Laboratorio II - TP4
ECO CENTER - School of English
Bianca Casetta

Introducción

Este programa refleja un programa utilizado por un instituto privado de inglés, donde se pueden cargar nuevos estudiantes, modificarlos, darlos de baja del sistema, listarlos y acceder a la información de los cursos dados.

Aspecto/Navegación

Al ejecutar el programa, solo se ve el logo y el menú horizontal como “pestañas” con las opciones Estudiantes, Cursos y Salir.
Según la opción horizontal que se seleccione, se despliega un menú vertical en el caso de Estudiantes y Cursos. Si se clickea el logo, se vuelve al estado original en el que empieza a correr el programa. Si se clickea Salir, se reconfirma la salida y se cierra el programa.
Según la opción vertical que se seleccione, se despliega un formulario acorde a la acción seleccionada a la derecha del menú vertical.

Aclaraciones sobre clases de Entidades

Instituto - Clase que contiene una lista de los diferentes tipos de estudiantes.
EstudianteParticular - Aquel que solo va al instituto a repasar algo específico y de manera individual. Paga por hora y no cuenta con una nivelación por ser personalizada.
EstudianteCurso - Aquel que cursa en el instituto durante el ciclo lectivo y de manera grupal. Paga por mes y tiene un Curso asociado.
EstudianteExterno - Aquel que no cursa en el instituto, solo asiste en época de exámenes para rendir y no tiene Curso, solo Nivel.

Temas Aplicados

10. Excepciones

Creé dos Exceptions propias:

CamposIncompletosException - Si aunque sea UN campo textBox queda sin completar al dar de Alta o Modificar un Estudiante, se lanza la Exception porque todos los campos son obligatorios. Luego con el mensaje de la Exception se muestra un mensaje de error por MessageBox.

EstudianteYaExistenteException - Si se intenta dar de alta a un estudiante cuyo DNI ya se encuentra registrado en el sistema, se lanza la Exception porque no se admiten dos Estudiantes con el mismo DNI. Luego con el mensaje de la Exception se muestra un mensaje de error por MessageBox.

11. Pruebas Unitarias

Se prueba si las sobrecargas de igualdad entre Instituto/Estudiante y Estudiante/Estudiante retornan True.

```
[TestMethod]
0 references
public void EstudianteSobrecargaIgualdad_CuandoElEstudianteTieneElMismoDni_DeberiaRetonarTrue()
{
    //Arrange
    Domicilio dom = new Domicilio("Calle Falsa", 123, "Springfield", "Missouri", 1234);
    Estudiante e1 = new EstudianteParticular("Juan", "Perez", 12345678, new DateTime(1990, 08, 25), "Masculino", 1122223333, dom,
    Estudiante e2 = new EstudianteExterno("Maria", "Perez", 12345678, new DateTime(2000, 12, 31), "Femenino", 5491177778888, dom,

    //Act
    bool estudianteYaIngresado = e1 == e2;

    //Assert
    Assert.IsTrue(estudianteYaIngresado);
}

[TestMethod]
0 references
public void InstitutoSobrecargaIgualdad_CuandoElEstudianteEstaEnElInstituto_DeberiaRetonarTrue()
{
    //Arrange
    Instituto instituto = new Instituto();
    Domicilio dom = new Domicilio("Calle Falsa", 123, "Springfield", "Missouri", 1234);
    Estudiante estudiante = new EstudianteParticular("Juan", "Perez", 12345678, DateTime.Now, "Masculino", 1122223333, dom, ETipoE

    //Act
    instituto.Estudiantes.Add(estudiante);
    bool estudianteExistente = instituto == estudiante;

    //Assert
    Assert.IsTrue(estudianteExistente);
}
```

12, 14. Generics, Archivos y Serialización

Serializer - Es una clase que (de)serializa tipos de referencia genéricos. En implementación, el tipo T puede ser List<Estudiante> o string. Al iniciar el programa, si existe un archivo de datos de Estudiantes, se carga y se puede ver directamente en la opción Listado. Si se da de alta, se modifica o se da de baja a un Estudiante, se crea (si no existe) o se sobreescribe el archivo. Los Estudiantes están en un archivo .xml.

A diferencia del TP3, este TP4 no utiliza serialización .json (la información de los cursos se detalla más adelante con SQL). Se agrega un botón Bibliografía en la pestaña Cursos donde se lee un archivo .txt predeterminado con la información de los libros utilizados en cada curso.

