Trabajo Practico 4 Laboratorio II

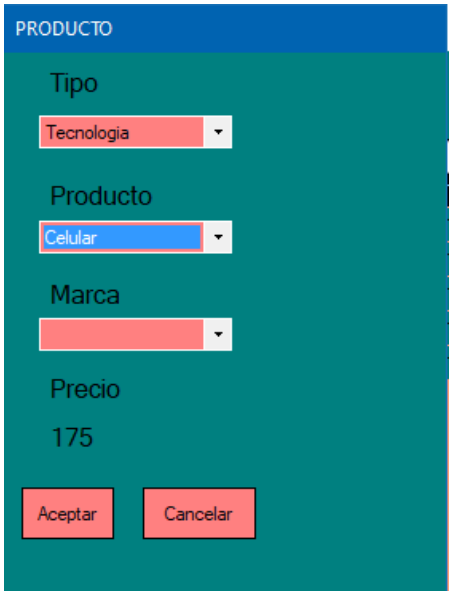
Alumno: Facundo Gabriel Falcioni

El programa consiste en una venta de 2 tipos de productos, contiene una jerarquía de clase simple, teniendo a la clase abstracta Producto como padre, de la cual se desprenden la clase Tecnología y Accesorio. También posee una clase sealed llamada Vendedora la cual almacena en una lista los distintos productos.

El programa contiene 2 form:

- El principal:

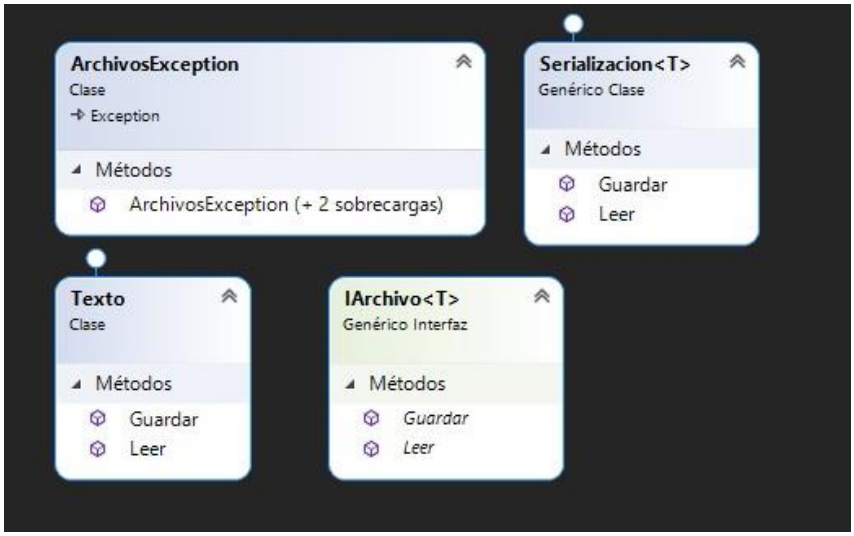
- Y el secundario:



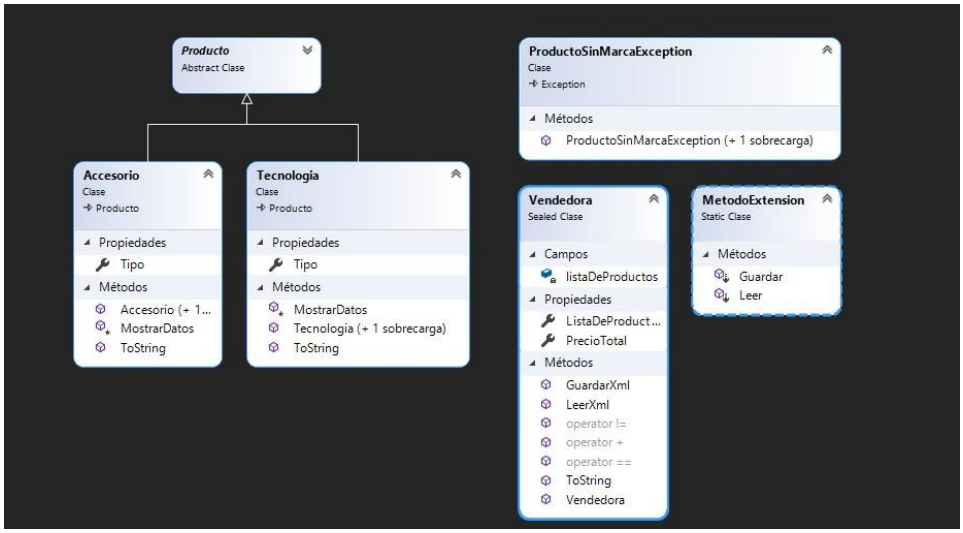
El programa posee 3 proyectos (sin contar los test unitarios y el test por consola), dos bibliotecas de clases y un proyecto de Windows Forms, con la siguiente jerarquía de clases:

Biblioteca de clases:

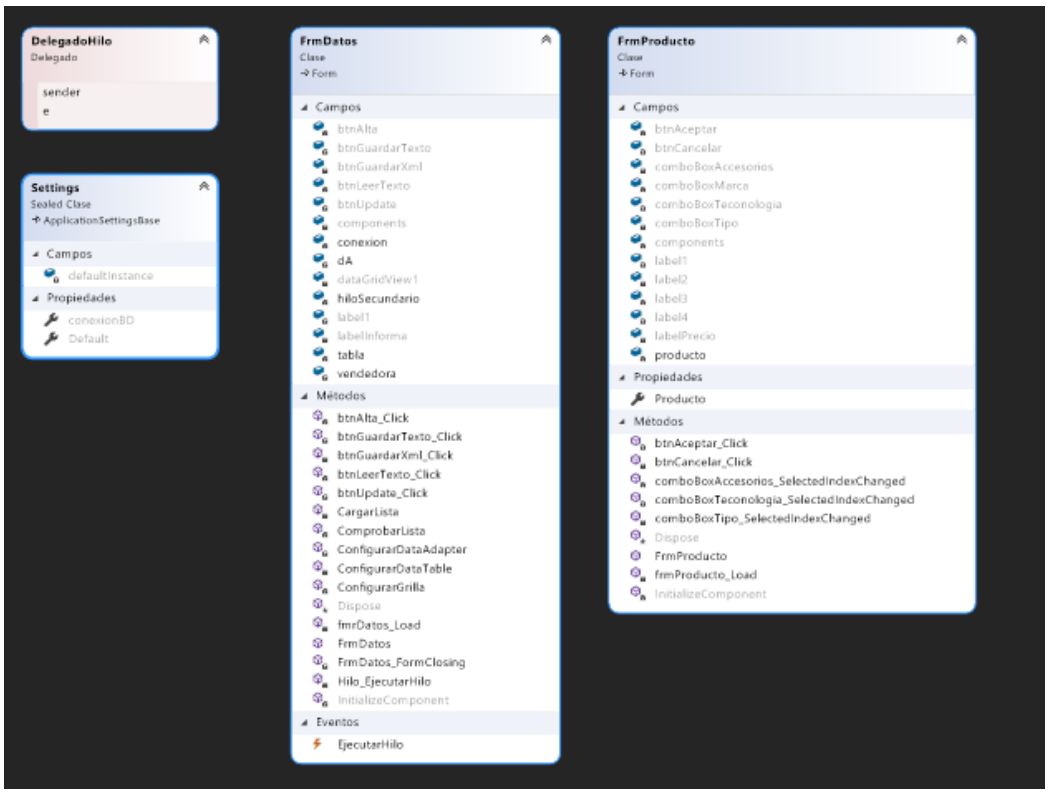
Archivos:



Entidades:



Proyecto Windows Forms:



Funcionalidad:

Al comenzar el formulario se cargan los datos de la base de datos y se hacen visibles en un DataGridView. El formulario que se carga al iniciar el programa permite, realizar un alta de una venta, la cual se va a ver reflejada en el DataGridView y con el botón 'Update' se va a poder sincronizar los datos con su respectiva base de datos. Ademas permite la escritura y lectura de un archivo .txt y la serialización de los datos a un archivo .xml. Al iniciarse el programa se lanza un evento llamado 'EjecutarHilo' el cual es de tipo 'DelegadoHilo' que devuelve void y recibe como parámetros un object y un EventArgs. Dicho evento tiene asociado el método 'Hilo_EjecutarHilo' el cual se encarga de ejecutar el hilo secundario.

El hilo secundario comprueba cada 2 segundos que la lista de ventas del programa sea igual a los elementos que están contenidos en el DataGridView, para que de esta forma al guardar en un .txt o serializar la lista, estén contenidos en esta los mismos datos, sin que el usuario tenga que estar pendiente constantemente de actualizarla antes de querer realizar una operación. Esto es posible mediante el método 'ComprobarLista', este comprueba si la cantidad de filas del DataGridView es distinta a la cantidad de elementos de la lista de productos. De ser iguales informa que los datos ya están sincronizados. En caso de no serlo los agrega a la lista e informa que la lista fue actualizada.

El form secundario permite agregar un producto al datagridview, seleccionando que tipo de producto es, que marca, etc.

Ademas, al intentar cerrar el form se pregunta al usuario si confirma la salida, ya que si se olvida de realizar el 'UPDATE' perderá todos los nuevos datos ingresados.

La clase vendedora contiene 2 métodos de extensión los cuales son creados en la clase estática 'MetodosExtension'. Estos métodos son los que permiten guardar y leer una clase de tipo Vendedora en un archivo .txt

La interfaz 'IArchivo' tiene como métodos Guardar y Leer y es genérica, esta está implementada en las clases 'Serializacion', que también es genérica, y en la clase texto.

Se reutiliza código y se aplican excepciones en todos los lugares necesarios.

En resumen:

Métodos de extensión: Se encuentran en la clase 'MetodoExtension' **Excepciones**: Se aplican a lo largo del programa y también se crea una excepción del tipo 'ArchivosException, la cual es utilizada en el manejo de archivos, tanto de serialización como con los archivos de .txt. La misma informa en caso de ocurrir un error. También se crea una excepción llamada 'ProductoSinMarcaException' la cual es lanzada si se intenta cargar un producto sin marca.

Interfaces, archivos y tipos genéricos: Se aplica en la interfaz 'IArchivo' y en las clases 'Serializacion' y 'Texto'

Delegados y eventos: Se crea por fuera del frmDatos el delegado 'DelegadoHilo' que retorna void y recibe como parámetros un object y un eventoArgs. Y dentro del formulario se crea el evento 'EjecutarHilo' el cual es de tipo DelegadoHilo, a dicho evento se le asocia un método y se lanza el mismo al iniciar el programa y al cerrarlo.

Hilos: Se inicia mediante el evento 'EjecutarHilo' al inicio del programa, se encarga de mantener la lista de productos actualizada con el DataGridView, esto lo realiza mediante el evento 'ComprobarLista'. Al cerrar el programa mediante el manejador de eventos FormClosing del frmDatos se comprueba si el hilo secundario está vivo, de ser así lanzo el evento 'EjecutarHilo' quien se encarga de abortar el hilo, para que el mismo no se siga ejecutando

Base de datos: Conecta mediante un DataTable y un DataAdapter la aplicación con una base de datos. Su implementación se encuentra en la clase frmDatos.

NOTA: Si bien los productos que se agregan a la base de datos pueden tener los mismos datos, ya sea tipo, marca, producto, precio etc. Cada uno posee un ID único. Es decir, se pueden agregar dos productos de tipo Tecnología, Notebook Asus con un precio de 520, pero eso no quiere decir que sean iguales ya que estos tienen ID distintos.