XLSReadWrite II

Manual

Title page 1

Use this page to introduce the product

---

by <AUTHOR>

This is "Title Page 1" - you may use this page to introduce your product, show title, author, copyright, company logos, etc.

This page intentionally starts on an odd page, so that it is on the right half of an open book from the readers point of view. This is the reason why the previous page was blank (the previous page is the back side of the cover)

XLSREadWriteII

Printed: maj 2002 in (whereever you are located)

Publisher

...enter name...

Managing Editor

...enter name...

Technical Editors

...enter name...
...enter name...

Cover Designer

...enter name...

Team Coordinator

...enter name...

Production

...enter name...

Special thanks to:

All the people who contributed to this document, to mum and dad and grandpa, to my sisters and brothers and mothers in law, to our secretary Kathrin, to the graphic artist who created this great product logo on the cover page (sorry, don't remember your name at the moment but you did a great work), to the pizza service down the street (your daily Capricciosas saved our lives), to the copy shop where this document will be duplicated, and and and...

Last not least, we want to thank EC Software who wrote this great help tool called HELP & MANUAL which printed this document.

Table of Contents

Foreword

This is just another title page
placed between table of contents
and topics

Top Level Intro

This page is printed before a new top-level chapter starts

Part I

1 Introduction

Welcome to XLSReadWriteII, the most complete Delphi component for reading and writing Excel files.

XLSReadWriteII is the completely rewritten succeeder of the XLSReadWrite components.

With XLSReadWriteII can you not only access cell values, you can also:
- Edit existing files. Add or delete the cells you want.
- Format cells. Select the same formatting options as in Excel.
- Read and write formulas. You can also calculate the formula in a cell.
- Read & write merged cells.
- Read & write pictures.
- Read & write charts.
- Read & write cell notes.
- Read & write hyperlinks.
- Naitive Delphi components. No extra files or software has to be present.
- And many more things...

XLSReadWriteII works with Delphi 4, 5 and 6.

The XLSReadWriteII package also includes components for:
- Import of any database table.
- Import of HTML tables. It extracts the TABLE tags from the HTML file. Very useful for reading stock quotes or similar data.
- Export to HTML.

XLSReadWriteII is easy to use. Cells are accessed in the same way as a grid. Just use any of the AsXXXCell[Col,Row] functions in order to read or write a cell.
Here is a small example on how to write to some cells. Assuming you have a TXLSReadWriteII component on the form named XLS.

```
procedure Form1.WriteCells;
begin
  // Set the filename.
  XLS.Filename := 'Test.xls';
  // Insert a string.
  XLS.Sheets[0].AsString[0,0] := 'Hello';
  // Insert some numbers.
  XLS.Sheets[0].AsFloat[2,0] := 10;
  XLS.Sheets[0].AsFloat[2,1] := 20;
  XLS.Sheets[0].AsFloat[2,2] := 30;
  XLS.Sheets[0].AsFloat[2,3] := 40;
  // Insert a formula which sums the above values.
  XLS.Sheets[0].AsFormula[2,4] := 'SUM(C1:C4)';
  // Write the file.
  XLS.Write;
end;
```

Top Level Intro

This page is printed before a new
top-level chapter starts

Part II

2 Contact

Address
Axolot Data
Dr. Forselius backe 38
SE-413 26 Goteborg
SWEDEN

Phone
+46 (0)31 708 54 47
Office time: 13:00 to 17:00
GMT +1 Please respect time zones!

Fax
+46 (0)31 18 77 93

E-mail
components@axolot.com

Web site
www.axolot.com

Top Level Intro

This page is printed before a new
top-level chapter starts

Part III

3 XLSReadWrite

3.1 TXLSReadWriteII

3.1.1 TXLSReadWriteII

3.1.2 TXLSReadWriteII.AreaNames

Provides access to named areas in the workbook.

```
property AreaNames: TAreaNames;
```

Description
Use AreaNames to access the named areas in the workbook.

3.1.3 TXLSReadWriteII.Backup

If backups shall be written.

```
property Backup: boolean;
```

Description
The Backup property specifies whether Microsoft Excel should save backup versions of a file.

3.1.4 TXLSReadWriteII.BookProtected

If the workbook is protected

```
property BookProtect: boolean;
```

Description
The BookProtect property stores the protection state for a sheet or workbook.

3.1.5 TXLSReadWriteII.Codepage

The code page for the file.

```
property Codepage: word;
```

Description
Codepage is used for localization.
The codepage property of an Excel file is not the same as the codepage used by Windows. To obtaine the correct value for the codepage, open an Excel file with XLSReadWriteII, check the value of the Codepage property, and then use it when you are witing Excel files.

3.1.6 TXLSReadWriteII.Filename

The name of the Excel read or written.

```
property Filename: string;
```

Description
Set the Filename property to the name of the file you want to read or write.

3.1.7 TXLSReadWriteII.Font

The standard font used by the workbook.

`property Font: TXFont;`

Description
Change this property if you want to change the standard font used by the workbook.

3.1.8 TXLSReadWriteII.Fonts

The Fonts property is a list of all fonts used by the formats of the file.

`property Fonts: TXFonts;`

Description
Add new fonts here for use in the cell formats.

Note
Never delete any fonts as this will change the index order used by the cell formats.

3.1.9 TXLSReadWriteII.Formats

The Formats property is a list of all cell formats.

`property Formats: TCellFormats;`

Description
Add new cell formats here for use when formatting the cells.

3.1.10 TXLSReadWriteII.FuncArgSeparator

Character used to separate function arguments in strings.

`property FuncArgSeparator: char;`

Description
Change this property if you want to change the character that is used to separate function arguments. It is by default set to the global variable ListSeparator.

3.1.11 TXLSReadWriteII.GetDEVMODE

Returns a DEVMODE structure, which can be used to alter various printer settings.

`function GetDEVMODE: PDeviceModeW;`

Description
A call to the GetDEVMODE method will create a DEVMODE structure and return it. if there allready is a previous created structure, it will be returned. The structure will also be written to the file. If the GetDEVMODE method not is called, no structure is written. The DEVMODE structure containes device specific information for the selected printer. This settings can after the structure is obtained be changed. Example of settings which can be set are: paper size, printing scale, number fo copies etc. For a full description, see the Win32.hlp file.

The printer settings which are used as a source for GetDEVMODE is the one selected by the Printer variable.

Note
The structure returned by GetDEVMODE is not compatible with the one used by Delphi's TPrinter object. Excel always uses wide strings, while Delphi don't do so.

