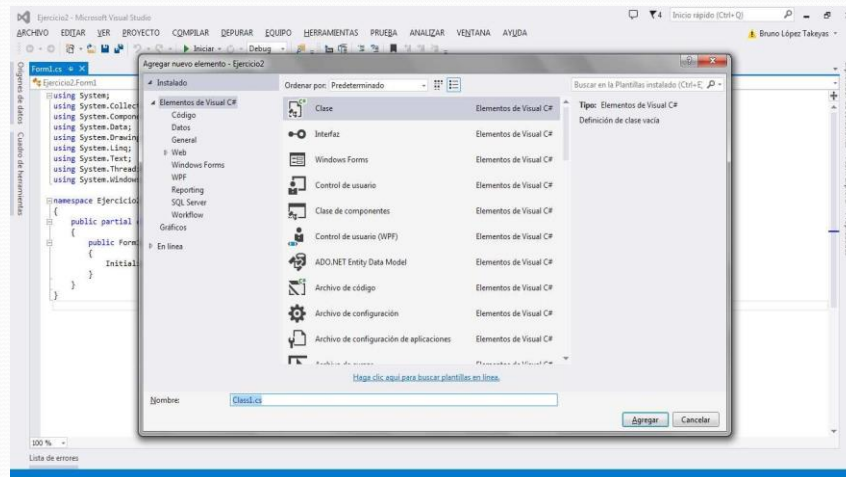
17

## Agregar clase al Proyecto (opción 2)



18

## Vía rápida para agregar clase al Proyecto (opción 3)



Shift + ALT + C

19

## Propiedades “Name” y “Text”

- ◆ Todos los controles tienen la propiedad **Name** que los identifica como un objeto de la forma
- ◆ La propiedad **Text** se utiliza para especificar el texto que despliega el control

20

## Propiedades "Name" y "Text"

The image shows a Windows Forms application with a form titled 'Form1'. On the form, there is a button labeled 'Ejecutar'. To the right of the form is the 'Propiedades' (Properties) window for the selected button, 'button1'. The properties window shows various attributes of the button. Red callout boxes highlight specific properties:

- Texto del botón (Text):** Points to the 'Text' property in the Properties window, which is set to 'Ejecutar'.
- Nombre del botón (Name):** Points to the 'Name' property in the Properties window, which is set to 'button1'.
- Otras propiedades del botón:** Points to the 'Enabled' property in the Properties window, which is set to 'True'.

Property	Value
(Name)	button1
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
Anchor	Top, Left
AutoEllipsis	False
AutoSize	False
AutoSizeMode	GrowOnly
BackColor	Control
BackgroundImage	(ninguno)
BackgroundImageLayout	Tile
CausesValidation	True
ContextMenuStrip	(ninguno)
Cursor	Default
DialogResult	None
Dock	None
Enabled	True
FlatAppearance	
FlatStyle	Standard
Font	Microsoft Sans Serif, 8.25
ForeColor	ControlText
GenerateMember	True
Image	(ninguno)

