



Controles que vamos a ver hoy

PictureBox

El PictureBox es un objeto que nos sirve para contener imágenes, ya sea que nosotros directamente dibujemos sobre él o bien que la imagen la importemos de otra parte.

Principales propiedades

Name. Permite asignarle un nombre al control. Se recomienda anteponer un **pbx**Nombre.

Image. Carga o importa una imagen desde otra carpeta.

SizeMode. Permite ajustar el tamaño del objeto al de la imagen importada.

Principales métodos

Click. Ocurre cuando se da clic con el mouse.

MouseMove. Ocurre cuando el mouse se mueve sobre el objeto.

MouseDown. Ocurre cuando se oprime el botón del mouse sobre el objeto.

MouseUp. Ocurre cuando se libera el botón del mouse sobre el objeto.

ContextMenuStrip

El ContextMenuStrip nos permite visualizar un menú desplegable.

Principales propiedades

Name. Permite asignarle un nombre al control. Se recomienda anteponer un **cmstp**Nombre.

AddRange. Permite adicionar los submenús dentro del control.

Principales métodos

Opening. Ocurre cuando es abierto el menú.

ToolStripMenuItem

Este control nos permite crear menús.

Principales propiedades

Name. Permite asignarle un nombre al control. Se recomienda anteponer un **tstp**Nombre.

Text. Es el texto que describe las funciones del menú.

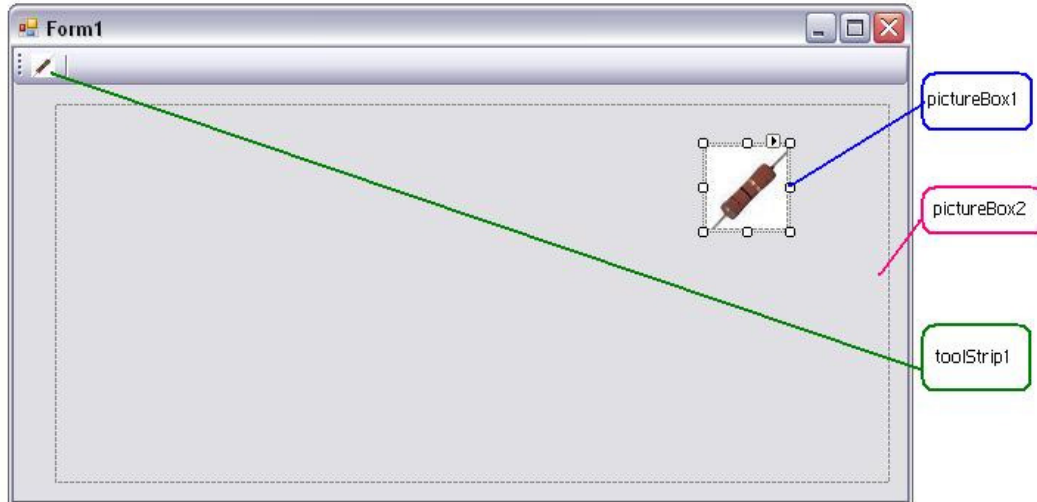
Principales métodos

Click. Ocurre cuando se da un clic con el mouse sobre el menú.

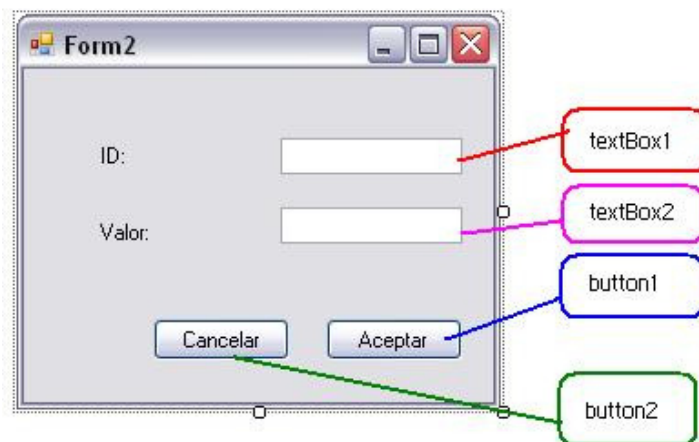
Drag and Drop

En esta práctica vamos a crear dinámicamente controles con efecto arrastre por medio de pictureBox.