See also
HasDEVMODE

3.1.12 TXLSReadWriteII.HasDEVMODE

Checks wether XLSReadWriteII has a DEVMODE structure assigned to it.

```
function HasDEVMODE: boolean;
```

Description
Use HasDEVMODE to check if the XLSReadWriteII object has a DEVMODE structure assigned to it. The result is True if this is the case.

See also
GetDEVMODE

3.1.13 TXLSReadWriteII.IsMac

True if the file is created by Excel for Machintosh.

```
property IsMac: boolean;
```

Description
Read this property to check if a file opened is created by Excel for Machintosh.

3.1.14 TXLSReadWriteII.Names

Enter topic text here.

3.1.15 TXLSReadWriteII.OptionsDialog

Parameters from the Options dialog in Excel.

```
property OptionsDialog: TOptionsDialog;
```

Description
Here you will find parameters from the Tools->Options dialog in Excel.

3.1.16 TXLSReadWriteII.Pictures

Provides access to the pictures used in the workbook.

```
property Pictures: TXLSPictures;
```

Description
Use Pictures to access the pictures in the workbook.

3.1.17 TXLSReadWriteII.PictureOptions

Options for pictures.

```
type TPictureOption = (poInMemory,poDeleteTempFiles);
     TPictureOptions = set of TPictureOption;
```

```
property PictureOptions: TPictureOptions;
```

Description
Change this property in order to controll how pictures are stored before/after pictures in Excel files are read/written.

Value	Meaning
poInMemory	Pictures are stored in memory. **This is not implemented yet**
poDeleteTempFiles	If pictures not are stored in memory, they are written as temporary files in the SYSTEMP directory. If this value is set, these temporary files are deleted when the component is destroyed or cleard.

3.1.18 TXLSReadWriteII.RefreshAll

If external data should be refreshed when the workbook is loaded.

```
property RefreshAll: boolean;
```

Description
Set the RefreshAll property to True if all external data should be refreshed when the workbook is loaded.

3.1.19 TXLSReadWriteII.Sheets

Provides access to the sheets in the workbook.

```
property Sheets: TSheets;
```

Description
Use Sheets to access the worksheets in the workbook

3.1.20 TXLSReadWriteII.ShowFormulas

If formulas, when shown as strings, shall show the formula or the result (value) of the formula.

```
property ShowFormulas: boolean;
```

Description
Set the ShowFormulas property to True if functions which reads cells and return string values shall return the formula itself or de result (value) of the formula.

See also
AsFmtString, AsString

3.1.21 TXLSReadWriteII.StrFALSE

String used for the boolean value False when represented in strings.

```
property StrFALSE: string;
```

Description
Change StrFALSE property to change the string representation of the boolean value False. The default is 'False'.

See also
StrTRUE

3.1.22 TXLSReadWriteII.StrTRUE

String used for the boolean value True when represented in strings.

```
property StrTRUE: string;
```

Description
Change StrTRUE property to change the string representation of the boolean value False. The default is 'True'.

See also
StrFALSE

3.1.23 TXLSReadWriteII.UserName

The user (creator) of the workbook.

```
property UserName: string;
```

Description
Change the UserName property to set the name of the creator of the workbook.

3.1.24 TXLSReadWriteII.Version

The Excel version of the file read or written.

```
type TExcelVersion =
(xvNone,xvExcelUnknown,xvExcel21,xvExcel30,xvExcel40,xvExcel50,xvExcel97);
```

```
property Version: TExcelVersion;
```

Description
Read the Version property to check the version of the Excel file read. Set the Version property to the version you want to write.

Note: It is probably not possible to change the version of a file. For files which not containes any formulas, it may work. But as soon as the file containes formulas, Excel will crash if it opens a file saved in another version than it was created in.

The Excel file format has not changed since Excel 97, and will probably not do so in the future, as it is possible to extend the file format with new features without loosing compability with previous versions.

Value	Meaning
xvNone	No version is defined.
xvExcelUnknown	The version can't be determinated.
xvExcel21	Excel 2.x
xvExcel30	Excel 3.x
xvExcel40	Excel 4.x
xvExcel50	Excel 5.x/Excel 95
xvExcel97	Excel 97/Excel 2000/Excel XP

3.1.25 TXLSReadWriteII.Workbook

Workbook options.

```
property Workbook: TWorkbookData;
```

Description
Options for the worksheets in the workbook.

3.1.26 TXLSReadWriteII.WriteUnicodeStrings

If unicode strings shall be written.

```
property WriteUnicodeStrings: boolean;
```

Description
Set WriteUnicodeStrings to true if unicode (multibyte) strings shall be written.

3.1.27 TXLSReadWriteII.Clear

Clears all data in the workbook.

```
procedure Clear;
```

Description
Use Clear to empty the workbook of all data.

See also
ClearCells

3.1.28 TXLSReadWriteII.ClearCells

Clears all cells in the workbook.

```
procedure ClearCells;
```

Description
Use Clear to delete all cells in the workbook. All other data are still kept.

See also
Clear

3.1.29 TXLSReadWriteII.CopyCells

Copies a range of cells.

```
procedure
CopyCells(SrcSheet,Col1,Row1,Col2,Row2,DestSheet,DestCol,DestRow);
```

Description
CopyCells copies the cells in area Col1,Row1 - Col2,Row2 on sheet SrcSheet to DestCol and DestRow on sheet DestSheet. The copying is carried out correctly even if the areas overlap. Associated notes in the area are copied as well. Merged cells which falls in the area are not copied.

See also
MoveCells, DeleteCells

3.1.30 TXLSReadWriteII.MoveCells

Moves a range of cells.

```
procedure
MoveCells(SrcSheet,Col1,Row1,Col2,Row2,DestSheet,DestCol,DestRow);
```

Description
MoveCells moves the cells in area Col1,Row1 - Col2,Row2 on sheet SrcSheet to DestCol and DestRow on sheet DestSheet. The moving is carried out correctly even if the areas overlap. Associated notes in the area are moved as well. Merged cells which falls in the area are not moved.

See also
CopyCells, DeleteCells

3.1.31 TXLSReadWriteII.Read

Reads an workbook.

```
procedure Read;
```

Description
Call Read to read the workbook indicated by Filename. Before the workbook is loaded, all data is cleard. The cells in the workbook can then be accessed trough the Sheets property.

See also
Filename, TSheets

3.1.32 TXLSReadWriteII.Write

Writes a workbook.

```
procedure Write;
```

Description
Call Write to write the workbook to the file indicated by the Filename property.

See also
Filename, WriteToStream

3.1.33 TXLSReadWriteII.WriteToStream

Writes a workbook to a stream.

```
procedure WriteToStream(Stream: TStream);
```

Description
Call Write to write the workbook to the stream indicated by the Stream parameter.

