

UT2E9. Aplicación de gestión con múltiples entidades.

Construye una aplicación MDI que permita gestionar una colección de películas y libros. En esta ocasión el modelo de datos y la clase encargada de su gestión se facilitan.

La clase película tiene la siguiente estructura

```
public class Pelicula
{
    public int PeliculaId { get; set; }

    public string Titulo { get; set; }

    public int Anno { get; set; }

    public string Genero { get; set; }
}
```

La clase libro tiene la siguiente estructura

```
public class Libro
{
    public int LibroId { get; set; }

    public string Titulo { get; set; }

    public int Anno { get; set; }

    public string Autor { get; set; }
}
```

Por último la clase encargada de la gestión del modelo, denominada “Negocio”, tiene la siguiente estructura. Es conveniente que identifiques la funcionalidad aportada antes de comenzar con la codificación.

```
public class Negocio
{
    private static List<Pelicula> _peliculas;
    private static List<Libro> _libros;
    static Negocio()
    {
        _peliculas = new List<Pelicula>();
        _libros = new List<Libro>();
    }

    public static List<Pelicula> ObtenerPeliculas()
    {
        return _peliculas;
    }

    public static void CrearPelicula(Pelicula nuevaPelicula)
    {
        if (_peliculas.Count > 0)
        {
            nuevaPelicula.PeliculaId = _peliculas.Max(x => x.PeliculaId) + 1;
        }
        else
        {
            nuevaPelicula.PeliculaId = 1;
        }
    }
}
```

```

        _peliculas.Add(nuevaPelicula);
    }

    public static Pelicula ObtenerPelicula(int peliculaId)
    {
        return _peliculas.FirstOrDefault(x => x.PeliculaId == peliculaId);
    }

    public static void BorrarPelicula(int peliculaId)
    {
        var borrar = ObtenerPelicula(peliculaId);
        _peliculas.Remove(borrar);
    }

    public static List<Libro> ObtenerLibros()
    {
        return _libros;
    }

    public static void CrearLibro(Libro nuevoLibro)
    {
        if (_libros.Count > 0)
        {
            nuevoLibro.LibroId = _libros.Max(x => x.LibroId) + 1;
        }
        else
        {
            nuevoLibro.LibroId = 1;
        }
        _libros.Add(nuevoLibro);
    }

    public static Libro ObtenerLibro(int libroId)
    {
        return _libros.FirstOrDefault(x => x.LibroId == libroId);
    }

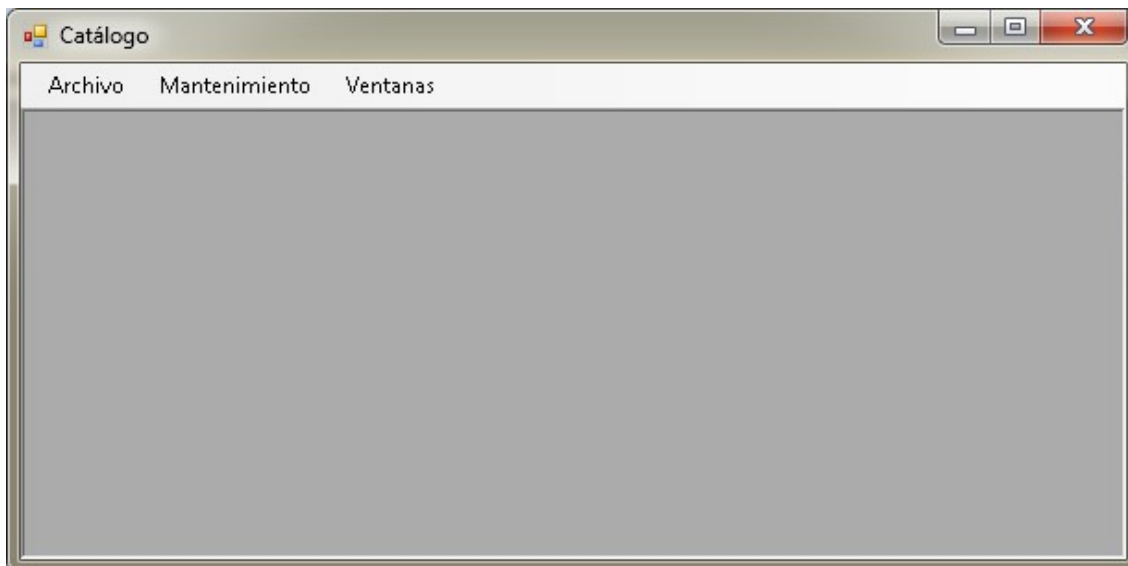
    public static void BorrarLibro(int libroId)
    {
        var borrar = ObtenerLibro(libroId);
        _libros.Remove(borrar);
    }
}

