Universidad Tecnológica Nacional
Facultad Regional Avellaneda

T.P. Nº4 de Laboratorio de Programación II

Alumno: Leandro Noguera

Curso: 2D

El programa realiza la carga de 2 tipos de productos (perecedero y no perecedero) los cuales se cargan en una base de datos. Por otra parte, cuenta con la funcionalidad de realizar ventas. La clase estática Inventario se encarga de gestionar la carga de productos a la base de datos y cuenta con una lista que guarda las ventas que se van realizando.

Clase 15 – Excepciones

El proyecto Excepciones contiene 3 tipos de excepciones propias:

- ProductosException: Se lanza en caso de no poder cargar un producto a la base de datos o eliminarlos. Se dispara en la sobrecarga de operadores "+" y "-" de la clase Producto del proyecto Entidades.
- VentasException: Se lanza en caso de no poder agregar un producto a la venta o en caso de que la venta ya se encuentre cerrada. Se dispara en la sobrecarga de operadores "+" y "-" de la clase Venta del proyecto Entidades.
- ArchivosException: Se lanza en caso de error en la lectura/escritura de archivos. Se dispara en los métodos EjecutarNonQuery, Leer y LeerPorID de la clase ProductoDAO del proyecto Entidades. En los métodos Guardar y Leer de la clase Texto, y en Guardar y Leer de la clase Xml, ambas del proyecto Archivos

Clase 16 – Test unitarios

El proyecto TestUnitario verifica que se lance la excepción ProductosException al intentar cargar 2 veces el mismo producto a la base de datos. También verifica que se instancie la lista de productos al instanciar un objeto de tipo Venta.

Clase 17 – Tipos Genéricos

Se utiliza en la interfaz IArchivos sita en el proyecto Archivos. Ésta determina que las clases que la apliquen, creen métodos genéricos de lectura y escritura de archivos.

Clase 18 – Interfaces

Se creó la interfaz IArchivos en el proyecto Archivos. Es implementada por las clases Texto y Xml pertenecientes al mismo proyecto.

Clase 19 – Archivos y Serialización

El método estático PrintTicket de la clase Venta (proyecto Entidades) genera un archivo de texto que simula el ticket de compra. El método estático Leer de la misma clase trae el ticket en función de la ruta que se le pasa por parámetro. El ticket se genera al agregar una venta a un listado de ventas (a través de la sobrecarga del operador +). La sobrecarga está hecha en la clase Venta.

El método estático Guardar de la clase Inventario (proyecto Entidades), serializa un listado de ventas en un archivo XML. El método Leer, lo deserializa. El proyecto archivos, gestiona todas estas ejecuciones.

Clase 21 / 22 – SQL / Base de datos

La clase ProductoDAO (proyecto Entidades) se encarga de administrar la base de datos. Cada vez que se carga un producto, este es guardado automáticamente en la base de datos. Cada vez que se agrega un producto a una venta, se descuenta el artículo de la cantidad de productos disponibles en la base de datos. También cuenta con métodos para modificar y eliminar artículos.