See also
Write

3.1.34 TXLSReadWriteII.OnProgress

This event is fired when a file is read.

```
type TIntegerEvent = procedure (Sender: TObject; Value: integer) of
object;

property OnProgress: TIntegerEvent;
```

Description
The OnProgress is fired while a file is read. The first time Value is zero, and the last time Value is 100.

3.1.35 TXLSReadWriteII.OnFunction

This event is fired when an unknown function is encounterd while calculating a formula.

```
type TFunctionEvent = procedure(Sender: TObject; FuncName: string; Args:
Variant; var Result: Variant) of object;

property OnFunction: TFunctionEvent;
```

Description
Use the OnFunction event to do the calculation of formulas which not are calculated by TXLSReadWrite.
FuncName is the name of the function. Args are the functions arguments, from left to right. Result is the result of the calculation. If the function not can be calculated, set Result to NULL

See also
Calculate

3.2 TAreaName

3.2.1 TAreaName

TAreaName is a named area, which can be used in formulas when referrering to an area on the worksheet.

Unit
Names

3.2.2 TAreaName.AreaName

The name of the named area.

```
property Name: string;
```

Description
Use Name to set the name of the named area. Names are case-insencitive. If the name allready is defined, an exeption is raised.

See also
NameDef

3.2.3 TAreaName.NameDef

The definition of a named area.

```
property NameDef: string;
```

Description
Use NameDef to define the named area. NameDef can be any valid excel formula.

Examples

Value	Meaning
B4	The value of cell B4.
Sheet3!B4	The value of cell B4 on sheet Sheet3.
B4:D7	The area with upper left corner at cell B4 and lower right corner at cell D7.
B4*100	The value of cell B4 multiplied by 100.
SIN(1.5)	Sinus of 1.5

See also
AreaName

3.3 TAreaNames

3.3.1 TAreaNames

unit
Names

TAreaNames is an ancestor of TCollection.
TAreaNames is a container for the TAreaName objects.

3.3.2 TAreaNames.Add

Add's a new named area.

```
function Add: TAreaName;
```

Description
Use Add to add a new named area.

3.3.3 TAreaNames.FindName

Search the named areas for a specific name.

```
function FindName(AName: string): TAreaName;
```

Description
Use FindName to search the named areas for AName name. The search is case-insensitive. If the name is found, the named area is returned. If it not is found, Nil is returned.

3.3.4 TAreaNames.Items

Provides access to defined named areas.

```
property Items[Index: integer]: TAreaName;
```

Description
Read Items to access the list of named areas.

3.4 TCellFormat

3.4.1 TCellFormat

TCellFormat is an individual format, which can be used to format cells by assigning the cell's FormatIndex property to the index of the TCellFormat created.

Unit
CellFormats

3.4.2 TCellFormat.BorderTopColor

The color of the top cell border.

```
property BorderTopColor: TExcelColor;
```

See also
TCellBorderStyle.

3.4.3 TCellFormat.BorderTopStyle

The style of the top cell border.

```
property BorderTopStyle: TCellBorderStyle;
```

See also
TExcelColor.

3.4.4 TCellFormat.BorderLeftColor

The color of the left cell border.

```
property BorderLeftColor: TExcelColor;
```

See also
TCellBorderStyle.

3.4.5 TCellFormat.BorderLeftStyle

The style of the left cell border.

```
property BorderLeftStyle: TCellBorderStyle;
```

See also
TExcelColor.

3.4.6 TCellFormat.BorderRightColor

The color of the right cell border.

```
property BorderRightColor: TExcelColor;
```

See also
TCellBorderStyle.

3.4.7 TCellFormat.BorderRightStyle

The style of the right cell border.

```
property BorderRightStyle: TCellBorderStyle;
```

See also
TExcelColor.

3.4.8 TCellFormat.BorderBottomColor

The color of the bottom cell border.

```
property BorderBottomColor: TExcelColor;
```

See also
TCellBorderStyle.

3.4.9 TCellFormat.BorderBottomStyle

The style of the bottom cell border.

```
property BorderbOTTOMStyle: TCellBorderStyle;
```

See also
TExcelColor.

3.4.10 TCellFormat.BorderDiagColor

The color of the diagonal cell border.

```
property BorderDiagColor: TExcelColor;
```

See also
TCellBorderStyle.

3.4.11 TCellFormat.BorderDiagStyle

The style of the diagonal cell border.

```
property BorderDiagStyle: TCellBorderStyle;
```

See also
TExcelColor.

3.4.12 TCellFormat.BorderDiagLines

How diagonal lines are drawn.

```
type TDiagLines = (dlNone,dlDown,dlUp,dlBoth);
```

```
property BorderDiagLines: TDiagLines;
```

Value	Meaning
dlNone	No lines
dlDown	From upper left corner to lower right.
dlUp	From lower left to upper right.
dlBoth	Bot lines are drawn.

See also
TExcelColor.

3.4.13 TCellFormat.FillPatternBackColor

The background fill color.

```
property FillPatternBackColor: TExcelColor;
```

Description
Use this property to set the color of the background fill pattern.
Note: If no fill pattern is assigned to the format, this property has no effect.
If you just want to set the cell color, use FillPatternForeColor.

See also
FillPatternForeColor, FillPattern.

3.4.14 TCellFormat.FillPatternForeColor

The foreground fill color.

```
property FillPatternForeColor: TExcelColor;
```

Description
Use this property to set the color of the foreground fill pattern.
Note: **This is the cell color**. If you want to have just one color for the cell, use this property.

See also
FillPatternBackColor, FillPattern.

3.4.15 TCellFormat.FillPatternPattern

The fill pattern for the cell.

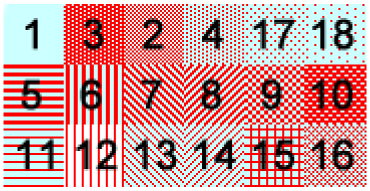
```
property FillPatternPattern: byte;
```

Description

Use this property to set the fill pattern for the cell. Values can range between 0-127. 0 (zero) is no fill pattern.

See also
FillPatternForeColor, FillPatternBackColor.

Fill patterns and there corresponding numbers, which are used in Excel:



3.4.16 TCellFormat.FontIndex

The index into the Fonts property for the font used by the format.

```
property FontIndex: integer;
```

Description
Set the FontIndex property to a font in the Fonts property in order to assign a new font for the format. A value of zero means that the default font is used. The default font is the font for the worksheet.

See also
Fonts

3.4.17 TCellFormat.FormatOptions

Cell format options.

```
type TFormatOption = (foWrapText,foShrinkToFit);
     TFormatOptions = set of TFormatOption;

property FormatOptions: TFormatOptions;
```

Value	Meaning
foWrapText	Wrap text in cells.
foShrinkToFit	Shrink text to fit horizontal cell space. The result is that the font size is changed so the text fit's the cell's horizontal size.