```

La aplicación contará con 5 formularios:

- **Contenedor:** que realizara la función de contenedor MDI y de formulario de entrada.
- **ListaLibroFrm:** mostrará el listado de libros disponibles.
- **LibroFrm:** formulario de propiedades de la clase libro.
- **ListaPeliculaFrm:** mostrará el listado de películas disponibles.
- **PeliculaFrm:** formulario de propiedades de la clase película.

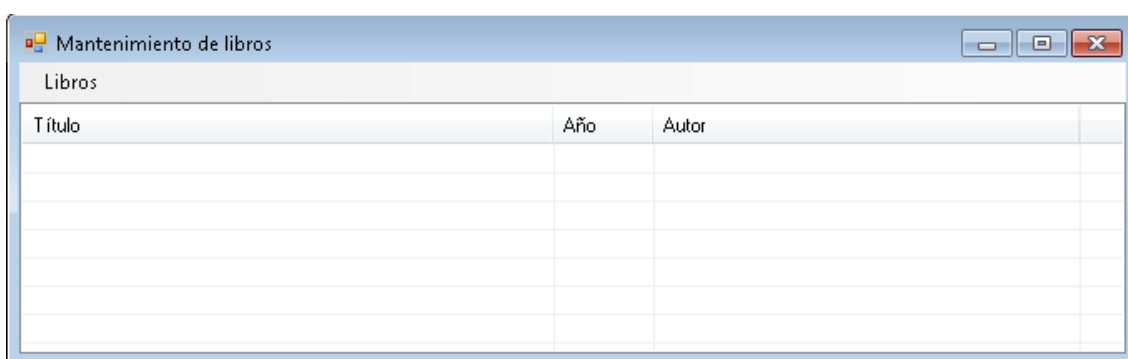
El aspecto del formulario **Contenedor** será el siguiente.



Consideraciones formulario contenedor.

- El menú “Archivo” únicamente tendrá la opción de “Salir”.
- El menú “Mantenimiento” tendrá las opciones “Películas” y “Libros”.
 - La opción “Películas” abrirá el formulario “ListaPeliculaFrm” como formulario hijo de “Contenedor”.
 - La opción “Libros” abrirá el formulario “ListaLibroFrm” como formulario hijo de “Contenedor”.
- El menú “Ventanas” tendrá las opciones “Cascada”, “Horizontal” y “Vertical”. En esta opción de menú se mostrará la lista automática de ventanas abiertas.

El aspecto del formulario ListaLibroFrm será el siguiente.

