

INTRODUCCIÓN

DataGridView.

Generalmente este control es utilizado para trabajar con información de Bases de Datos, permite adicionar registros, modificar datos y desplegarlos en una tabla.

Para nuestro objetivo lo utilizaremos como visor de los datos contenidos en el archivo.

Principales propiedades:

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

Columns (Collection). Permite adicionar las columnas y la información que se desplegará como encabezado en cada una de las columnas.

BackColor. Cambia el color de fondo del objeto.

BorderStyle. Indica como se verá el borde del objeto. Puede quitarlo si no desea que se visualice seleccionando None.

Principales eventos:

Clear. Permite limpiar todas las celdas del Data Grid. Ejemplo:

```
DataGridView1.Rows.Clear();
```

Rows.Add. Permite adicionar datos en forma de renglones al data grid. Ejemplo:

```
DataGridView1.Rows.Add(datoColumna1, datoColumna1, datoColumna1, ..., datoColumnaN);
```

Más propiedades de la clase DataGridView

RowCount. Contiene el número total de renglones actualmente en el DataGridView. Ejemplo:

```
int num_renglones;  
num_renglones = DataGridView1.RowCount
```

CurrentCell.RowIndex. Devuelve el índice del renglón que este activo al ser seleccionado por el mouse.

Ejemplo:

```
int renglon_actual;  
Renglon_actual = DataGridView1.CurrentCell.RowIndex;
```

El CurrentCell.RowIndex devuelve un -1 si no ha sido seleccionado ninguna celda.

Rows.RemoveAt(indice_renglon_a_eliminar). Elimina el renglón que indiquemos como parámetro.

Ejemplo:

```
DataGridView1.Rows.RemoveAt(renglon_actual);
```

El renglon_actual debe ser un número que sea mayor que -1 y menor al número de renglones totales del DataGridView.

DataGridView1[indice_columna, indice_renglon].Value.ToString()

Devuelve el contenido de la celda especificada. Ejemplo:

```
string miDato;  
miDato = DataGridView1[0, 1].Value.ToString();
```

También se asigna un valor a la celda especificada

```
string MiDato="Hola";  
DataGridView1[1, 0].Value.ToString()=miDato;
```

DataGridView1.Rows[indice_renglon].Cells[indice_columna].Value.ToString()

Devuelve el contenido de la celda especificada. Ejemplo:

```
string miDato;  
miDato = DataGridView1.Rows[0].Cells[1].Value.ToString();
```

También se asigna un valor a la celda especificada

```
string MiDato="Hola";  
DataGridView1.Rows[0].Cells[2].Value.ToString()=miDato;
```

Básicamente con estos dos objetos podemos comenzar a trabajar en nuestra práctica.

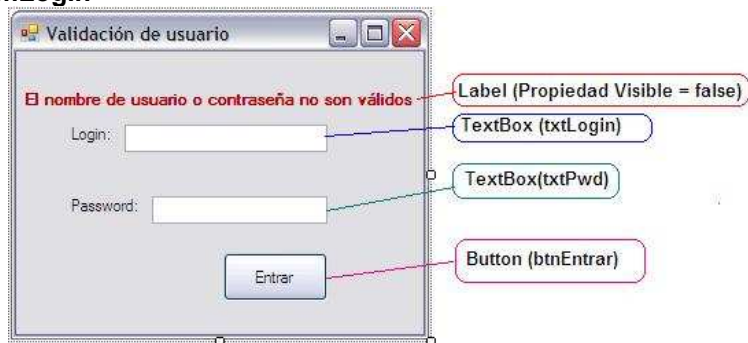
Ejercicios de la Práctica:

A través de esta práctica el alumno utilizará todas las herramientas vistas hasta ahora para crear una Aplicación funcional.

1. Crea un nuevo proyecto seleccionando la opción **Windows Application**. Llámala a la solución Practica27_bdAlumnos, y el nombre del proyecto Practica27_bdAlumnos

Crea los siguientes formularios dentro de tu proyecto:

1) Formulario frmLogin



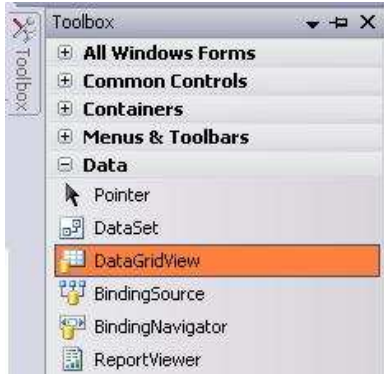
Cambia la propiedad del txtPwd PasswordChar a "*"

2) Formulario frmPrincipal (Selecciona del Explorador de solución el proyecto con el botón derecho del Mouse Add> New Item > Windows Form)



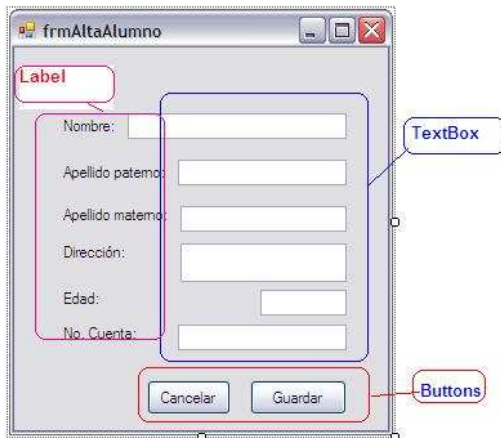
Adiciona imágenes a tus botones mediante la propiedad **Image > Import...** y cambia de lugar el texto del botón con la propiedad **TextAlign**

