Bastardo Valentino - TP3 - Division 2A

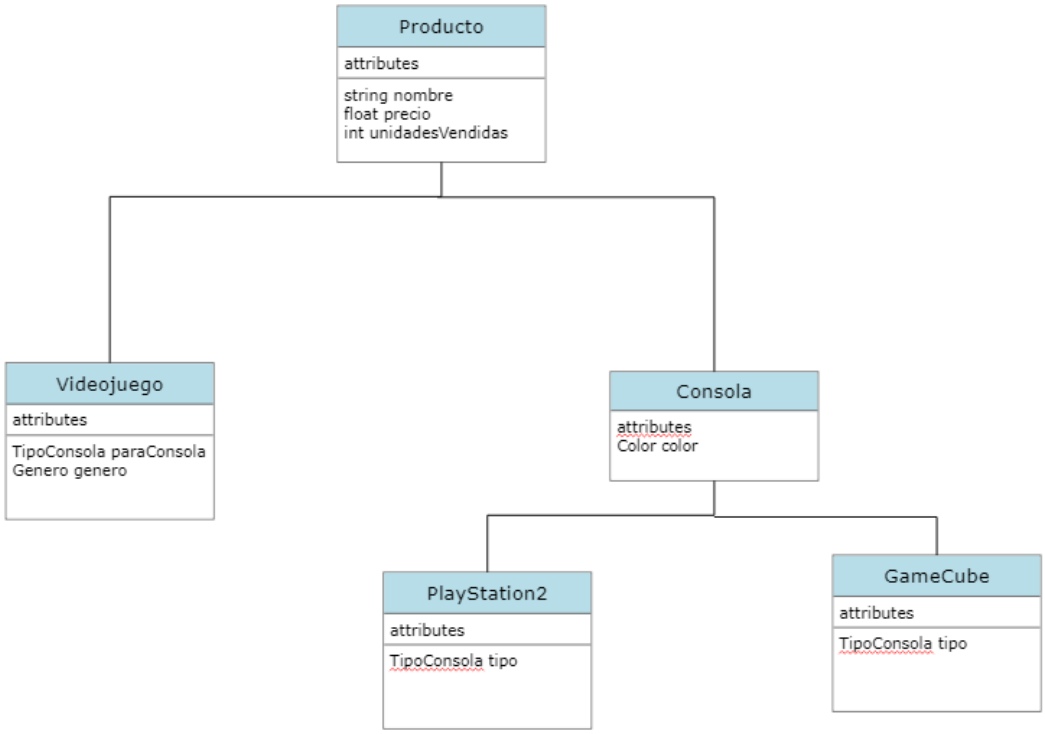
Objetivo del TP:

Desarrollar un sistema de atencion al publico

El contexto de mi empresa es un local donde se venden videojuegos y consolas, en donde se puede gestionar un inventario de productos, una lista de clientes, y en el futuro podra manejar pedidos.

La clase Inventario cuenta con un atributo de tipo

List<Producto> en el cual cargará, modificara, agregará y guardará productos, los cuales pueden ser por un lado videojuegos, o por otro lado consolas, estando estas divididas entre GameCube y PlayStation 2.