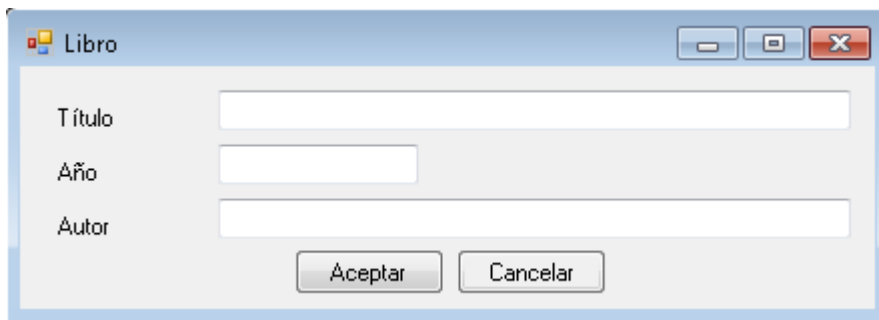


Consideraciones del formulario ListaLibroFrm

- Es un formulario hijo del formulario contenedor
- Los datos de los libros se mostrarán a través de un listview con 3 columnas, título, año y autor.
- Cada fila de la lista almacenara en su propiedad Tag el dato del libro asociado.

- Únicamente se podrá seleccionar un único elemento de la lista a la vez.
- El listview tendrá asociado un menú contextual con las opciones crear, ver y borrar. La opción crear solamente se habilitará cuando no exista ningún elemento seleccionado en la lista, las opciones ver y borrar solamente se habilitarán cuando exista un elemento seleccionado en la lista.
- Las opciones crear, ver y borrar contarán con la iconografía más adecuada para cada caso. Las opciones crear y ver emplearán el formulario “LibroFrm”, la opción borrar solicitará al usuario confirmación a través de un MessageBox.
- Cuenta con un menú principal denominado “Libros” con la única opción “Crear” a través de la cual se podrán crear libros a través del formulario “LibroFrm”.
- Después de las operaciones crear, actualizar y borrar los datos de la lista deberán refrescarse automáticamente.
- Al hacer doble click sobre un libro de la lista se ejecutará la acción de “Ver” sobre el libro seleccionado.

El aspecto del formulario LibroFrm será el siguiente.

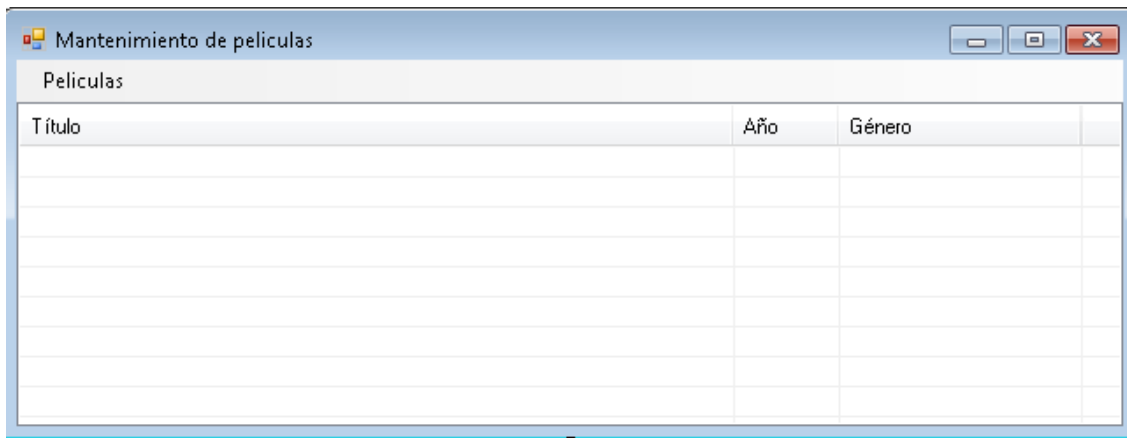


Consideraciones del formulario LibroFrm

- El constructor sin parámetros será privado.
- Existirá un constructor público que reciba como único parámetro un objeto libro.
- El constructor público asignará los valores del objeto a los distintos controles del formulario. Además se establecerá el valor DialogResult.Cancel al formulario.
- Al pulsar el botón aceptar se volcarán los valores de los controles al objeto libro asociado al formulario. Además se establecerá el valor DialogResult.Ok al formulario.
- El tamaño del formulario no podrá ser modificado por el usuario.
- Las propiedades AcceptButton y CancelButton estarán debidamente asignadas a los botones del formulario.