3.4.18 TCellFormat.HorizAlignment

Horizontal alignment of text in cells.

```
type TCellHorizAlignment =
(chaGeneral,chaLeft,chaCenter,chaRight,chaFill,chaJustify,chaCenterAcross)
;

property HorizAlignment: TCellHorizAlignment;
```

Value	Meaning
chaGeneral	No alignment.
chaLeft	Left alignment
chaCenter	Center alignment.
chaRight	Right alignment
chaFill	Fill's the entire cell with the text or character. Like: 'XXXXXXXXXX'.
chaJustify	Justify's the word space to fit the text in the cell.
chaCenterAcross	Don't know what this is.

3.4.19 TCellFormat.Indent

Text indent.

```
property Indent: byte;
```

Description
The indent values can range from 0-15. A value of 0 (zero) is no indent. An increase with one increases the indent with about one character.

3.4.20 TCellFormat.Name

The name of the format.

```
property Name: string;
```

Description
Set the name of the format if you want to retrieve the format's index by it's name.

See also
IndexByName

3.4.21 TCellFormat.NumberFormat

The number format mask of the format.

```
property NumberFormat: string;
```

Description
Change the NumberFormat property in order to change the way numbers are displayed.
Number formats controls how many decimals are displayed, if there shall be thousands separator, etc.
Note: Date and time are stored in Excel in the same way as in Delphi, ie. as floating point numbers. To write an date or time value to an cell, use any standard Delphi date and time functions to create the value. Then write it as an number and format it as an date or time.

See also
Excel Number Format Strings

3.4.22 TCellFormat.Protection

Protection of cells.

```
type TCellProtection = (cpLocked,cpHidden);
     TCellProtections = set of TCellProtection;

property Protection: TCellProtections;
```

Value	Meaning
cpLocked	Cell is locked. This does not mean that the cell value not can be change. To prevent the cell from being changed, the worksheet has to be locked.
cpHidden	Cell value is hidden.

3.4.23 TCellFormat.Rotation

Rotation of cell text.

```
property Rotation: smallint;
```

Description
Rotation, in degrees.
The value 0 90 is rotation up 0 90 deg. The value 91 180 is rotation down 1 90 deg. The value 255 is vertical text.

3.4.24 TCellFormat.VertAlignment

Horizontal alignment of text in cells.

```
type TCellVertAlignment = (cvaTop,cvaCenter,cvaBottom,cvaJustify);

property VertAlignment: TCellVertAlignment;
```

Value	Meaning
cvaTop	Top alignment.
cvaCenter	Center alignment.
cvaBottom	Bottom alignment
cvaJustify	Justify's the line space to fit the text in the cell.

3.4.25 TCellFormat.FormatIsDateTime

True if the formats NumberFormat property is a date or time format.

```
function FormatIsDateTime: boolean;
```

Description
The method returns true if the NumberFormat property of the format is a date or time format.

See also
Excel Number Format Strings, NumberFormat

3.4.26 TCellFormat.Equal

Compares another format with itself. Returns true if they are equal.

```
function Equal(F: TCellFormat): boolean;
```

3.5 TCellFormats

3.5.1 TCellFormats

TCellFormats provides access to the formats used to format cells.

Unit
CellFormats

In order to assign a format to a cell, you first need to create a format and then use the format's index for the FormatIndex property of the cell. If you read an Excel file, don not delete any formats, as this will most likely end with an corrupt file if you write it again.

3.5.2 TCellFormats.Items

Provides access to the formats used by TXLSReadWrite.

```
property Items[Index: integer]: TCellFormat;
```

Description
Read Items to access the list of cell formats used to format cells.

3.5.3 TCellFormats.NumberFormats

Provides access to the number formatting masks.

```
property NumberFormats: TStringList;
```

Description
Number formats are used to format numbers, like decimals, percent etc. Normally, there is no need to directly access the NumberFormats property.

3.5.4 TCellFormats.Add

Add's a new format.

```
procedure Add;
```

Description
Use Add to add new formats.

3.5.5 TCellFormats.IndexByName

Returns a formats index by the format's name.

```
function IndexByName: integer;
```

Description
Use IndexByName to get a format's index by its name. If the name not is found, -1 is returned.

3.6 TXFont

3.6.1 TXFont

unit
XLSFonts

3.6.2 TXFont.Charset

The character set of the font.

```
property Charset: TFontCharset;
```

3.6.3 TXFont.Color

The color of the font.

```
property Color: TExcelColor;
```

3.6.4 TXFont.Name

The name of the font.

```
property Name: string;
```

3.6.5 TXFont.Size

The size of the font.

```
property Size: integer;
```

Description
Change the Size property to set the font's size. The Size property is in points.

3.6.6 TXFont.Style

Font style.

```
type TXFontStyle = (xfsBold,xfsItalic,xfsStrikeOut);
     TXFontStyles = set of TXFontStyle;

property Style: TXFontStyle;
```

Value	Meaning
xfsBold	Set the font style to boldface.
xfsItalic	Set the font style to italic.
fxsStrikeOut	The font is displayed with a horizontal line through it.

3.6.7 TXFont.SubSuperScript

Sets sub or superscript of the font.

```
type TXSubSuperscript = (xssNone,xssSuperscript,xssSubscript);

property SubSuperscript: TXSubSuperscript;
```

Value	Meaning
xssNone	No sub or superscript.
xssSuperscript	The font is superscript.
xssSubscript	The font is subscript.

3.6.8 TXFont.Underline

Underline style of the font.

```
type TXUnderline =
(xulNone,xulSingle,xulDouble,xulSingleAccount,xulDoubleAccount);

property Underline: TXUnderline;
```

Value	Meaning
xulNone	No underline.
xulSingle	Single line underline.
xulDouble	Double line underline.
xulSingleAccount	Single accounting line underline.

xulDoubleAccount	Double accounting line underline.

3.6.9 TXFont.Assign

Assignes another TXFont to the font.

procedure Assign(Source: TPersistent); **override;**

Description
Use Assign to assign Source to the font.

3.6.10 TXFont.AssignTFont

Assigns a TFont to the font.

procedure AssignTFont(Source: TFont);

Description
Use AssignTFont to assign the properties of a TFont to the font.

3.6.11 TXFont.CopyToTFont

Copies the properties of the font to an TFont..

procedure CopyToTFont(Dest: TFont);

Description
Use CopyToTFont to copy the properties of the font to an TFont object.

