Dpto. Software Nea F3 Master S.L.

Manual de Usuario para utilización de la programación del DLL CamposAdicionales

**Campos Adicionales**

Manual de Usuario

[**1.** **Introducción** 2](#_Toc180398154)

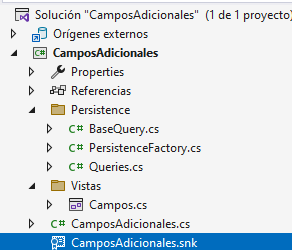
[**2.** **CamposAdicionales** 3](#_Toc180398155)

[**3.** **Queries** 5](#_Toc180398156)

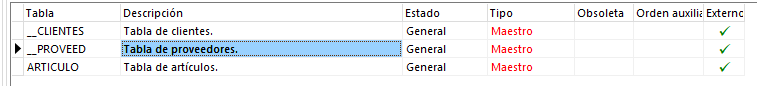
[**4.** **Campos** 8](#_Toc180398157)

# **Introducción**

La programación se compone de un DLL principal llamado *CamposAdicional.cs* que a su vez contiene un Windows Form: *Campos.cs*.

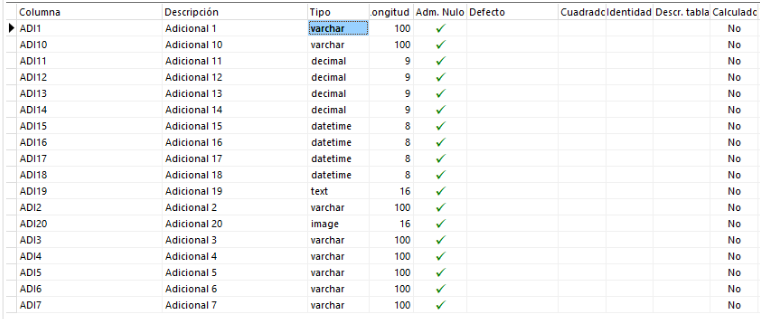


Este programa se encarga de permitir que los usuarios puedan guardar valores adicionales en 20 campos nuevos que se han añadido en un diccionario en las tablas de **Clientes** (\_\_CLIENTES), **Proveedores** (\_\_PROVEED) y **Artículos** (ARTICULO):



Dichos campos son los siguientes:

* **ADI1, ADI2, ADI3, ADI4, ADI5, ADI6, ADI7, ADI8, ADI9** y **ADI10** para valores de tipo **VARCHAR**.
* **ADI11, ADI12, ADI13** y **ADI14** para valores del tipo **DECIMAL**.
* **ADI15, ADI16, ADI17** y **ADI18** para valores del tipo **DATETIME**.
* **ADI19** para valores del tipo **TEXT**.
* **ADI20** para valores del tipo **IMAGE**.



# **CamposAdicionales**

Esta clase se encarga de gestionar la interacción con la base de datos y proporcionar funcionalidades adicionales a a3ERP. Sus principales funcionalidades son:

* **Conexión a la Base de Datos**: Mediante la string *conexion*.