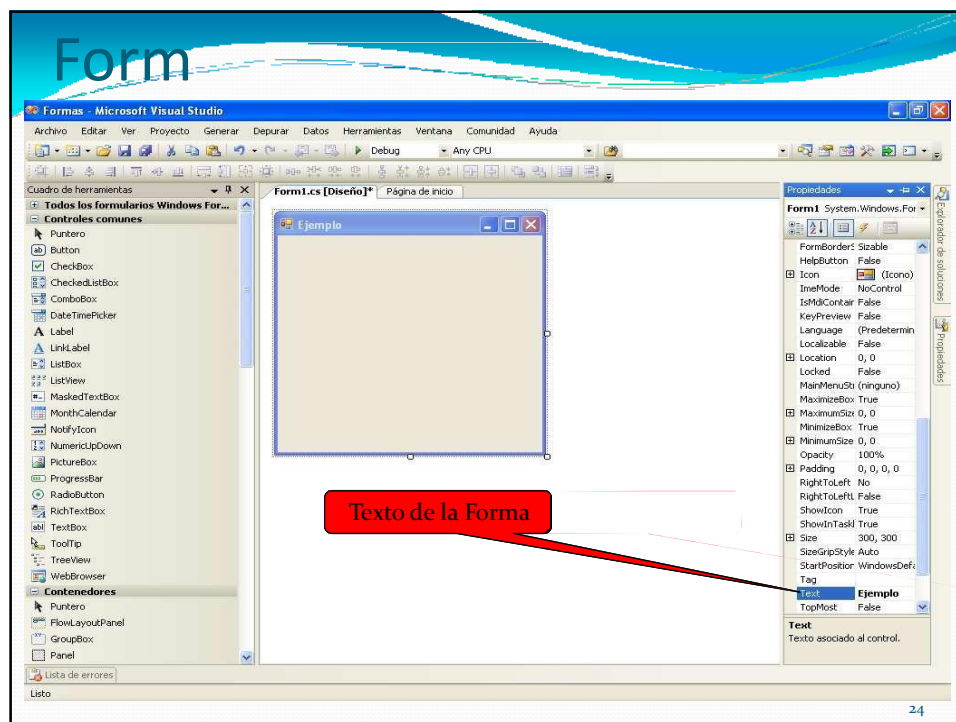
## Controles visuales

The image shows the 'Cuadro de herramientas' (Toolbox) in Visual Studio, displaying various Windows Forms controls. Red lines connect each control to its description in a blue box:

- Botón:** Ejecuta un conjunto de Sentencias cuando se oprime
- CheckBox:** Se utiliza para seleccionar varias opciones de un conjunto
- Etiqueta:** Muestra un mensaje fijo En la forma
- Lista:** Muestra una cuadro con una lista de mensajes
- RadioBotón:** Se utiliza para seleccionar Sólo una opción de un conjunto
- Cuadro de Texto:** Se utiliza para introducir o mostrar datos

The controls listed in the toolbox are: Puntero, Button, CheckBox, CheckedListBox, ComboBox, DateTimePicker, Label, LinkLabel, ListBox, ListView, MaskedTextBox, MonthCalendar, NotifyIcon, NumericUpDown, PictureBox, ProgressBar, RadioButton, RichTextBox, TextBox, ToolTip, TreeView, and WebBrowser.

## Terminar una aplicación



## textBox

Se usa para capturar datos

◆ Para ver su contenido se usa:

◆ `textBox1.Text`

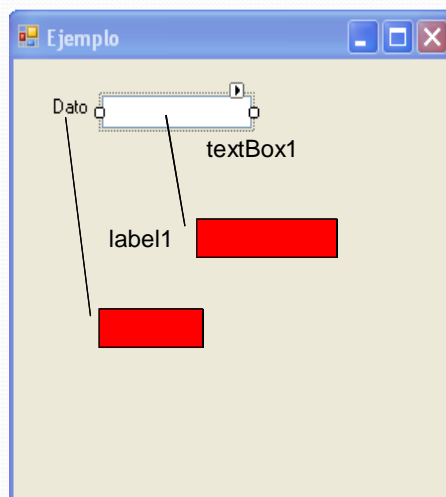
◆ Siempre es de tipo cadena

◆ Si requiere otro tipo de dato, es necesario usar `Parse( )`

◆ `int x = int.Parse(textBox1.Text);`

◆ Para limpiar su contenido:

◆ `textBox1.Clear( );`



25

## button

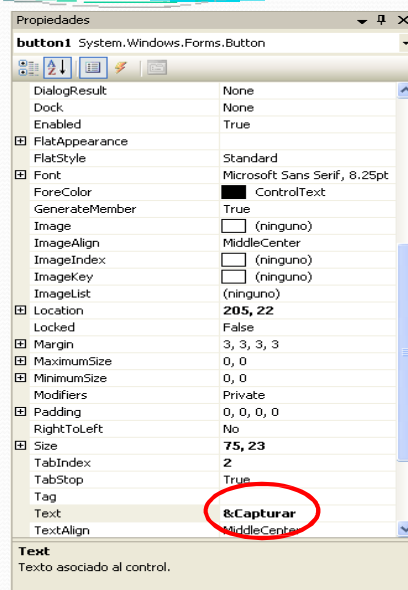
■ Use la propiedad `Text` para colocar el mensaje al botón