3.6.12 TXFont.Equal

Returns true if Value TXFont is equal to the font.

function Equal(Value: TXFont): boolean;

3.7 TXFonts

3.7.1 TXFonts

TXFonts is a container for the fonts used in the file.

unit
XLSFonts

Description
Change the Size property to set the font's size. The Size property is in points.

3.7.2 TXFonts.Add

Add a new font to the container.

```
function Add: TXFont;
```

Description
Use the Add method to add a new font to the fonts container. The newly added font is returned.

See also
AddIndex

3.7.3 TXFonts.AddIndex

Add a new font to the container.

```
function AddIndex: integer;
```

Description
Use the AddIndex method to add a new font to the fonts container. The index of the font added is return. This index can be used to set the FontIndex property for the cell formats.

See also
Add

3.7.4 TXFonts.GetIndex

Returns the index of a font.

```
function GetIndex(Font: TXFont): integer;
```

Description
Use the GetIndex method to retrieve the index of a font. If there is a font in the container which matches the properties of the Font parameter, the index of that font is returned. If not match is found, a new font is added (and the properties of the Font parameter is assigned to the new font) and the index of the new font is returned.

See also
Add, AddIndex

3.7.5 TXFonts.Items

Returns a font from the container.

```
function Items[Index: integer]: TXFont; default;
```

Description
Use the Items method in order to retrieve a font from the container.

3.8 TOptionsDialog

3.8.1 TOptionsDialog

unit
XLSUtils

Description
The TOptionsDialog contains options for the workbook. Please referrer to the Excel documentation for the meaning of the options.

```
property SaveExtLinkVal: boolean;
property CalcCount: word;

type TCalcMode = (cmManual,cmAutomatic,cmAutoExTables);
property CalcMode: TCalcMode;

property Delta: double;
property UserName: string;

type TShowObjects = (soShowAll,soPlaceholders,soHideAll);
property ShowObjects: TShowObjects;

property Iteration: boolean;
property PrecisionAsDisplayed: boolean;
property R1C1Mode: boolean;
property RecalcBeforeSave: boolean;
property Uncalced: boolean;
```

3.9 TXLSPicture

3.9.1 TXLSPicture

unit
Picture

Description
Pictures used in workbooks are stored in the Pictures property of the TXLSReadWriteII object. Pictures are inserted into the worksheets trough the SheetPictures property of the TSheet objects. This is to avoid storing the same picture in several's worksheets.
Note: Excel uses three formats for pictures, BMP, JPEG and PNG. BMP is only used for very small pictures. Even if you insert a BMP picture into Excel it will probably be converted to a PNG picture. As there is no native support for PNG pictures in Delphi, you need an external library to view these pictures. You don't need this library if you just want to read or write a picture to the XLS file.

See also
Pictures, SheetPictures

3.9.2 TXLSPicture.Filename

The name of the file with the picture.

```
property Filename: string;
```

If you write a file, this shall point to a valid picture file. Please note that the file must persist until the file is written.
When you read a file, Filename points to a temporary file. You can copy this temporary file to a permanent location if you want to keep it. The temporary file is deleted when the TXLSReadWrite object is destroyed, or cleared.

See also
PictureOptions

3.9.3 TXLSPicture.Height

The height of the picture in pixels.

```
property Height: integer;
```

Description
Set the Height property to the height of the picture in pixels. XLSReadWrite can calculate the height if you set Height to zero, but in order to do so a library which can handle the picture format has to be present. For BMP pictures nothing extra is needed. For JPEG pictures, the jpeg unit has to be added to the uses clause. For PNG pictures, an external library has to be used.

See also
TXLSPictures Width

3.9.4 TXLSPicture.Name

The name of the TXLSPicture object.

3.9.5 TXLSPicture.PictureType

Read only.
The picture type, as detected by TXLSReadWrite.

3.9.6 TXLSPicture.Size

Read only.
The sixe of the picture file in bytes.

3.9.7 TXLSPicture.Width

The width of the picture in pixels.

```
property Width: integer;
```

Description
Set the Width property to the width of the picture in pixels. XLSReadWrite can calculate the width if you set Width to zero, but in order to do so a library which can handle the picture format has to be present. For BMP pictures nothing extra is needed. For JPEG pictures, the jpeg unit has to be added to the uses clause. For PNG pictures, an external library has to be used.

See also
TXLSPictures Height

3.10 TXLSPictures

3.10.1 TXLSPictures

unit
Pictures

TXLSPictures is an ancestor of TCollection.
TXLSPictures is a container for the TXLSPicture objects.

3.10.2 TXLSPictures.Add

Add a new TXLSPicture to the container.

```
function Add: TXLSPicture;
```

Description
Use the Add method to add a new picture to the container.

3.10.3 TXLSPictures.Items

Provides access to the TXLSPicturel objects.

```
property Items[Index: integer]: TXLSPicture; default;
```

Description
Read Items to access a TXLSPicture object.

See also
TXLSPicture

3.11 TSheet

3.11.1 TSheet

TSheet holds the cells and data for an worksheet.

unit
SheetData

description
TSheet is an ancestor of TCollectionItem. TSheet is used to read and write cell values and to set properties specific for the worksheet. Pictures and Notes are also accessed trough TSheet.

3.11.2 TSheet.AsBlank

Access a cell as an blank cell value.

```
property AsBlank[Col,Row: integer]: boolean;
```

Description
Use AsBlank to read or write the cell at Col and Row as an blank cell value. If there is no cell at the position, a blank cell will be inserted, when the property is write-accessed. If there already is a cell there, it will be converted to a blank cell. If the previous cell was formatted, the formatting will be

copied to the inserted cell.

See also
WriteBlank

3.11.3 TSheet.AsBoolean

Access a cell as an boolean cell value.

```
property AsBoolean[Col,Row: integer]: boolean;
```

Description
Use AsBoolean to read or write the cell at Col and Row as an boolean cell value. If there is no cell at the position, an boolean cell will be inserted, when the property is written. If there already is a cell there, it will be deleted, and an boolean cell will be inserted. If the previous cell was formatted, the formatting will be copied to the inserted cell. When the property is read, the value of the cell will be returned. If there is no cell at the position, False will be returned.

See also
WriteBoolean

3.11.4 TSheet.AsFloat

Access a cell as an float cell value.

```
property AsFloat[Col,Row: integer]: double;
```

Description
Use AsFloat to read or write the cell at Col and Row as an float cell value. If there is no cell at the position, an float cell will be inserted, when the property is written. If there already is a cell there, it will be deleted, and an float cell will be inserted. If the previous cell was formatted, the formatting will be copied to the inserted cell. When the property is read, the value of the cell will be returned. If there is no cell at the position, zero will be returned.

