Trabajo Practico n 4

Franco Buonfrate

Descripción

Mi programa está enfocado en facilitar la venta de artículos en un local de pintura. Esta esta dirigido directamente para el vendedor el cual manejara el 100% del programa, teniendo “del otro lado del mostrador” al cliente.

Funcionamiento

La aplicación consta de 6 forms los cuales se generarán y cerrarán a medida que avance el programa.
En primera instancia se mostrará el form de clientes, en el cual se encuentra un Datagrid que es cargado de manera asíncrona mediante la función Leer de la biblioteca de ManejoDeDatos, Con la información de los clientes almacenada previamente en la base de datos.

En este form se puede:

-Dar de alta un nuevo cliente, completando todos los campos y presionando el botón “crear”. Internamente este botón crea una nueva instancia de cliente con los parámetros previamente obtenidos, y validados mediante eventos; y lo envía directamente a la base de datos mediante la función estática BaseDeDatos.Guardar().

-Modificar un usuario. Al clickear sobre la barra lateral derecha del DataGrid del usuario a modificar se mostrarán los datos del mismo (gracias al evento dtgvClientes_CellClick). Luego de modificar el campo deseado se oprime el botón “Modificar” e internamente buscara al usuario por su id única en la base de datos y lo sobrecarga con los nuevos datos.

-Eliminar Usuario. Se selecciona el usuario y se presiona el botón “Eliminar”, el programa buscará nuevamente el usuario por su id en la base de datos y lo eliminará de la misma.

Para pasar al siguiente form se deberá seleccionar un cliente y presionar el botón "Productos”, este generará un form Productos pasándole el cliente.

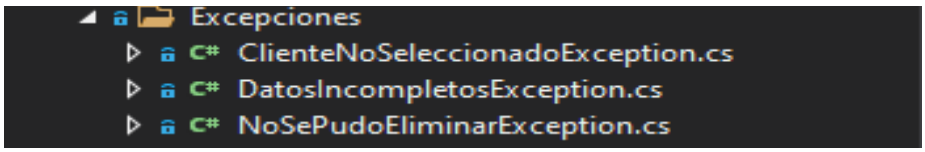
El form Productos consta de 3 botones, uno parar cada tipo de producto. Cada botón abrirá un form distinto mediante el cual se podrán agregar los distintos productos a la compra.

Finalmente se pasa al form de Factura, en el que se podrá generar un ticket que se guardará en el escritorio con los datos de la facturación en una nueva carpeta llamada “Facturas” y a su vez se los datos de la compra serán serializados a fin de poder ser enviados a otro programa en el que se administren las compras.

Temario

-Excepciones

Biblioteca de clase: Excepciones



Se utilizan en el form Clientes y en el form factura

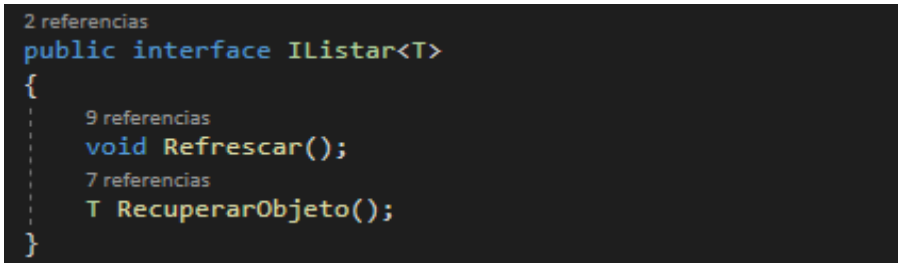
-Pruebas Unitarias

Proyecto de pruebas: EntidadesTest

Se prueba la funcionalidad de Entidades

-Tipos Genéricos

Interface Ilistar del proyecto de Windows Forms



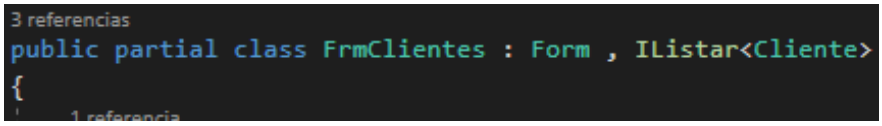
En los forms frmCliente y frmFactura los cuales aplican el tipo genérico para retornar los ítems cargados a la lista.