■ Coloque un `&` para habilitar la tecla directa

■ P. ejem. ALT-C

■ Al dar doble click, codifique:


```
private void button1_Click(object sender, EventArgs e)
{
    //Aquí se coloca el código
}
```



26



## MessageBox



The image shows three examples of MessageBox windows. The first is an information message box titled 'Este es un MessageBox' with an information icon and an 'OK' button. The second is a simple OK message box titled 'Hola Mundo' with an 'OK' button. The third is an error message box titled 'Este es un Mensaje de Error' with an error icon and 'OK' and 'Cancel' buttons.

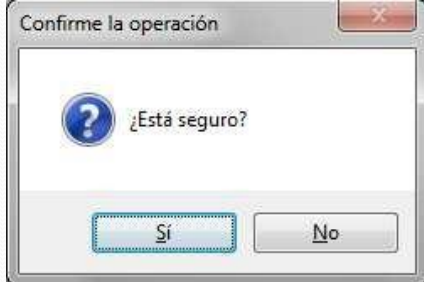
```
MessageBox.Show("Hola Mundo");
```

```
MessageBox.Show("Hola Mundo", "Este es un MessageBox", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

```
MessageBox.Show("Hola Mundo", "Este es un Mensaje de Error", MessageBoxButtons.OK, Cancel, MessageBoxIcon.Error);
```

27

## MessageBox



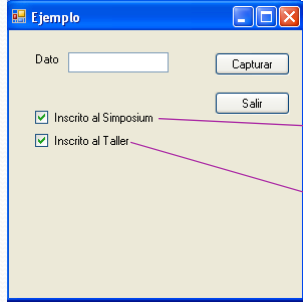
The image shows a confirmation message box titled 'Confirme la operación' with a question icon and 'Sí' and 'No' buttons.

```
DialogResult Respuesta;  
Respuesta = MessageBox.Show("¿Está seguro?", "Confirme la operación",  
MessageBoxButtons.YesNo, MessageBoxIcon.Question);  
  
if (Respuesta==DialogResult.Yes)  
    MessageBox.Show("Aceptado");  
else  
    MessageBox.Show("Rechazado");
```

28

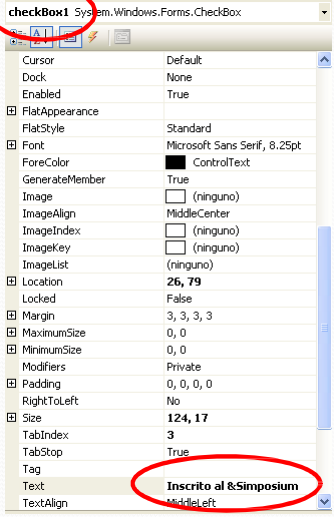
## checkBox

■ Permite seleccionar varias opciones



```
if (checkBox1.Checked)
    MessageBox.Show("Opción : "+checkBox1.Text);

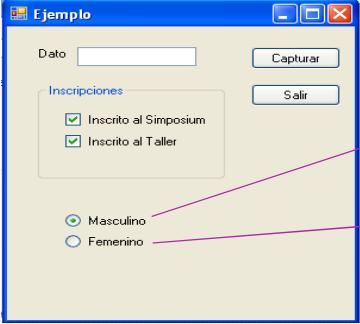
if (checkBox2.Checked)
    MessageBox.Show("Opción : "+checkBox2.Text);
```



29

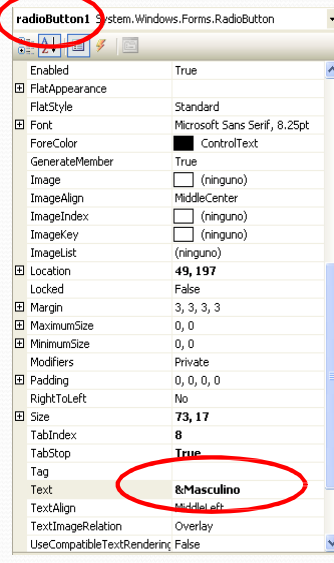
## radioButton

■ Permite seleccionar sólo una opción



