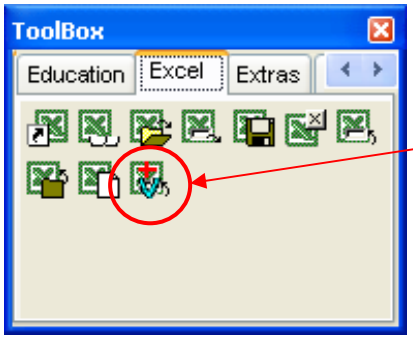
Make Excel Files
Visual IO

Ver 7.xx

How to create an excel file

Principle:

An excel file is structured in columns; each column is delimited by a separator character.



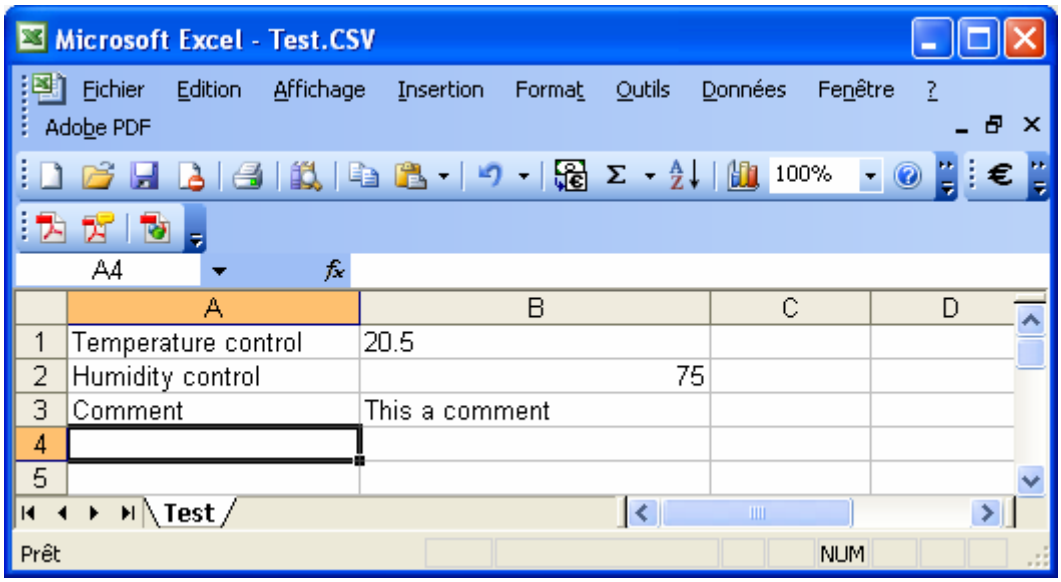
To create an excel file, you can use the **ColVarToXLS** Component for each column of your excel file. If you need 5 columns or 5 rows you need to use 5 components associated or one component in the case of the variables written are consecutives and of the same type.

Example:

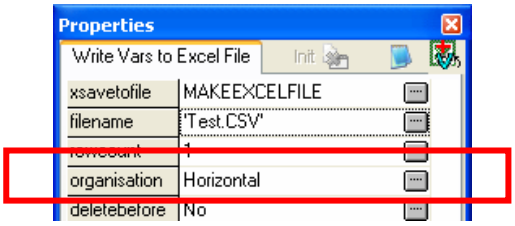
We want to store 3 different values in a excel file named TEST.CSV. We need to place 3 components on a form. In the global variable is declared

```
Temp          :  Real;
Humidity      :  Integer;
Comment       :  String;
MakeExcelFile :  Boolean; //** command to create the file
```