- El formulario se mostrará siempre de forma modal.

El aspecto del formulario ListaPeliculaFrm será el siguiente.

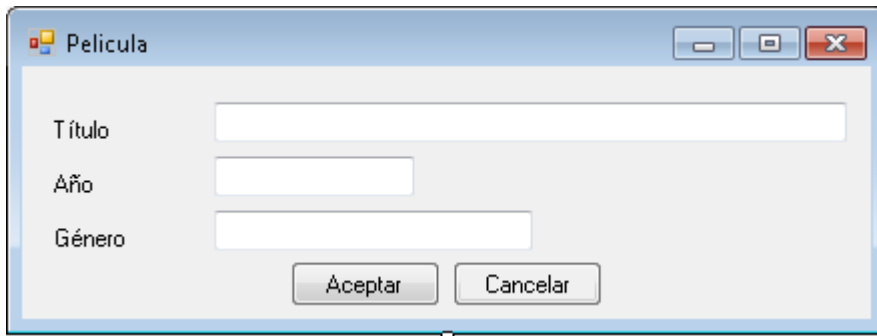


The screenshot shows a Windows-style application window titled "Mantenimiento de películas". Inside the window, there is a section labeled "Películas" containing a listview. The listview has three columns: "Título", "Año", and "Género". The listview is currently empty, showing only the column headers and several empty rows below them.

Consideraciones del formulario ListaPeliculaFrm

- Es un formulario hijo del formulario contenedor
- Los datos de las películas se mostrarán a través de un listview con 3 columnas, título, año y género.
- Cada fila de la lista almacenara en su propiedad Tag el dato de la película asociada.
- Únicamente se podrá seleccionar un único elemento de la lista a la vez.
- El listview tendrá asociado un menú contextual con las opciones crear, ver y borrar. La opción crear solamente se habilitará cuando no exista ningún elemento seleccionado en la lista, las opciones ver y borrar solamente se habilitarán cuando exista un elemento seleccionado en la lista.
- Las opciones crear, ver y borrar contaran con la iconografía más adecuada para cada caso. Las opciones crear y ver emplearan el formulario "PeliculaFrm", la opción borrar solicitara al usuario confirmación a través de un MessageBox.
- Cuenta con un menú principal denominado "Películas" con la única opción "Crear" a través de la cual se podrán crear películas a través del formulario "PeliculaFrm".
- Después de las operaciones crear, actualizar y borrar los datos de la lista deberán refrescarse automáticamente.
- Al hacer doble click sobre una película de la lista se ejecutará la acción de "Ver" sobre la película seleccionada.

El aspecto del formulario PeliculaFrm será el siguiente.



Consideraciones del formulario PeliculaFrm

- El constructor sin parámetros será privado.
- Existirá un constructor público que reciba como único parámetro un objeto película.
- El constructor público asignará los valores del objeto a los distintos controles del formulario. Además se establecerá el valor DialogResult.Cancel al formulario.
- Al pulsar el botón aceptar se volcarán los valores de los controles al objeto película asociado al formulario. Además se establecerá el valor DialogResult.Ok al formulario.
- El tamaño del formulario no podrá ser modificado por el usuario.
- Las propiedades AcceptButton y CancelButton estarán debidamente asignadas a los botones del formulario.
- El formulario se mostrará siempre de forma modal.

NOTA: se deja a elección del alumno la forma de gestionar los formularios de lista libros y lista películas. ¿se podrán abrir múltiples instancias de cada uno o solo una a la vez?, ¿en qué caso debería incluir una opción de menú para refrescar los listados?

Al terminar sube a la plataforma un documento con formato zip y el nombre "UT2E8_NOMBREAPELLIDOS" con el proyecto de código. **RECUERDA limpiar la solución antes de comprimirla.**