Tanto el .xml como el .txt se encuentran en la carpeta **\Casetta.Bianca.2A.TPFinal\TP4\bin\Debug\net5.0-windows**

13. Interfaces

ISubForm - Dado que varios Forms distintos comparten la funcionalidad de instanciar un subForm para mostrar, creé esta interfaz que contiene un método **AbrirSubForm**, que recibe el Form a instanciar como primer parámetro y el Panel donde se mostrará dicho Form como segundo parámetro.

```
/// <summary>
/// A partir de este FrmHome, se llama a otro Form para abrirlo en el Panel correspondiente
/// </summary>
/// <param name="subForm"></param>
/// <param name="panel"></param>
8 references
void ISubForm.AbrirSubForm(Form subForm, Panel panel)
{
    if(subFormMostrado is not null)
    {
        subFormMostrado.Close();
    }

    subFormMostrado = subForm;
    subForm.TopLevel = false;
    subFormMostrado.Dock = DockStyle.Fill;
    panel.Controls.Add(subForm);
    subForm.BringToFront();
    subForm.Show();
}
```

15, 16 y 20. Introducción a SQL, Conexión a Bases de Datos y Métodos de Extensión

- Ejecutar archivo **Instituto_BD.sql** que contiene la query para crear la base de datos antes de ejecutar el programa.

La base de datos cuenta con dos tablas, EstudiantesCurso y Cursos. La primera tiene los campos DNI, Nombre, Apellido y Nivel_Curso. Este último campo es una clave foránea que apunta a la segunda tabla Cursos, donde Nivel es el campo de la clave primaria de esa tabla. Además, los otros campos son Cuota, Horarios y Docente.

Los elementos insertados en Cursos son predeterminados y en ningún momento se van a modificar, por lo que el código dentro de Visual Studio solo va a manipular la tabla EstudiantesCurso a través de los métodos de la clase **BaseDeDatos**.

La clase **BaseDeDatos** y todos sus métodos son estáticos. Para poder extender las funcionalidades de los datos int, string y Estudiante en los métodos correspondientes, se utilizan métodos de extensión:

```
/// <summary>
/// Obtiene la información de la base de datos filtrando la tabla de elementos por Nivel
/// Como la tabla principal es la de Cursos, se muestra toda la informacion de esa tabla y se une con EstudiantesCurso
/// para mostrar tambien los datos basicos de los estudiantes que conforman el curso (Nombre, Apellido, DNI)
/// </summary>
/// <param name="nivel">El nivel a filtrar</param>
/// <returns>Una lista de EstudianteCurso que pertenecen al nivel indicado por parámetro</returns>
1 reference
public static List<EstudianteCurso> LeerPorNivel(this string nivel)
{
```

```
/// <summary>
/// Segun la opcion seleccionada en el cmbCursos, se muestra la informacion del curso seleccionado
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 reference
private void cmbCursos_TextChanged(object sender, EventArgs e)
{
    dgvCursos.AutoGenerateColumns = false;
    dgvCursos.DataSource = cmbCursos.Text.LeerPorNivel();
}
```

17 y 19. Delegados y Eventos

Al instanciar el **FrmDni**, puede que su objetivo sea dar de baja al Estudiante con el que coincida el DNI ingresado o modificar sus datos. Esto se puede distinguir según el texto que aparezca en el **lblTitulo** del Form, entonces a partir de esa información se invoca al delegado **DelegadoBaja** o **DelegadoModificar** según corresponda con sus respectivos métodos asociados para realizar las acciones necesarias.

Dentro de los métodos invocados por cada delegado, se verifica si el DNI ingresado existe antes de proceder, y para esto se utiliza el evento **OnBuscarDni**, que a su vez invoca al método que tiene suscrito, **VerificarDni**.

17 y 18. Expresiones Lambda e Hilos

Al ingresar a Cursos > Bibliografía, se puede visualizar un .txt mostrado en un RichTextBox con la información de los libros que se utilizan para cada Curso. Al mismo tiempo que se instancia el **FrmBibliografía**, se inicia un subproceso en el que se muestran las portadas de los libros de texto en secuencia iterativa infinita al costado del RichTextBox, de manera que se puedan visualizar identificar fácilmente.

Al rotar las portadas, se sigue pudiendo subir y bajar dentro del RichTextBox así como seleccionar los demás botones del sistema para ir a otra opción.

Al crear y correr el Task que genera el hilo, se utiliza una expresión lambda.