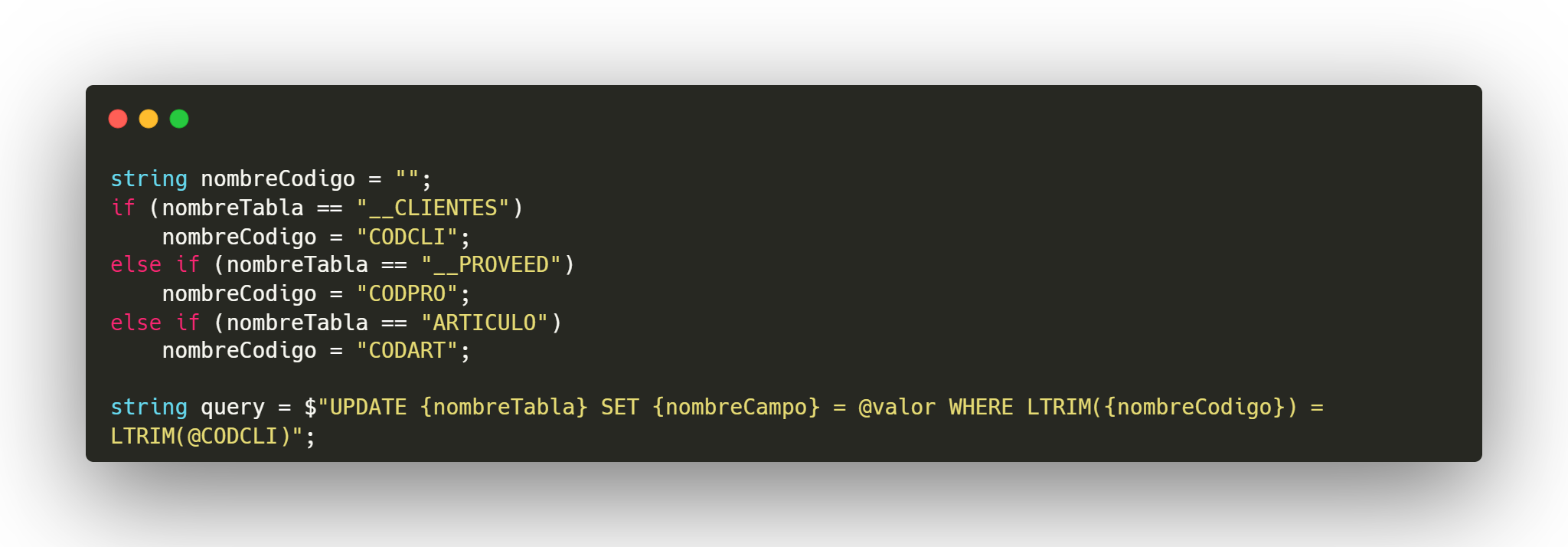


* **Integración con a3ERP:** Mediante el objeto Enlace establece la comunicación con el ERP
* **Gestión de los Windows Forms.**
* **Eventos de documentos:** Implementa métodos para manejar eventos específicos relacionados con documentos, como:
  + *AntesDeGuardarDocumentoV2*: Prepara datos antes de guardar documentos, estableciendo valores en los formularios adicionales.
  + *DespuesDeGuardarDocumentoV2*: Configura el documento después de ser guardado.
  + *DespuesDeCargarDocumentoV2*: Muestra formularios y botones adicionales basados en el tipo de documento cargado.
* **Manejo de la creación del Windows Form:** Cuando se llama al método **Opcion(string IdOpcion, double idDoc)** si IdOpcion es CLIENTES o ARTICULOS o PROVEEDORES, se creará el formulario y se mostrará:

# **Queries**

Esta clase se encarga de actualizar campos específicos de las tablas de clientes, proveedores y artículos de la base de datos. Todo ello lo hace mediante el método **ActualizarCampo()** que recibe como parámetros el nombre del campo a modificar (ADI1, ADI2…ADI20), el valor que se quiere guardar en dicho campo, el código y el nombre de la tabla.

Primero, se obtiene el nombre del código (CODCLI, CODPRO o CODART) mediante el nombre de la tabla y se declara la *query* con los parámetros sin asignar aún:



A continuación, añado los parámetros a la *query* y la ejecuto en función del tipo de dato que es. Si se quiere guardar un DATETIME:



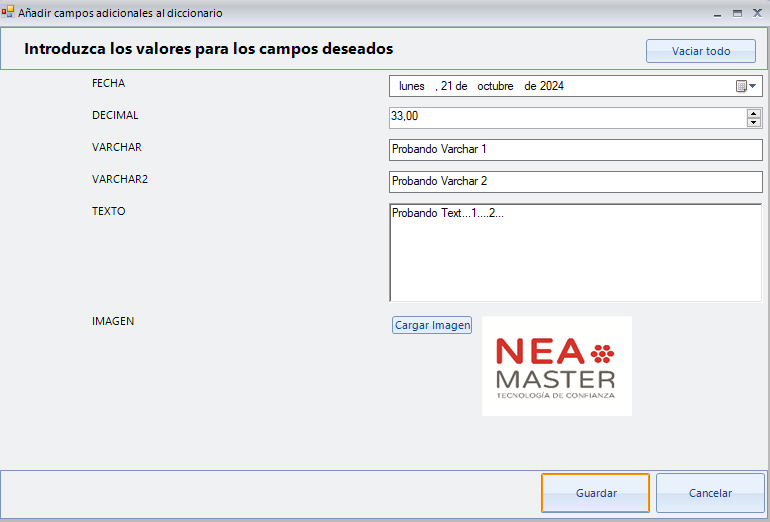
Si se quiere guardar un DECIMAL o IMAGE:



Y finalmente, si se quiere guardar un VARCHAR o TEXT:



# **Campos**

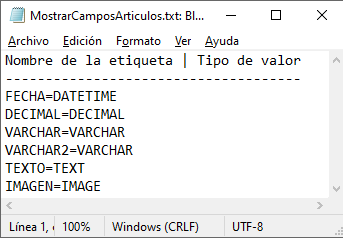


El formulario *Campos* está formado por un tableLayoutPanel donde se añaden los elementos a mostrar: **Label** para el nombre de cada campo, **NumericUpDown** para los campos de tipo DECIMAL, **TextBox** para los campos de tipo VARCHAR, **DateTimePicker** para los campos de tipo DATETIME, **RichTextBox** para los campos de tipo TEXT y **FlowLayoutPanel** para los campos de tipo IMAGE.

Primero, se cargan los elementos en el formulario con el método de **CargarCampos()**, que lee el fichero .txt correspondiente (*MostrarCamposArticulos.txt* para artículos, *MostrarCamposClientes.txt* para clientes y *MostrarCamposProveedores.txt* para proveedores) mediante el método de **leerFichero()**:



Por otro lado, en el archivo .txt se indican los campos a mostrar en la ventana con el nombre del campo y el tipo del campo separados por un símbolo de “=”:



Después, verifica que no se sobrepase la cantidad de elementos de cierto tipo en el .txt (por ejemplo, solo puede haber 10 elementos de tipo VARCHAR como máximo), agrega los controles de forma dinámica a la ventana y carga el valor de la base de datos para mostrarlo en la ventana (si ese campo ya tuviera un valor asignado previamente) y finalmente muestro la ventana:



Al clicar el botón de *Guardar*, se crea un diccionario con los datos del fichero (método **leerFichero()** visto con anterioridad) y con el método **AsignarColumnas(Dictionary <string, string> diccionario)** asigna cada dato del diccionario con su campo ADI correspondiente dependiendo del tipo del dato y llama al método **ActualizarCampo()** de Queries para guardar los valores en la base de datos y finalmente muestra un mensaje indicando que los campos se han guardado con éxito:



Al clicar el botón de *Cancelar*, simplemente se cierra el formulario:



Finalmente, al clicar el botón de Vaciar todo se limpian todos los campos de input llamando al método BorrarCamposDeInput (Control control):