```
if (radioButton1.Checked)
    MessageBox.Show(radioButton1.Text);

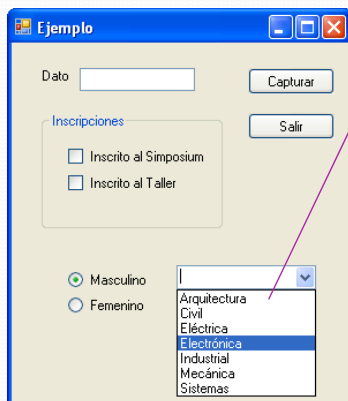
if (radioButton2.Checked)
    MessageBox.Show(radioButton2.Text);
```



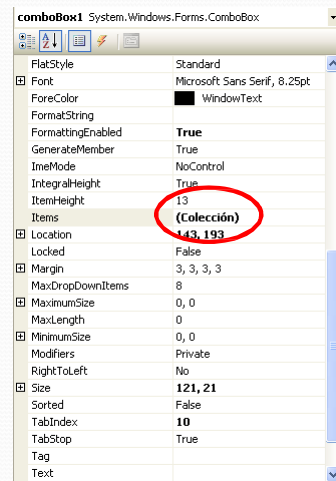
30

## comboBox

- Permite seleccionar sólo una opción de una lista desplegable



comboBox1

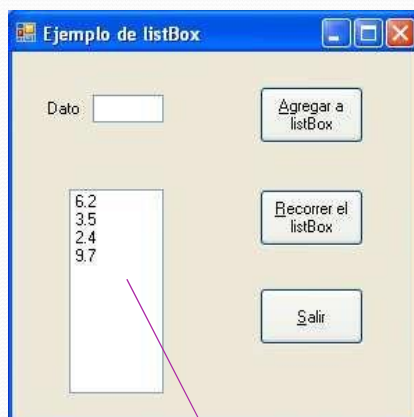


```
MessageBox.Show(comboBox1.Text);
```

31

## listBox

- Muestra una lista de datos de tipo string
- Es semejante a un arreglo



listBox1

- Para agregar datos:

```
listBox1.Items.Add(textBox1.Text);
```

- Para recorrer el listBox:

```
string salida = "";

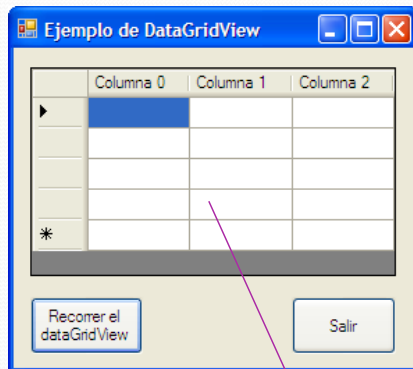
for (int i = 0; i <
listBox1.Items.Count; i++)
    salida =
salida+"\n"+listBox1.Items[ i ];

MessageBox.Show(salida);
```

32

## dataGridView

- Muestra un conjunto de datos de tipo string
- Es semejante a un arreglo bidimensional (matriz)



- Para crear las columnas:

```
dataGridView1.Columns.Add("Columna 0", "Columna 0");  
dataGridView1.Columns.Add("Columna 1", "Columna 1");  
dataGridView1.Columns.Add("Columna 2", "Columna 2");
```

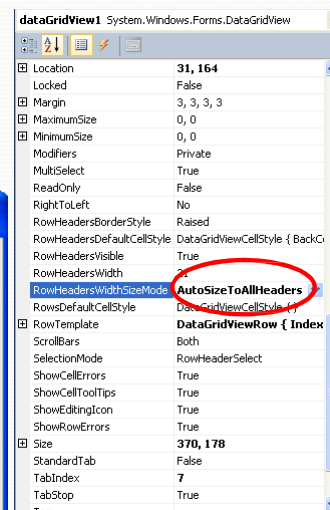
dataGridView1

33

## dataGridView

- Para etiquetar los renglones:

```
dataGridView1.Rows.Add(12);  
dataGridView1.Rows[0].HeaderCell.Value = "Enero";  
dataGridView1.Rows[1].HeaderCell.Value = "Febrero";  
dataGridView1.Rows[2].HeaderCell.Value = "Marzo";  
dataGridView1.Rows[3].HeaderCell.Value = "Abril";
```



34



## Propiedades de un dataGridView

Permitir agregar renglones

Permitir borrar columnas

Ancho de columna automatico

dataGridView1 System.Windows.Forms.DataGridView

- (ApplicationSettings)
- (DataBindings)
- (Name) dataGridView1
- AccessibleDescription
- AccessibleName
- AccessibleRole Default
- AllowDrop False
- AllowUserToAddRows True
- AllowUserToDeleteRows True
- AllowUserToOrderColumns False
- AllowUserToResizeColumns True
- AllowUserToResizeRows True
- AlternatingRowsDefaultCellStyle { }
- Anchor Top, left
- AutoSizeColumnsMode Fill
- AutoSizeRowsMode None
- BackgroundColor AppWorkspace
- BorderStyle FixedSingle
- CausesValidation True
- CellBorderStyle Single
- ClipboardCopyMode EnableWithAutoHeaderText
- ColumnHeadersBorderStyle Raised
- ColumnHeadersDefaultCellStyle { BackC
- ColumnHeadersHeight 18
- ColumnHeadersHeightSizeMode AutoSize
- ColumnHeadersVisible True
- Columns (Colección)

35

## Recorrido de un dataGridView

- Se requieren 2 ciclos: Columnas y Renglones
- Cada celda se accesa mediante:
  - `dataGridView1[columna, renglon].Value`

	Columna 0	Columna 1	Columna 2
1	2	3	
4	5	6	
*			

Recorrer el dataGridView

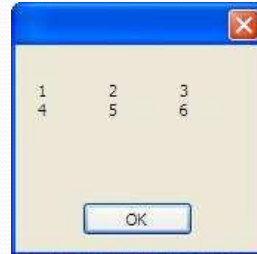
Salir

**NOTA:** El direccionamiento de las celdas es: [columna, renglon]

36



## Recorrido de un dataGridView



```
int r, c;
string salida = "";
for (r = 0; r < dataGridView1.Rows.Count; r++)
{
    salida = salida + "\n";
    for (c = 0; c < dataGridView1.Columns.Count; c++)
        salida = salida + dataGridView1[c, r].Value + "\t";
}
MessageBox.Show(salida);
```

37

## Ajustando las propiedades del dataGridView

```
// No permitir agregar ni eliminar renglones
dataGridView1.AllowUserToAddRows = false;
dataGridView1.AllowUserToDeleteRows = false;

// Autoajustar el ancho de las columnas
dataGridView1.AutoSizeColumnsMode =
DataGridViewAutoSizeColumnsMode.Fill;

// Seleccionar un renglón completo al hacer click
dataGridView1.SelectionMode =
DataGridViewSelectionMode.FullRowSelect;
```

38

## Ajustando las propiedades del dataGridView

```
// No se permite seleccionar varios renglones
dataGridView1.MultiSelect = false;

// Modo de solo lectura
dataGridView1.ReadOnly = true;
```

39

## ¿Cómo pasar los datos de un renglón de un dataGridView a los textBoxes?

- Hacer doble click en el dataGridView para abrir el método `CellClick()`

```
private void dataGridView1_CellClick(object sender, DataGridViewCellEventArgs e)
{
    . . .
}
```

- Establecer la propiedad para seleccionar solamente un renglón del dataGridView

```
// Seleccionar un renglón completo al hacer click
dataGridView1.SelectionMode = DataGridViewSelectionMode.FullRowSelect;
// No se permite seleccionar varios renglones
dataGridView1.MultiSelect = false;
```

40

## ¿Cómo seleccionar un renglón de un dataGridView?