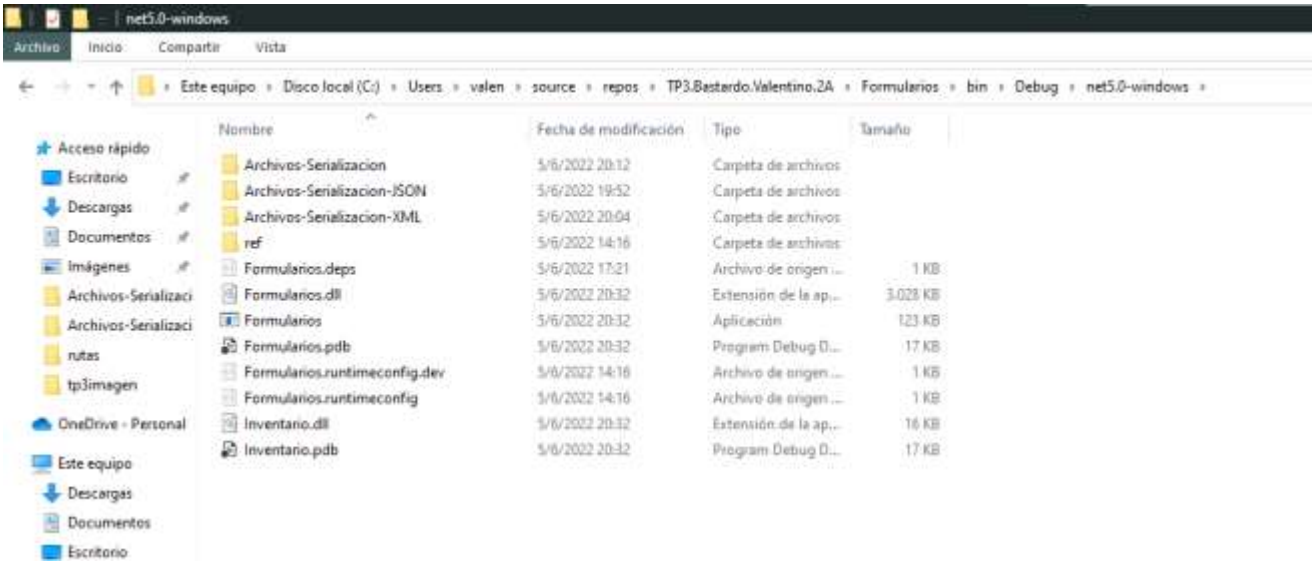


El programa es capaz de gestionar un inventario de productos, pudiendo cargar mediante XML o cargarlos a mano especificando su tipo, precio, unidades vendidas (para calcular ganancias) y luego guardarlo en xml, o imprimirlo en un .txt)

Tambien maneja una lista de clientes, la cual utiliza JSON para ser importada y exportada.

TODOS LOS ARCHIVOS SE GENERAN EN LA CARPETA DEL PROYECTO

Formularios>Bin>Debug>net5.0-windows

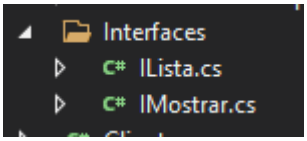


TEMAS APLICADOS:

Archivos fue aplicado pudiendo exportar un .txt con la lista de productos utilizando el boton del menu.

Serializacion fue aplicado utilizando JSON para importar y exportar la lista de clientes desde el menu y XML para importar y exportar una lista de productos.

Interfaces se utilizo para los Mostrar() de las clases y para retornar listas y mostrarlas por el FormMostrar.



en las mismas tambien se implementó **generics**:

```
namespace InventarioNS
{
    2 referencias
    public interface IListar<T> where T: class
    {
        /// <summary>
        /// interfaz para el manejo de listas
        /// </summary>
        7 referencias | 1/1 pasando
        List<T> Lista
```

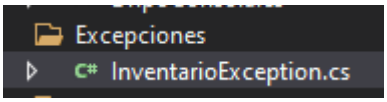
```
    private List<T> lista;
    1 referencia
    private FormMostrar()
    {
        InitializeComponent();
    }
    2 referencias
    public FormMostrar(List<T> lista) : this()
    {
        this.lista = lista;
    }
```

y los tipos genericos tambien son utilizados por los serializadores de archivos:

```
namespace InventarioNS
{
    3 referencias
    public static class SerializadorXML<T>
    {
```

```
namespace InventarioNS
{
    3 referencias
    public static class SerializadorJSON<T>
    {
```

Se creó una **excepcion** de Inventario para cuando se intente agregar un producto que ya este dentro de la lista




```
public static Inventario operator +(Inventario i, Producto nuevoProd)
{

    if (i != nuevoProd)
    {
        i.AgregarProducto(nuevoProd);
    }
    else
    {
        throw new InventarioException("Ese producto ya se encuentra en el inve
    }
    return i;

}
```

Tambien se implementaron 3 pruebas unitarias para testear funciones de Inventario. Se agregaran mas para el tp4.

```
namespace TestUnitario
{
    [TestClass]
    0 referencias
    public class InventarioTest
    {
        [TestMethod]
        0 referencias
        public void Inventario_InstanciaLaListaAlSerCreado()
        {
            // arrange
            Inventario inv;
            // act
            inv = new Inventario("Prueba");
            // arrange
            Assert.IsNotNull(inv.Lista);
        }
```

Por el momento el programa no puede "atender" clientes, es decir, recibir un pedido de un cliente, y concretar una venta en donde se calcula la ganancia y se elimina el pedido, pero para el tp4 desarrollaré bien esta función, junto con un formulario un poco mas vistoso y amigable para el usuario, ya que ahora mismo esta pensado unicamente para testear la biblioteca de clases.