

# Trabajo Práctico N°4

Esta aplicación simulará un sistema de fabricación de dispositivos móviles.

The screenshot shows a Windows Forms application window titled "Lorenzo Nahuel Cea Ko, 2A". The window contains a form for manufacturing mobile devices. The form has several input fields: "Producto" (Celular, Tablet, SmartWatch), "Modelo" (miCelular), "Tamaño" (Chico, Mediano, Grande), "Procesador" (Snapdragon), "RAM" (3), "ROM" (32), and "Cámara" (13). There are three status boxes: "En espera" (empty), "Fabricando" (empty), and "Finalizado" (Celular, miCelular, Grande, Snapdragon, 3GB, 32GB, 13mpx). A "Fabricar" button is at the bottom left.

Mediante esta ventana de Windows Forms podremos cargar al sistema distintos dispositivos con características específicas y prepararlos para su fabricación.

En primer lugar y para su correcto funcionamiento, necesitaremos de la base de datos llamada [Cea.Lorenzo.2A] y también una tabla llamada Productos. Los archivos correspondientes están incluidos en la solución para que sean utilizados con este fin, en el zip **database**. En caso de que la tabla no este presente, el siguiente script generará la tabla dentro de la base [Cea.Lorenzo.2A]:

```
USE [Cea.Lorenzo.2A]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Productos](
  [Id] [int] IDENTITY(1,1) NOT NULL,
  [Modelo] [varchar](50) NOT NULL,
  [Ram] [int] NOT NULL,
  [Rom] [int] NOT NULL,
  [Tamaño] [varchar](50) NOT NULL,
  [Procesador] [varchar](50) NOT NULL,
  [Camara] [int] NOT NULL,
  [Tipo] [varchar](20) NOT NULL
) ON [PRIMARY]
GO
```

También hay que asegurarse que la aplicación se conecte correctamente a la base. En la clase Fabrica podemos encontrar la connectionString seteada como @"Data Source=.;Initial Catalog=Cea.Lorenzo.2A;Integrated Security=true;", modifíquese de ser necesario (si el formulario

no inicia al momento, luego de unos segundos debería levantar un MessageBox indicando el cambio a realizar)

### Campos:

#### **Producto** (lblProducto & lbxProducto)

Listbox para seleccionar entre los 3 tipos de producto que se fabrican.

Debe seleccionarse uno para empezar a setear las demás propiedades del Producto.

#### **Tamaño** (lblTamano & lbxTamano)

Configura el ETamano del Celular (Tablet será Grande por defecto, y SmartWatch Chico por defecto)

#### **Procesador** (lblProcesador & cbxMarca)

Setea la EMarca del Producto (si se ingresa un texto distinto de las opciones del Combobox, setea EMarca.Generico)

#### **Modelo** (lblModelo & txtModelo)

Setea el modelo del Producto según lo que se ingrese en la caja de texto

#### **RAM y ROM** (lblRam & cbxRam, lblRom & cbxRom)

Setea la memoria RAM y ROM que tendrá el Producto. (*Detalle: valida que no se ingresen caracteres no numéricos, y también impide ingresar un número mayor a 16 y 256 respectivamente*)

#### **Cámara** (lblCamara & cbxCamara)

Setea los megapixeles del Producto. Al igual que con RAM y ROM, se valida el dato ingresado, e igualmente impide ingresar un número mayor a 64

### Botones:

#### **Agregar**

Toma todos los datos ingresados en el formulario, genera un nuevo Producto a la Fabrica y la lista en el Listbox de la derecha. Es en este momento que se hacen las validaciones de los datos ingresados.

#### **Remover**

Remueve el item seleccionado en el ListBox de la derecha (lbxFabrica). Si no hay ítems para remover se advertirá al usuario al momento de presionar el botón.

#### **Fabricar**

Exporta la lista de Productos en un archivo de texto y un XML. Los mismos se contendrán en una carpeta en el Escritorio del usuario.

Detalle: Si se intenta Fabricar solamente un Producto, la aplicación lo impedirá y pedirá al usuario agregar un Producto más. Esto es teniendo en cuenta el punto 9.a. de las Condiciones de corrección y aprobación.

## Excepciones

Presentes en el proyecto de tipo *Biblioteca de Clases* **Excepciones**.

### **RemoverObjetoException:**

Excepcion hecha para situaciones en que se haga referencia a un objeto que no se encuentra en una lista determinada.