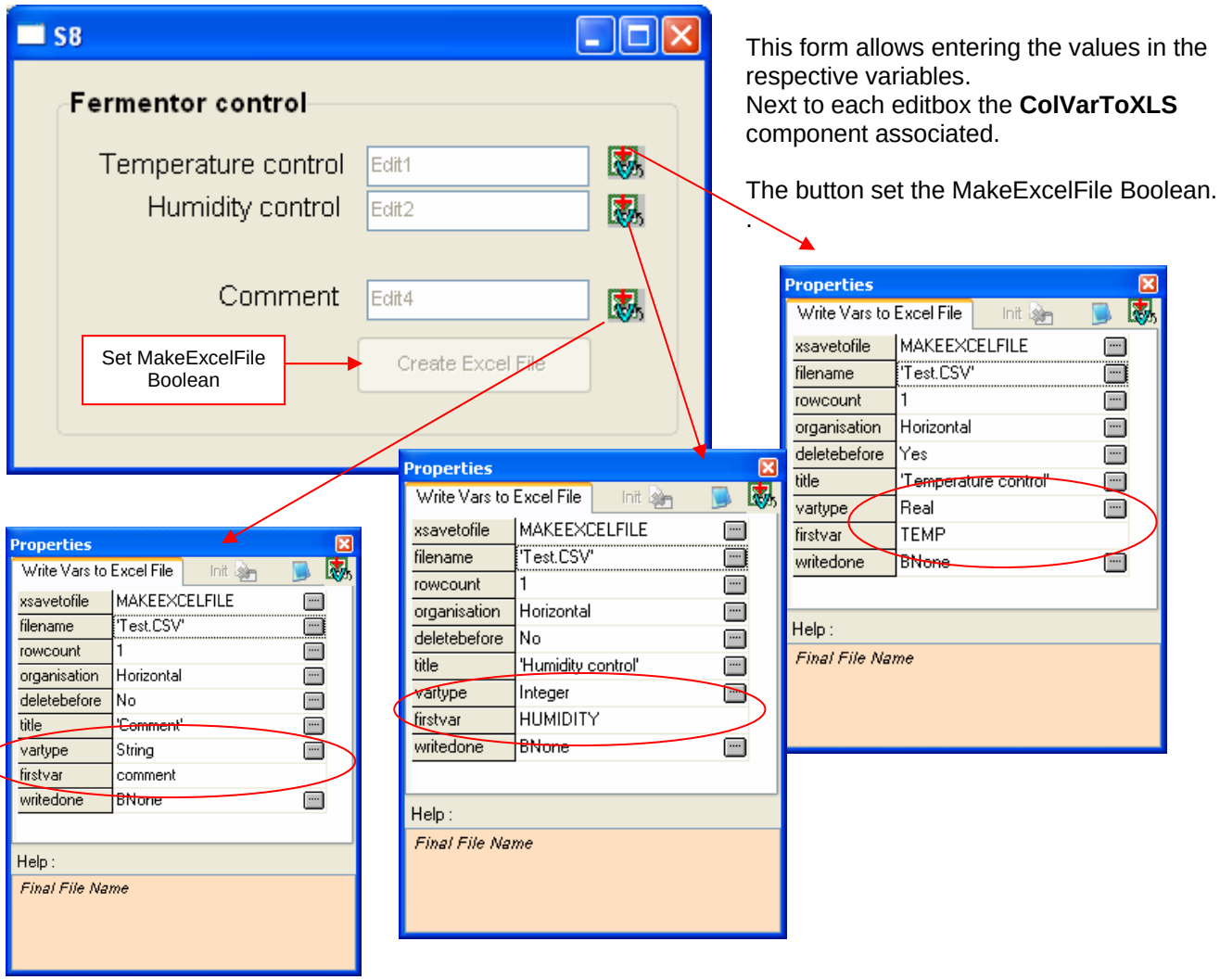
Final excel file generated



☞Note : in this example, the titles are in Column A. The representation is Horizontal so you must set the Organisation parameter to Horizontal. In the case of the titles are Row 0 the representation is Vertical (the values are in each column). so you must set the Organisation parameter to Vertical.



Form



This form allows entering the values in the respective variables.
Next to each editbox the **ColVarToXLS** component associated.

The button set the MakeExcelFile Boolean.
.

Component description

XSaveToFile : Command to save the file with the name specified in **FileName**.
Rowcount : Number of cells written with consecutives variables into vertical or horizontal direction.
Organisation : Writing in a column (Vertical) or in a row (Horizontal).
DeleteBefore : Yes allows to erase the final file before the treatment. Set Yes for the first column or first row.
Title : Column or row Title. If it is an empty string, no title is added to this one.
Vartype : Type of the series of variables to be displayed in the column or row.
FirstVar : Name of the first variable written in the column or the row. The other variables are those declared consecutively in the global variables.
WriteDone : Optional bit set when the writing is performed.

☞ Note : You can generate rows or columns with only one component but the variables must be declared consecutively in the Global Variables and must be of the same type.

If you need to write Single variables, it is necessary to transfer them in Real variables and work with real because Single and Double type are not treated by the component.

By program

You must use Tab character (Chr(9) or #9) as separators in XLS files.

Create an excel file by File functions

Global Variables are declared. The file generated is Test.xls.

```
  Temp          :  Real;
  Humidity      :  Integer;
  Agitator      :  Integer;
```