See also
WriteNumber

3.11.5 TSheet.AsFmtString

Returns the cell value as an string, if formatted the string will be formatted according to that.

```
property AsFmtString[Col,Row: integer]: string;
```

Description
Use AsFmtString to read or write the cell at Col and Row as an string value. If there is no cell at the position, an string cell will be inserted, when the property is written. If there already is a cell there, it will be deleted, and an string cell will be inserted. If the previous cell was formatted, the formatting will be copied to the inserted cell. When the property is read, the value of the cell will be returned. If the cell is of another type than string, a string value of the cell will be returned. When the cell is formatted, formatting rules will be used. If there is no cell at the position, an empty string will be returned.

See also
AsString, WriteString

3.11.6 TSheet.AsString

Access a cell as an string cell value.

```
property AsString[Col,Row: integer]: string;
```

Description
Use AsString to read or write the cell at Col and Row as an string cell value. If there is no cell at the position, an string cell will be inserted, when the property is written. If there already is a cell there, it will be deleted, and an string cell will be inserted. If the previous cell was formatted, the formatting will be copied to the inserted cell. When the property is read, the value of the cell will be returned. If there is no cell at the position, an empty string will be returned.

See also
WriteString

3.11.7 TSheet.AsInteger

Access a cell as an integer cell value.

```
property AsInteger[Col,Row: integer]: integer;
```

Description
Use AsInteger to read or write the cell at Col and Row as an integer cell value. If there is no cell at the position, an integer cell will be inserted, when the property is write-accessed. If there already is a cell there, it will be deleted, and an integer cell will be inserted. If the previous cell was formatted, the formatting will be copied to the inserted cell. When the property is read, the value of the cell will be returned. If there is no cell at the position, zero will be returned.

See also
WriteNumber

3.11.8 TSheet.AsError

Access a cell as an error cell value.

```
property AsError[Col,Row: integer]: TCellError;
```

Description
Use AsError to read or write the cell at Col and Row as an error cell value. If there is no cell at the position, an error cell will be inserted, when the property is write-accessed. If there already is a cell there, it will be deleted, and an error cell will be inserted. If the previous cell was formatted, the formatting will be copied to the inserted cell. When the property is read, the value of the cell will be returned. If there is no cell at the position, zero will be returned.

See also
WriteError

3.11.9 TSheet.AsFormula

Access a cell as a formula cell value.

```
property AsFormula[Col,Row: integer]: string;
```

Description
Use AsFormula to read or write the cell at Col and Row as an formula cell value. If there is no cell at the position, a formula cell will be inserted, when the property is written. If there already is a cell there,

it will be deleted, and a formula cell will be inserted. If the previous cell was formatted, the formatting will be copied to the inserted cell. When the property is read, the value of the cell will be returned. If there is no cell at the position, an empty string will be returned.
The value inserted into the cell must be a valid Excel formula.
If you want to read or write the value of the formula, use AsNumFormulaValue, AsStrFormulaValue or AsBoolFormulaValue.

See also
WriteNumFormula, WriteStrFormula, WriteBoolFormula

3.11.10 TSheet.AsNumFormulaValue

Access the value of a number formula cell.

```
property AsNumFormulaValue[Col,Row: integer]: double;
```

Description
Use AsNumFormulaValue to read or write the value of a numeric formula cell at Col and Row. If the cell not is a number formula cell, or there is no cell at the position, an exception will be raised. To read or write the formula, use AsFormula.

See also
AsStrFormulaValue, AsBoolFormulaValue

3.11.11 TSheet.AsStrFormulaValue

Access the value of a string formula cell.

```
property AsStrFormulaValue[Col,Row: integer]: double;
```

Description
Use AsStrFormulaValue to read or write the value of a string formula cell at Col and Row. If the cell not is a string formula cell, or there is no cell at the position, an exception will be raised. To read or write the formula, use AsFormula.

See also
AsNumFormulaValue, AsBoolFormulaValue

3.11.12 TSheet.AsBoolFormulaValue

Access the value of a boolean formula cell.

```
property AsBoolFormulaValue[Col,Row: integer]: double;
```

Description
Use AsBoolFormulaValue to read or write the value of a boolean formula cell at Col and Row. If the cell not is a boolean formula cell, or there is no cell at the position, an exception will be raised. To read or write the formula, use AsFormula.

See also
AsNumFormulaValue, AsStrFormulaValue

3.11.13 TSheet.AutoColWidth

Sets column width automatically.

```
function AutoWidthCol(Col: integer): integer;
```

Description
Use AutoColWidth to calculate the column width automatically. The width is set to the max string width for text or numbers in the column.

3.11.14 TSheet.CalcCount

CalcCount is the maximum number of iterations used by the worksheet. Default is 100.

```
property CalcCount: word;
```

3.11.15 TSheet.CalcDimensions

Calculate worksheet dimensions.

```
procedure CalcDimensions;
```

Description
Use CalcDimensions to set the FirstCol, LastCol, FirstRow and LastRow properties to the size of the worksheet. CalcDimensions is called automatically before the worksheet is saved.

3.11.16 TSheet.Calculate

Calculate the value of a formula cell.

```
procedure Calculate(Col, Row: integer);
```

Description
Use Calculate to calculate the value of a formula. The result of the calculation is assigned to the cell.

See also
OnFunction

3.11.17 TSheet.Cell

Reads the cell as a TCell object.

```
property Cell[Col,Row: integer]: TCell;
```

Description
Use Cell to read the cell at Col and Row as a TCell object. You may use the Cell property in order to obtain the FormatIndex of a cell.

Cell is read-only.

3.11.18 TSheet.CellType

Reads the cell type of a cell.

```
property CellType[Col,Row: integer]: TCellType;
```

Description
Use CellType to read the cell type of the cell at Col and Row.

CellType is read-only.

3.11.19 TSheet.Charts

Charts.

```
property Charts: TXLSCharts;
```

Description
Charts is a contaioner for the charts assigned to the worksheet.

3.11.20 TSheet.ClearData

Clears (empties) the worksheet of all data.

```
procedure ClearData;
```

See also
DeleteCell, DeleteCells

3.11.21 TSheet.ColGutter

Size of the column gutter that appears above of the columns.

```
property ColGutter: integer;
```

3.11.22 TSheet.ColOutlineGutter

Maximum outline level for the column gutter.

```
property ColOutlineGutter: integer;
```

3.11.23 TSheet.ColumnFormats

List of formats for column formatting.

```
property ColumnFormats: TColumnFormats;
```

3.11.24 TSheet.DefaultColWidth

Default column width measured in characters. To change the width of an column, either use AutoColWidth method, or add a ColumnFormat.

```
property DefaultColWidth: word;
```

3.11.25 TSheet.DefaultRowHeight

Default row height measured in 1/20th of a character point, based on the selected font for the worksheet.

```
property DefaultRowHeight: word;
```

See also
RowHeights

3.11.26 TSheet.DeleteCell

Deletes a cell.

```
procedure DeleteCell(Col, Row: integer);
```

Description
Use DeleteCell to delete the cell at Col and Row.