### **ValorInvalidoException:**

Excepcion hecha para el caso en que se vaya a ingresar un dato inválido (como por ejemplo un caracter para un campo numérico) o un dato vacío

## Test Unitarios

Presentes en el proyecto de tipo *Tests Unitarios* **Unit.Tests**

### DebeAgregarUnProducto:

Test unitario para Fabrica.**Agregar()**, el operador +, Fabrica.ToString() y Producto.ToString()

### DebeRemoverUnProducto:

Test unitario para Fabrica.**Remover()**

### DebeLimpiarLaLista:

Test unitario para Fabrica.**Limpiar()**

### DebeImpedirAgregarDuplicados:

Test unitario para probar que arroje **AgregarObjetoException**

### DebeImpedirRemoverUnObjetoNoExistente:

Test unitario para probar que arroje **RemoverObjetoException**

### DebeValidarDatosIngresadosAlProducto:

Test unitario para probar que arroje **ValorInvalidoException**

### DebeGenerarYLeerTexto:

Test unitario para Fabrica.**GuardarComoTexto(string archivo)** y Fabrica.**LeerArchivoTexto(string archivo)**

#### DebeGenerarYLeerXml:

Test unitario para Fabrica.**GuardarComoXml(string archivo)** y Fabrica.**LeerArchivoXml(string archivo)**

#### TestConnectionString:

Test unitario para probar la conexión a la base.

#### DebeInsertarUnRegistroEnLaBase:

Test unitario para probar que se inserte un registro exitosamente.

## Tipos Genéricos

### **Fabrica<T>**

Es la clase que va a contener la información de nuestra aplicación. Se usa de manera genérica con vistas a futuro para otros usos fuera del TP.

Disponible en el proyecto de tipo *Biblioteca de Clases* **Entidades**

**Xml<T>** (*descripción disponible en la sección [Archivos y serialización](#)*)

## Interfaces

**IArchivos<T>** (*descripción disponible en la sección [Archivos y serialización](#)*)

## Archivos y serialización

Se encuentran contenidas en el proyecto de tipo *Biblioteca de Clases* **Archivos**.

### **IArchivos<T>**

Interface que declara los métodos Guardar y Leer, apuntado a archivos que puedan ser consumidos por la aplicación. Asimismo, hace uso de tipos genéricos.

Se implementa en las clases Xml y Texto.

### **Xml<T>**

Es la clase que va a manejar el guardado y lectura de archivos XML. Se usa de manera genérica para que sus métodos sean compatibles con cualquier clase que se implemente.

### **Texto**

Es la clase que va a manejar el guardado y lectura de archivos de texto. En el sentido del TP, va a permitir guardar la información de la lista de Productos en un archivo de texto.

## SQL y Bases de datos

Instrucciones para uso de la base de datos al principio del documento.

Se implementaron dos métodos en Fabrica para manipular la base:

```
public static bool GuardarEnLaBase(Producto producto)
```

Guarda los Productos de la Fabrica en la base de datos

```
public static Fabrica LeerDeLaBase()
```

Hace una lectura de la base de datos y recupera la información de la misma

Además se realizó el siguiente método (también en la clase **Fabrica**) para testear al inicio del formulario que la connectionString sea correcta:

```
public static bool TestConnectionString()
```

## Hilos

Se implementó el uso de Hilos en el formulario Inicio, en el evento Click del botón btnAgregar (btnAgregar\_Click), asociado al método de instancia CambiarStatus de la clase Fabrica.

## Eventos

Se creó el Delegado `void CambioStatus(object objeto)` en el documento **Delegados.cs**, y asimismo se implementó el evento `event CambioStatus CambiarStatusEvent` en la clase **Fabrica**, que se disparará al ejecutarse el método `CambiarStatus(object producto)`.

Como se mencionó antes, éste último método se ejecutará en un hilo del WinForm **Inicio**, y luego el evento que dispara será capturado por el manejador `CambiarEstado(object producto)`, que pasará el Producto generado, por cada proceso de fabricación. Una vez finalizado, generará el Xml de ese objeto y también guardará un registro en la base de datos.

## Métodos de extensión

Presente en el documento MetodosDeExtension.cs

```
public static EMarca ToEMarca(this string cadena)
```

Este método de extensión se encarga de transformar un string de origen para devolver su equivalente EMarca, o en su defecto EMarca.Generico