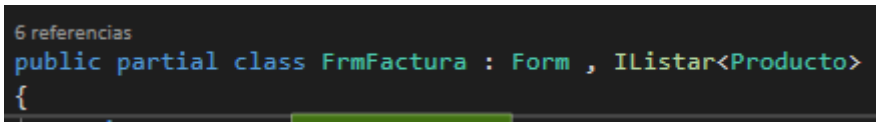
-Interfaces

Form Program: Pinturería

Interfaz: IListar

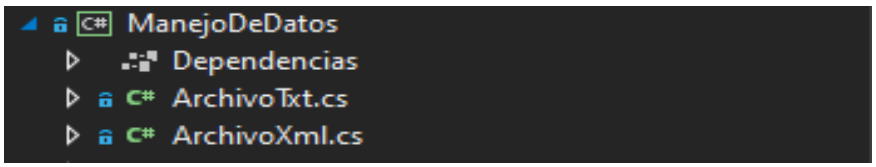
Se utiliza en los forms frmClientes y frmFactura





-Archivos

Biblioteca de clases: ManejoDeDatos

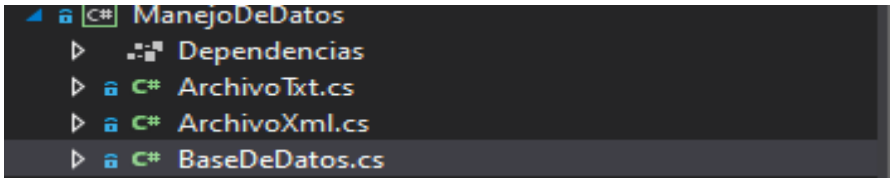


Se utiliza para crear archivos de texto en el form Factura y Para serializar los datos de los clientes en el form Clientes

-Conexión a base de datos

Biblioteca de clases: ManejoDeDatos

Clase estática: BaseDeDatos



Se aplica en el form frmClientes en el que se hace puede cargar, leer, modificar y eliminar directamente a la base de datos.

-Delegados

Método Refrescar del frmClientes

```
6 referencias
public async void Refrescar()
{
    if (dtgvClientes.InvokeRequired)
    {
        Action refrescar = Refrescar;
        dtgvClientes.Invoke(refrescar);
    }
    else
    {
        dtgvClientes.DataSource = await BaseDeDatos.Leer();
        dtgvClientes.Refresh();
        dtgvClientes.Update();
        dtgvClientes.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
        dtgvClientes.AutoSizeRowsMode = DataGridViewAutoSizeRowsMode.None;
    }
}
```

Declaro un delegado de tipo Action, el cual recibe métodos que no reciben parámetro y retornan void, para invocar el datagrid en otro hilo mediante el método Invoke.

-Hilos

Método Leer de la Clase BaseDeDatos

```
1 referencia
public static async Task<List<Cliente>> Leer()
{
    List<Cliente> clientes = new List<Cliente>();

    try
    {
        clientes = await Task.Run(() => {
            command.Parameters.Clear();
            connection.Open();
            command.CommandText = $"SELECT * FROM Datos_Clientes";
            using (SqlDataReader dataReader = command.ExecuteReader())
            {
                while (dataReader.Read())
                {
                    Cliente cliente = new Cliente(dataReader["nombre"].ToString(),
                                                  dataReader["apellido"].ToString(),
                                                  long.Parse(dataReader["dni"].ToString()),
                                                  dataReader["celular"].ToString(),
                                                  dataReader["mail"].ToString(),
                                                  int.Parse(dataReader["ID"].ToString()));
                    clientes.Add(cliente);
                }
            }
            return clientes;
        });
    }
    catch (Exception)
    {
        throw;
    }
    finally
    {
        connection.Close();
    }
    return clientes;
}
```

Este método es asincrónico, para que, de esta forma al momento de llamarlo, si se posee una base de datos muy extensa, este no congele el programa y se pueda seguir trabajando hasta que se termine de cargar la lista.

-Eventos y Métodos extensión

Clase MetodoExtensionValidarForms en el proyecto de Windows Forms Pintura

```
namespace Pintureria
{
    0 referencias
    public static class MetodoExtensionValidarForms
    {
        2 referencias
        public static void SoloTexto(this Form frmAValidar ,object sender, KeyPressEventArgs e)
        {
            if (!char.IsLetter(e.KeyChar) && (e.KeyChar != (char)Keys.Space && e.KeyChar != (char)Keys.Back))
            {
                e.Handled = true;
            }
        }
        4 referencias
        public static void SoloNumero(this Form frmAValidar ,object sender, KeyPressEventArgs e)
        {
            if (!char.IsNumber(e.KeyChar) && e.KeyChar != (char)Keys.Back)
            {
                e.Handled = true;
            }
        }
    }
}
```

Estos eventos son utilizados por los forms en los cuales encontramos textBox los cuales solamente deberían recibir un tipo de dato.
Estos son de extensión, ya que los textBox se generan con la instancia de los forms.