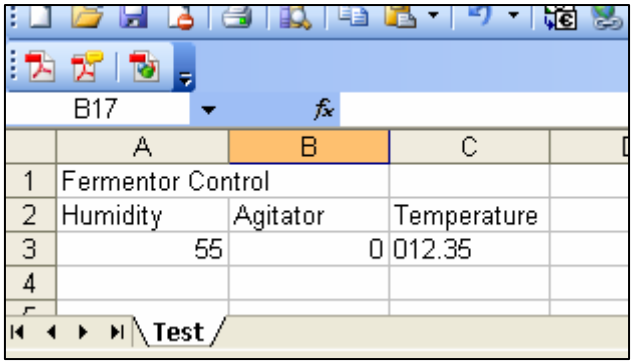
```
Uses GestFile;
{*** Button1 ************************************}
SObject Procedure Button1;
Var
  Hdle: Integer;
Begin
  If Button Then
  Begin
    FileDelete('Test.xls');  //*** Erase the old file before
    FileTextOpen(HDle,'Test.xls');

    //*** Single Text in the first Line
    Writeln(Hdle,'Fermentor Control');

    //** Write Each Column Title (3)
    Writeln(Hdle,'Humidity'+#9+'Agitator'+#9+'Temperature');

    //*** Write The associated values
    Writeln(Hdle,IntToStr(Humidity)+#9+
                 IntToStr(Agitator)+#9+
                 Formatfloat('00.00',Temp));

    //*** Close the File
    Filetextclose(Hdle);
   End;
End;
```



Create an excel file by the JCLEXcel library.

The JCLExcel library is similar to writing an excel file by file functions.
However the generated file is at XLS format.

```
Uses JCLEXCEL;
{*** Button1 ************************************}
SObject Procedure Button1;
Begin
  If Button Then
  Begin
    ExcelWriterCreate( 'Test.Xls');
    //*** Single Text in the first Line
    ExcelWriterWriteLABEL( 'Fermentor Control',0,0);
    //** Write Each Column Title (3)
    ExcelWriterWriteLABEL('Humidity',0,1);
    ExcelWriterWriteLABEL('Agitator',1,1);
    ExcelWriterWriteLABEL('Temperature',2,1);
    //*** Write The associated values
     ExcelWriterWriteLABEL(IntToStr(Humidity),0,2);
     ExcelWriterWriteLABEL(IntToStr(Agitator),1,2);
     ExcelWriterWriteLABEL(Formatfloat('00.00',Temp),2,2);
    //*** Close the File
    ExcelWriterClose;
   End;
End;
```

Create an excel file by OLE commands.

You need to have excel installed on your PC because your application will load excel and send command to write cell as you can do manually in the excel grid.

```
Uses EXCEL;

{*** Button1 ***************************************}
SObject Procedure Button1;
Begin
  If Button Then
  Begin
     StartExcel(False); //Excel hidden
     CreateFileExcel;   //** Create a new excel file
     //*** SelectSheet('Sheet1'); Can choose the sheet if necessary

     //*** Single Text in the first Line
     WriteCell(1,1, 'Fermentor Control');

     //** Write Each Column Title (3)
     WriteCell(1,2,'Humidity');
     WriteCell(2,2,'Agitator');
     WriteCell(3,2,'Temperature');

     //*** Write the associated values
     WriteCell(1,3,IntToStr(Humidity));
     WriteCell(2,3,IntToStr(Agitator));
     WriteCell(3,3,Formatfloat('00.00',Temp));

     //*** Save Close the File and release memory
     SaveAsExcel('Test.Xls');

     CloseFileExcel;
     QuitExcel;
  End;
End;
```

☞ Note : This program needs excel, memory and Windows mechanisms (OLE),
To write few cells may be a little complex.

You also need the ExcelIO.dll library in the C:\Windows directory.

All commands are available in individual components in the toolbox under excel tab.

Create an excel file by DDE command (valid but a little bit obsolete).

In addition to its DDE components, Visual I/O allows to control a DDE link by program.
These **commands** allow you to ship data to other programs and to retrieve data from them.
Attention only one channel DDE can be open in a program. And you need Excel installed in your PC.
Same mechanism than OLE command..

// Open a DDE channel
Procedure **OpenDDE** (Server : String) ;
// Close DDE Channel
Procedure **CloseDDE** ;
// Retrieve a data from the DDE server open with OpenDDE
Function **DDEGetData** (Topic, Item : String) : String ;
// Send a data into the DDE server open with OpenDDE
Function **DDESendData** (Topic, Item, Value : String) : String ;
// Send an Macro-command to the DDE server open with OpenDDE function
Function **DDESendMacro** (Macro : String) : String ;

☞ **Note** To load and save the excel file refer to macro command in excel documentation...

```
Var
  TS : String ;
  Data : Array [1..10] Of String ;

If Ok then
Begin
   {** Open the Excel server **}
   OpenDDE('C:\Program Files\Microsoft Office\OFFICE11\Excel.exe') ;
   For I :=1 to 10 Do              {** Retrieve the 10 cells of column 1 **}
   Begin
      TS := 'L'+IntToStr(I)+ 'C1' ;      {** compute the target by program **}
      Data[I] :=DDEGetData('Sheet1',TS) ; {** Get the cell **}
   End ;
  CloseDDE ;                 {** Close the DDE channel **}
End ;
```