Adiciona a tu formulario el Control DataGridView de la **ToolBox > Data > DataGridView**



Cambia la propiedad **SelectionMode** a FullRowSelect así como **ReadOnly=True**

3) Formulario frmAltaAlumno (Selecciona del Explorador de solución el proyecto con el botón derecho del Mouse Add> New Item > Windows Form)



Edita ahora el código para cada uno de los formularios de la siguiente manera:

1) frmLogin

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

namespace bdAlumnos
{
    public partial class frmLogin : Form
    {
        public frmLogin()
        {
            InitializeComponent();
        }
    }
}
```

```

private void btnValidarUsuario_Click(object sender, EventArgs e)
{
    if ((txtLogin.Text == "tecnicas") && (txtPwd.Text == "123"))
    {
        frmPrincipal fp = new frmPrincipal();
        fp.Show();
        this.Visible=false;
    }
    else
    {
        lblError.Visible = true;
    }
}
}
}

```

2) frmPrincipal

```

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

namespace bdAlumnos
{
    public partial class frmPrincipal : Form
    {
        private long ncuenta=-1;
        private int renglon=-1;
        public frmPrincipal()
        {
            InitializeComponent();
            this.formato_datagrid(dataGridView1);
        }

        public void formato_datagrid(DataGridView dg)
        {
            dg.Columns.Clear();
            dg.Columns.Add("nom", "Nombre");
            dg.Columns.Add("ap", "Apellido paterno");
            dg.Columns.Add("am", "Apellido materno");
            dg.Columns.Add("dir", "Direccion");
            dg.Columns.Add("edad", "Edad");
            dg.Columns.Add("ncuenta", "No. Cuenta");
        }

        private void btnAltaAlumno_Click(object sender, EventArgs e)
        {
            frmAltaAlumno alta = new frmAltaAlumno(this);
            alta.Show();
        }
    }
}

```

```

private void btnCerrar_Click(object sender, EventArgs e)
{
    Application.Exit();
}

public void adicionar_alumno(alumno alum)
{
    dataGridView1.Rows.Add(alum.nombre, alum.apellido_materno, alum.apellido_materno,
alum.direccion, alum.edad, alum.ncuenta);
}

private void btnEliminar_Click(object sender, EventArgs e)
{
    renglon = dataGridView1.CurrentRow.Index;
    ncuenta = Convert.ToInt64(dataGridView1[5, renglon].Value.ToString());
    if (ncuenta != -1)
    {
        if (MessageBox.Show("Esta seguro que desea eliminar el alumno " + dataGridView1[0,
renglon].Value.ToString(), "Eliminar registro", MessageBoxButtons.YesNo) == DialogResult.Yes)
            dataGridView1.Rows.RemoveAt(renglon);
    }
}
}
}
}

```

3) frmAltaAlumno

```

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

namespace Sistema_alumnos
{
    public partial class frmAltaAlumno : Form
    {
        private frmPrincipal frmP;
        public frmAltaAlumno()
        {
            InitializeComponent();
        }

        public frmAltaAlumno(frmPrincipal frmP)
        {
            InitializeComponent();
            frmP = frmP;
        }

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

        private void btnGuardar_Click(object sender, EventArgs e)

```

```

    {
        alumno nuevo_alumno = new alumno(txtNombre.Text, txtApellido_paterno.Text,
txtApellido_materno.Text, txtDireccion.Text, Convert.ToInt32(txtEdad.Text),
Convert.ToInt64(txtNcuenta.Text));
        this.frmP.adicionar_alumno(nuevo_alumno);
        this.Close();
    }
}
}

```

Ejercicios adicionales:

1. Adiciona dentro de tu clase un constructor para que reciba directamente los valores y puedan ser utilizados al guardar el alumno:

Ejemplo:

```

alumno nuevo_alumno = new alumno(txtNombre.Text, txtApellido_paterno.Text,
txtApellido_materno.Text, txtDireccion.Text, Convert.ToInt32(txtEdad.Text),
Convert.ToInt32(txtNcuenta.Text));

```

2. Adiciona tu librería validaciones.dll para que valides los campos antes de ser enviados en el constructor

Ejemplo:

```

string nombre,apellido_p,apellido_m,direccion;
int edad, ncuenta;

```

```

nombre= valida_texto(txtNombre.Text);
apellido_p= valida_texto(txtApellido_paterno.Text);
apellido_m= valida_texto(txtApellido_materno.Text);
direccion= valida_texto(txtDireccion.Text);
edad= valida_entero(txtEdad.Text);
ncuenta= valida_entero(txtNcuenta.Text);

```

```

alumno nuevo_alumno = new alumno(nombre, apellido_p, apellido_m, direccion, edad, ncuenta);

```

3. Adiciona los comentarios.
4. Completa el código para que el botón btnActualizarAlumno, envíe a un nuevo formulario mostrando los datos del alumno seleccionado y actualice su información.
5. Completa el código para que el botón btnGuardar envíe esta información a un archivo alumno.dat (reutiliza el código ya generado en las prácticas anteriores, y en lugar de vaciar los datos de la lista hazlo recorriendo el DataGridView).
6. Adiciona código para que al entrar a la aplicación cargue la información de alumnos desde el archivo alumno.dat (Modifica tu código de las prácticas anteriores para que en lugar de cargar en una lista lo cargue en el DataGridView).