```
// Declaración y creación de un objeto local
Auto miAuto = new Auto();

// Verificar si se seleccionó un renglón del dataGridView1
if (dataGridView1.CurrentRow == null)
{
    MessageBox.Show("Seleccione un auto de la lista");
    return;
}

// Obtiene las placas del auto seleccionado
miAuto.Placas = dataGridView1.CurrentRow.Cells[0].Value.ToString();
miAuto.Marca = dataGridView1.CurrentRow.Cells[1].Value.ToString();
miAuto.Modelo = dataGridView1.CurrentRow.Cells[2].Value.ToString();
```

41

## Prefijos sugeridos para los nombres de los controles visuales (notación húngara)

Control	Prefijo	Ejemplo(s)
Button	btn	btnSalir
CheckBox	chk	chkInscrito
ComboBox	cbo	cboEspecialidad
DataGridView	dtg	dtgIngredientes
Form	frm	frmPrincipal
GroupBox	grp	grpDatosAlumnos
Label	lbl	lblNombre
ListBox	lst	lstAlumnos
RadioButton	rad	radMasculino
TextBox	txt	txtRadio

<http://support.microsoft.com/kb/173738/es>

42

Control	Prefijo	Ejemplo
Button	btn	btnSave
Calendar	cal	calMyDate
CheckBox	chk	chkMailList
CheckBoxList	chk1	chk1Address
ComboBox	cbo	cboProvincias
DataGrid	dtg	dtgClientes
DataList	dlst	dlstProductos
DateTimePicker	dtp	dtpFechaInicio
Dialog, ColorDialog, FontDialog	dlg	dlgFileSave, dlgColores
Image	img	imgLogo
Label	lbl	lblApellido
LinkLabel	lbl	lblWebPage
ListBox	lst	lstCompany
ListView	lvw	lvwRecibos
MainMenu	mnu	mnuArchivo
MenuItem	mnu	mnuCerrar
Panel	pnl	panSection
PictureBox	pic	picLogo
RadioButton	rad	radSex
RadioButtonList	rbl	rblAgeGroup
RichTextBox	rtb	rtbDocumento
Repeater	rep	repSection
TabControl	tab	tabUsuario
TextBox	txt	txtApellido
TreeView	tvw	tvwDirectorios
ToolBar	tbr	tbrStatus
Timer	tmr	tmrSegundos
Validators (Todos)	val	valCreditCardNumber
ValidationSummary	vals	valsErrors

3

## Calcular diferencia de tiempo

- ◆ Se declaran las variables de tipo **DateTime**

```
DateTime HoraInicio, HoraFin;
```

```
HoraInicio = DateTime.Now;
```

```
HoraFin = DateTime.Now;
```

```
int tiempo = Math.Abs(HoraFin.Millisecond -  
HoraInicio.Millisecond);
```

```
MessageBox.Show("Tiempo de ejecución: " +  
tiempo.ToString()+" ms.");
```

44



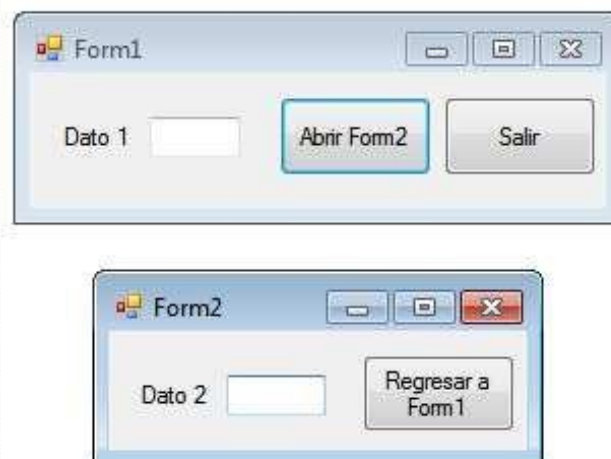
## Otra forma de calcular diferencia de tiempo

- ◆ La palabra reservada **var** sólo se puede utilizar en un contexto local

```
var cronometro = System.Diagnostics.Stopwatch.StartNew();  
. . . . .  
cronometro.Stop();  
  