3.11.27 TSheet.DeleteCells

Deletes a range of cells.

```
procedure DeleteCells(Col1, Row1, Col2, Row2: integer);
```

Description
Use DeleteCells to delete the cells between Col1,Row2 and Col2,Row2.

See also
DeleteCell, ClearData

3.11.28 TSheet.Delta

Maximum iteration change.

```
property Delta: double;
```

3.11.29 TSheet.FirstCol

The first used column of the worksheet.

```
property FirstCol: word;
```

Description
Use FirstCol to read the first used column of the worksheet. FirstCol is updated to it correct value

when the file is read or written, or when the method CalcDimensions is called.

3.11.30 TSheet.FirstRow

The first used row of the worksheet.

```
property FirstRow: word;
```

Description
Use FirstRow to read the first used row of the worksheet. FirstRow is updated to it correct value when the file is read or written, or when the method CalcDimensions is called.

3.11.31 TSheet.FixedCols

Number of fixed columns.

```
property FixedCols: byte;
```

Description
Use FixedCols to set the number of fixed columns to the left of the worksheet. Fixed columns are not scrolled with the rest of the worksheet. The area created can either be a scrollable or frozen pane, which depends on the setting of the Options property.

3.11.32 TSheet.FixedRows

Number of fixed rows.

```
property FixedRows: byte;
```

Description
Use FixedRows to set the number of fixed rows on the top of the worksheet. Fixed rows are not scrolled with the rest of the worksheet. The area created can either be a scrollable or frozen pane, which depends on the setting of the Options property.

3.11.33 TSheet.Hyperlink

Reads the hyperlink associated with a cell.

```
property Hyperlink[Col,Row: integer]: string;
```

Description
Use Hyperlink to read the hyperlink associated with the cell at Col and Row. If there is no hyperlink at the position, an empty string will be returned.

Hyperlink is read-only.

See also
WriteHyperlink

3.11.34 TSheet.LastCol

The last used column of the worksheet.

```
property LastCol: word;
```

Description
Use LastCol to read the last used column of the worksheet. LastCol is updated to it correct value when the file is read or written, or when the method CalcDimensions is called.

3.11.35 TSheet.LastRow

The last used row of the worksheet.

```
property LastRow: word;
```

Description
Use LastRow to read the last used row of the worksheet. LastRow is updated to it correct value when the file is read or written, or when the method CalcDimensions is called.

3.11.36 TSheet.MergedCells

List of merged cells.

```
property MergedCells: TMergedCells;
```

3.11.37 TSheet.Name

The name of the worksheet.

```
property Name: string;
```

See also
NameWideString

3.11.38 TSheet.NameWideString

The name of the worksheet.

```
property NameWideString: string;
```

Description
Use NameWideString to set the name of the worksheet as a wide (unicode) string.

See also
Name

3.11.39 TSheet.Notes

Cell notes (annotations).

```
property Notes: TNotes;
```

3.11.40 TSheet.Options

Worksheet options.

```
type TSheetOption =
(soGridlines,soRowColHeadings,soProtected,soAutoCalc,soR1C1Mode,soIteratio
n,soShowFormulas,soFrozenPanes,soShowZeros);
type TSheetOptions = set of TSheetOption;

property Options: TSheetOptions;
```

Value	Meaning
soGridlines	Grid lines are shown in the worksheet.
soRowColHeadings	Row and column headings are shown in the worksheet.
soProtected	The Protection flag for the worksheet is on.
soAutoCalc	Auto calc option is on.
soR1C1Mode	Cell references are shown in R1C1 mode.
soIteration	
soShowFormulas	Formulas are shown in cells, instead of formula results.
soFrozenPanes	Panes are frozen. See also FixedCols and FixedRows.
soShowZeros	Zeros are shown.

3.11.41 TSheet.PaintCell

Paint the cell on a canvas.

```
procedure PaintCell(Canvas: TCanvas; ARect: TRect; ACol, ARow: integer);
```

Description
Use PaintCell to draw the contents of the cell at ACol and ARow on Canvas. ARect is a clipping rectangle, nothing is painted outside it.

Example
This example shows how to handle the DrawCell event on a TDrawGrid object:

```
procedure TForm1.DrawGrid1DrawCell(Sender: TObject; ACol, ARow: Integer;
Rect: TRect; State: TGridDrawState);
begin
  if (ACol > 0) and (ARow > 0) then
    XLSReadWrite1.Sheets[0].PaintCell(DrawGrid1.Canvas,Rect,ACol - 1,ARow
- 1);
end;
```

3.11.42 TSheet.PrintSettings

Printing options.

```
property PrintSettings: TPrintSettings;
```

TPrintSettings properties

Property	Meaning
`Copies: word;`	Number of copies to print.
`ColsOnEachPage: string;`	Colums to repeat on each page when printing. The format is [Col1]:[Col2]. Columns are zero relative. Example 2:4, repeate column 3 to 4 on each page. If no columns are to be repeated, set the property to -1:-1.
`Footer: string;`	Footer text. In order to format text, the following codes can be used: **&L**=Align left. **&C**=Align center. **&R**=Align right. **&NN**=Set font size to NN points. **&"Fontname"**=Set font to Fontname. It is also possible to set the font to bold weight, but this is localized, ie the word used for "Bold" must be in the same language as the windows version which is used! English example: &"Arial,Bold". Swedish example: &"Arial,Fet". Example: &CThis text is center aligned&12&RThis text is right aligned in 12 points
`FooterMargin: double;`	Footer margin.
`Header: string;`	Header text. See Footer for text options.
`HeaderMargin: double;`	Header margin.
`HorizPageBreaks: THorizPageBreaks;`	Array of horizontal page breaks.
`MarginBottom: double;`	Bottom print margin.
`MarginLeft: double;`	Left print margin.
`MarginRight: double;`	Right print margin.
`MarginTop: double;`	Top print margin.
`Options: TPrintSetupOptions;`	Printing options, see below.
`RowsOnEachPage: string;`	Rows to repeat on each page when printing. The format is [Row1]:[Row2]. Rows are zero relative. Example 2:4, repeate row 3 to 4 on each page. If no rows are to be repeated, set the property to -1:-1.
`ScalingFactor: word;`	???
`StartingPage: word;`	First page to print.
`VertPageBreaks: TVertPageBreaks;`	Array of horizontal page breaks.

```
type TPrintSetupOption =
(psoLeftToRight,psoPortrait,psoNoColor,psoDraftQuality,psoNotes,psoRowColH
eading,psoGridlines,psoHorizCenter,psoVertCenter);
type TPrintSetupOptions = set of TPrintSetupOption;
```

Property	Meaning
psoLeftToRight	Print over, and then down.
psoPortrait	Portrait mode if set. Otherwise landscape mode.
psoNoColor	Print in black and white.
psoDraftQuality	Print draft quality.
psoNotes	Print cell notes.
psoRowColHeading	Print row and column headings.
psoGridlines	Print grid lines.
psoHorizCenter	Center sheet between horizontal margins.
psoVertCenter	Center sheet between vertical margins.