Script:

```
USE [master]
GO
/****** Object:  Database [TPNro4]    Script Date: 15/11/2020 17:57:40 ******/
CREATE DATABASE [TPNro4]
 CONTAINMENT = NONE
 ON  PRIMARY
( NAME = N'TPNro4', FILENAME = N'C:\Users\CX
SLIM\Documents\SQL\MSSQL15.SQLEXPRESS\MSSQL\DATA\TPNro4.mdf' , SIZE = 8192KB , MAXSIZE = UNLIMITED,
FILEGROWTH = 65536KB )
 LOG ON
( NAME = N'TPNro4_log', FILENAME = N'C:\Users\CX
SLIM\Documents\SQL\MSSQL15.SQLEXPRESS\MSSQL\DATA\TPNro4_log.ldf' , SIZE = 8192KB , MAXSIZE = 2048GB ,
FILEGROWTH = 65536KB )
 WITH CATALOG_COLLATION = DATABASE_DEFAULT
GO
ALTER DATABASE [TPNro4] SET COMPATIBILITY_LEVEL = 150
GO
IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))
begin
EXEC [TPNro4].[dbo].[sp_fulltext_database] @action = 'enable'
end
GO
ALTER DATABASE [TPNro4] SET ANSI_NULL_DEFAULT OFF
GO
ALTER DATABASE [TPNro4] SET ANSI_NULLS OFF
GO
ALTER DATABASE [TPNro4] SET ANSI_PADDING OFF
GO
ALTER DATABASE [TPNro4] SET ANSI_WARNINGS OFF
GO
ALTER DATABASE [TPNro4] SET ARITHABORT OFF
GO
ALTER DATABASE [TPNro4] SET AUTO_CLOSE OFF
GO
ALTER DATABASE [TPNro4] SET AUTO_SHRINK OFF
GO
ALTER DATABASE [TPNro4] SET AUTO_UPDATE_STATISTICS ON
GO
ALTER DATABASE [TPNro4] SET CURSOR_CLOSE_ON_COMMIT OFF
GO
ALTER DATABASE [TPNro4] SET CURSOR_DEFAULT  GLOBAL
GO
ALTER DATABASE [TPNro4] SET CONCAT_NULL_YIELDS_NULL OFF
GO
ALTER DATABASE [TPNro4] SET NUMERIC_ROUNDABORT OFF
GO
ALTER DATABASE [TPNro4] SET QUOTED_IDENTIFIER OFF
GO
ALTER DATABASE [TPNro4] SET RECURSIVE_TRIGGERS OFF
GO
ALTER DATABASE [TPNro4] SET  DISABLE_BROKER
GO
ALTER DATABASE [TPNro4] SET AUTO_UPDATE_STATISTICS_ASYNC OFF
GO
```

```
ALTER DATABASE [TPNro4] SET DATE_CORRELATION_OPTIMIZATION OFF
GO
ALTER DATABASE [TPNro4] SET TRUSTWORTHY OFF
GO
ALTER DATABASE [TPNro4] SET ALLOW_SNAPSHOT_ISOLATION OFF
GO
ALTER DATABASE [TPNro4] SET PARAMETERIZATION SIMPLE
GO
ALTER DATABASE [TPNro4] SET READ_COMMITTED_SNAPSHOT OFF
GO
ALTER DATABASE [TPNro4] SET HONOR_BROKER_PRIORITY OFF
GO
ALTER DATABASE [TPNro4] SET RECOVERY SIMPLE
GO
ALTER DATABASE [TPNro4] SET  MULTI_USER
GO
ALTER DATABASE [TPNro4] SET PAGE_VERIFY CHECKSUM
GO
ALTER DATABASE [TPNro4] SET DB_CHAINING OFF
GO
ALTER DATABASE [TPNro4] SET FILESTREAM( NON_TRANSACTED_ACCESS = OFF )
GO
ALTER DATABASE [TPNro4] SET TARGET_RECOVERY_TIME = 60 SECONDS
GO
ALTER DATABASE [TPNro4] SET DELAYED_DURABILITY = DISABLED
GO
ALTER DATABASE [TPNro4] SET ACCELERATED_DATABASE_RECOVERY = OFF
GO
ALTER DATABASE [TPNro4] SET QUERY_STORE = OFF
GO
USE [TPNro4]
GO
/****** Object:  Table [dbo].[Productos]    Script Date: 15/11/2020 17:57:40 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Productos](
	[descripcion] [nvarchar](50) NOT NULL,
	[idProducto] [numeric](18, 0) NOT NULL,
	[precio] [float] NOT NULL,
	[cantidad] [numeric](18, 0) NOT NULL,
	[tipoProducto] [nvarchar](50) NOT NULL
) ON [PRIMARY]
GO
USE [master]
GO
ALTER DATABASE [TPNro4] SET  READ_WRITE
GO
```

Clase 23 – Hilos

Los métodos estáticos PuntoVenta1 y PuntoVenta2 de la clase Inventario (proyecto Entidades) se encargan de crear dos hilos diferentes desde los cuales se efectúan ventas, simulando 2 puntos de venta trabajando de forma simultánea.

Clase 24 – Eventos

Se declara el delegado "`public delegate bool DelegadoVentas(Venta venta)`". Se instancia un evento estático de dicho tipo de delegado en la clase Venta y se dispara al cerrar una venta (se agrega a una lista de tipo Venta a través de la sobrecarga del operador +). Este evento nuclea los siguientes métodos estáticos, que se ejecutan al momento del cierre de la venta:

CalcularMontoTotal (Clase Venta): recorre la lista de artículos y hace una sumatoria de todos sus valores y los carga en el atributo montoTotal del objeto de tipo Venta.

ModificarStock(Clase Inventario): Recorre la base de datos y modifica el stock disponible de los artículos que se agregaron al objeto de tipo Venta.
PrintTicket(Clase Venta): Imprime el ticket de la venta en cuestión.
CargarVenta(Clase inventario): Finalmente, agrega la venta al listado de la clase inventario.

Clase 25 – Métodos de extensión

En el proyecto entidades se creó la clase estática “ListExtension”. Dicha clase está contenida dentro del Namespace “`System.Collections.Generic`” y agrega las siguientes funcionalidades:

Método: `public static int StockTotal(this List<Producto> listaProductos)`

Recorre una lista de tipo Producto y realiza la sumatoria de los atributos cantidad de los objetos contenidos en ésta. Devuelve un elemento de tipo int que representa el stock total.

Método: `public static double TotalVentas(this List<Venta> listaVentas)`

Recorre una lista de tipo Venta y realiza la sumatoria de los atributos montoTotal de los objetos contenidos en ésta. Devuelve un elemento de tipo double que representa el valor acumulado de ventas.