MessageBox.Show(cronometro.ElapsedMilliseconds.ToString());
```

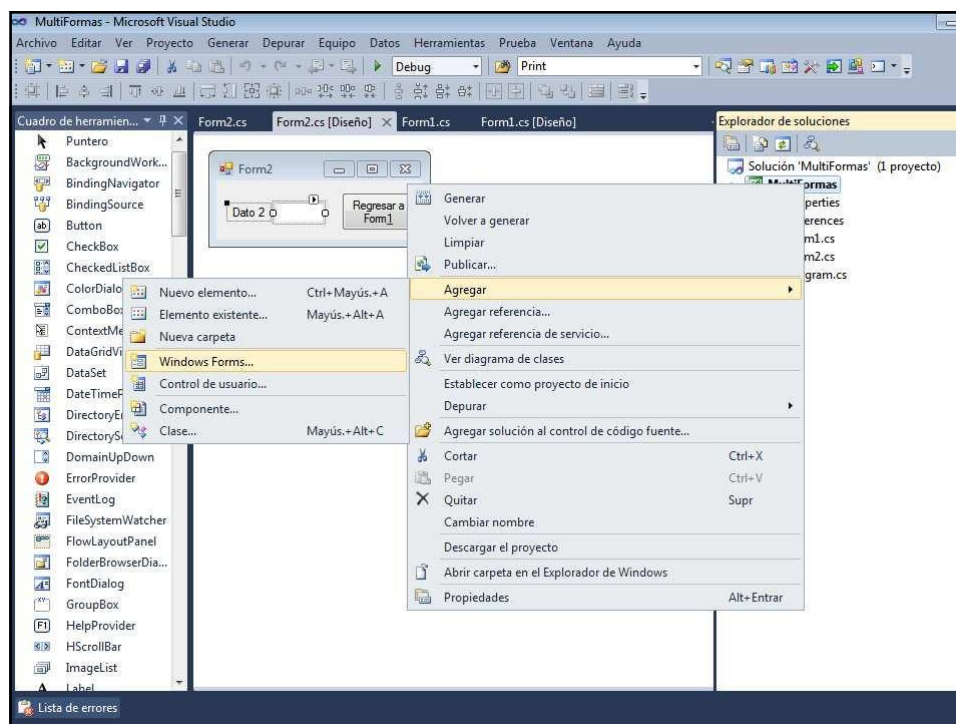
45

## Proyecto con varias formas

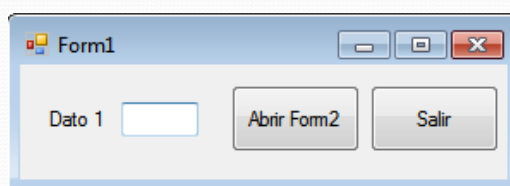


46



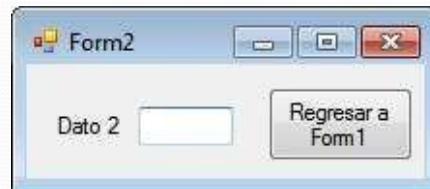


## Invocar la Form2 desde la Form1



```
private void btnAbrirForm2_Click(object sender, EventArgs e)
{
    Form2 miForma2 = new Form2();
    miForma2.Show();
}
```

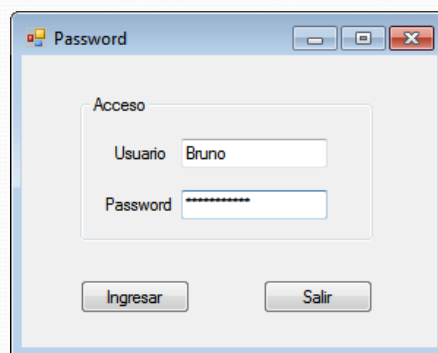
## Regresar a la Form1



```
private void btnRegresarForm1_Click(object sender, EventArgs e)
{
    this.Close();
}
```

49

## Uso de contraseña (password)



```
private void Form1_Load(object sender, EventArgs e)
{
    txtPassword.PasswordChar = '*';
}
```

50