1. Crea un nuevo proyecto y llama a la solución drag_and_drop.
2. Diseña el siguiente formulario.



3. Importa imágenes para cada uno de los controles.
4. Adiciona un nuevo formulario y diseñalo de la siguiente manera:





TEMA4.PROGRAMACIÓN ORIENTADA A OBJETOS

5. Crea una clase **resistencias.cs** y adiciona el siguiente código:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Drawing;
using System.Windows.Forms;

namespace drag_and_drop
{
    public class resistencias
    {
        private int id_resistencia;
        private int valor_resistencia;
        private System.Windows.Forms.PictureBox pbxR;
        private System.Windows.Forms.ContextMenuStrip cms;
        private System.Windows.Forms.ToolStripMenuItem PropiedadesToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
        public bool mover = false;
        private bool isDragging = false;
        private int clickOffsetX, clickOffsetY;
        public resistencias(int ID, int Val, PictureBox pb, PictureBox contenedor, Form1 frmPricpal)
        {
            Id_Resistencia = ID;
            Valor_Resistencia = Val;

            cms = new ContextMenuStrip();
            //
            // contextMenuStrip1
            //
            this.cms.Items.AddRange(new System.Windows.Forms.ToolStripItem[] {
            this.PropiedadesToolStripMenuItem});
            this.cms.Name = "contextMenuStrip1";
            this.cms.Size = new System.Drawing.Size(167, 48);
            //
            // PropiedadesToolStripMenuItem
            //
            this.PropiedadesToolStripMenuItem.Name = "PropiedadesToolStripMenuItem";
            this.PropiedadesToolStripMenuItem.Size = new System.Drawing.Size(166, 22);
            this.PropiedadesToolStripMenuItem.Text = "Ver Propiedades";
            this.PropiedadesToolStripMenuItem.Click += new
System.EventHandler(this.PropiedadesTSMItem_Click);

            pbxR = new PictureBox();
            pbxR.Name = "Resistencia" + ID;
            pbxR.Size = pb.Size;
            pbxR.Image = pb.Image;
            pbxR.Location = new System.Drawing.Point(10, 10);
            pbxR.MouseDown += new
System.Windows.Forms.MouseEventHandler(pbxR_MouseDown);
```



TEMA4.PROGRAMACIÓN ORIENTADA A OBJETOS

```
pbxR.MouseMove += new
System.Windows.Forms.MouseEventHandler(pbxR_MouseMove);
pbxR.MouseUp += new System.Windows.Forms.MouseEventHandler(pbxR_MouseUp);
pbxR.ContextMenuStrip = cms;

contenedor.Controls.Add(pbxR);
}

public int Id_Resistencia
{
    set
    {
        if (value > 0)
            id_resistencia = value;
        else
            id_resistencia = 0;
    }
    get
    {
        return id_resistencia;
    }
}

public int Valor_Resistencia
{
    set
    {
        if (value > 0)
            valor_resistencia = value;
        else
            valor_resistencia = 0;
    }
    get
    {
        return valor_resistencia;
    }
}

public void pbxR_MouseUp(object sender, MouseEventArgs e)
{
    isDragging = false;
}

public void pbxR_MouseMove(object sender, MouseEventArgs e)
{
    if (isDragging == true)
    {
        pbxR.Left = e.X + pbxR.Left - clickOffsetX;
        pbxR.Top = e.Y + pbxR.Top - clickOffsetY;
    }
}

public void pbxR_MouseDown(object sender, MouseEventArgs e)
```



```
{
    isDragging = true;
    clickOffsetX = e.X;
    clickOffsetY = e.Y;
}

public void PropiedadesTSMItem_Click(object sender, EventArgs e)
{
    Form2 prop = new Form2(this);
    prop.Show();
}
}
```

Edita el formulario Form1.cs

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

namespace drag_and_drop
{
    public partial class Form1 : Form
    {
        private int contador;

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            contador = 0;
        }

        private void toolStripButton1_Click(object sender, EventArgs e)
        {
            resistencias R = new resistencias(contador, 1000, this.pictureBox1, this.pictureBox2, this);
            contador++;
        }
    }
}
```



TEMA4.PROGRAMACIÓN ORIENTADA A OBJETOS

6. Edita el formulario Form2.cs

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

namespace drag_and_drop
{
    public partial class Form2 : Form
    {
        resistencias R;
        public Form2()
        {
            InitializeComponent();
        }

        public Form2(resistencias R)
        {
            this.R = R;
            InitializeComponent();
            textBox1.Text = ""+R.Id_Resistencia;
            textBox2.Text = ""+R.Valor_Resistencia;
        }

        private void button2_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            R.Valor_Resistencia = Convert.ToInt32(textBox2.Text);
            this.Close();
        }

        private void Form2_Load(object sender, EventArgs e)
        {
            textBox1.ReadOnly = true;
            textBox1.BackColor = Color.Gray;
        }
    }
}
```

7. Compílalo, ejecútalo y anota tus comentarios. **NO OLVIDES DECLARAR LOS EVENTOS!!!**