3.11.43 TSheet.RecalcFormulas

Set this property to true if formulas shall be marked for recalculation. As TXLSReadWrite don't calculate formulas, this is advised. Default is True.

```
property RecalcFormulas: boolean;
```

3.11.44 TSheet.RowGutter

Size of the row gutter that appears to the left of the rows.

```
property RowGutter: integer;
```

3.11.45 TSheet.RowOutlineGutter

Maximum outline level for the row gutter.

```
property RowOutlineGutter: integer;
```

3.11.46 TSheet.RowHeights

Reads or writes a row height.

```
property RowHeights[Index: integer]: integer;
```

Description
Use RowHeights to read or write the height of a row. If there is no row height assigned to a row, the value 255 is returned. Row heights are in units of 1/20th of a character point, based on the selected font for the worksheet.

See also
DefaultRowHeight

3.11.47 TSheet.SheetPictures

Pictures inserted into the worksheet.

```
property SheetPictures: TSheetPictures;
```

See also
TXLSPictures

3.11.48 TSheet.Validations

Cell validations used in the worksheet.

```
property Validations: TDataValidations;
```

3.11.49 TSheet.WorkspaceOptions

Workspace information.

```
type TWorkspaceOption =
(woShowAutoBreaks,woApplyStyles,woRowSumsBelow,woColSumsRight,woFitToPage,
woOutlineSymbols);
type TWorkspaceOptions = set of TWorkspaceOption;

property WorkspaceOption: TWorkspaceOption;
```

Value	Meaning
woShowAutoBreaks	Automatic page brakes are visible.
woApplyStyles	Automatic styles are applied to an outline.
woRowSumsBelow	Summary rows appear below detail in an outline.
woColSumsRight	Summary columns appear to the right of detail in an outline.
woFitToPage	The Fit option is on (Page Setup dialog box, Page tab).
woOutlineSymbols	Outline symbols are displayed.

3.11.50 TSheet.WriteBlank

Writes a blank cell.

```
procedure WriteBlank(Col,Row,FormatIndex: integer);
```

Description
Use WriteBlank to write a blank cell at Col and Row. The cell may be formatted by setting FormatIndex to the index of a format. If no format is to be used, set FormatIndex to -1. If there already is a cell at the position, it will be deleted.

See also
AsBlank, Formats

3.11.51 TSheet.WriteBoolean

Writes a boolean cell.

```
procedure WriteBlank(Col,Row,FormatIndex: integer; Value: boolean);
```

Description
Use WriteBoolean to write a boolean cell at Col and Row. The cell may be formatted by setting FormatIndex to the index of a format. If no format is to be used, set FormatIndex to -1. If there already is a cell at the position, it will be deleted.

See also
AsBoolean, Formats

3.11.52 TSheet.WriteError

Writes an error cell.

```
procedure WriteError(Col,Row,FormatIndex: integer; Value: TCellError);
```

Description

Use WriteError to write an error cell at Col and Row. The cell may be formatted by setting FormatIndex to the index of a format. If no format is to be used, set FormatIndex to -1. If there already is a cell at the position, it will be deleted.

See also
AsError, Formats

3.11.53 TSheet.WriteNumber

Writes a numeric cell.

```
procedure WriteNumber(Col,Row,FormatIndex: integer; Value: boolean);
```

Description
Use WriteNumber to write a number cell at Col and Row. The cell may be formatted by setting FormatIndex to the index of a format. If no format is to be used, set FormatIndex to -1. If there already is a cell at the position, it will be deleted.

See also
AsFloat, Formats

3.11.54 TSheet.WriteString

Writes a string cell.

```
procedure WriteString(Col,Row,FormatIndex: integer; Value: string);
```

Description
Use WriteString to write a string cell at Col and Row. The cell may be formatted by setting FormatIndex to the index of a format. If no format is to be used, set FormatIndex to -1. If there already is a cell at the position, it will be deleted.

See also
AsString, WriteWideString, Formats

3.11.55 TSheet.WriteWideString

Writes a string cell with the value as a wide string.

```
procedure WriteWideString(Col,Row,FormatIndex: integer; Value: string);
```

Description
Use WriteWideString to write a string cell at Col and Row. Value will be saved as a wide (multi byte) string. The cell may be formatted by setting FormatIndex to the index of a format. If no format is to be used, set FormatIndex to -1. If there already is a cell at the position, it will be deleted.

See also
AsString, WriteString, Formats

3.11.56 TSheet.WriteHyperlink

Writes a hyperlink.

```
procedure WriteHyperlink(Col,Row,FormatIndex: integer; Text,Link: string);
```

Description
Use WriteHyperlink to write a hyperlink at Col and Row. Text is the string shown in the cell. Link is the hyperlink (URL), which the web browser will be opened at, when the user clicks the text. The cell may be formatted by setting FormatIndex to the index of a format. If no format is to be used, set FormatIndex to -1. If there already is a cell at the position, it will be deleted and replaced by a string cell which contains the Text value.

Note: If the cell shall be formatted as a "standard" hyperlink (blue text and underlined), a format with these attributes must be created and the index of it provided in the FormatIndex parameter.

See also
Hyperlink, Formats

3.11.57 TSheet.WriteNumFormula

Writes a numeric formula cell.

```
procedure WriteNumFormula(Col,Row,FormatIndex: integer; Formula: string;
Value: double);
```

Description
Use WriteNumFormula to write a number formula cell at Col and Row. Formula is the formula of the cell. Value is the calculated result of the formula. The cell may be formatted by setting FormatIndex to the index of a format. If no format is to be used, set FormatIndex to -1. If there already is a cell at the position, it will be deleted.

See also
AsNumFormulaValue, WriteStrFormula, WriteBoolFormula, Formats

3.11.58 TSheet.WriteStrFormula

Writes a string formula cell.

```
procedure WriteStrFormula(Col,Row,FormatIndex: integer; Formula,Value:
string);
```

Description
Use WriteStrFormula to write a string formula cell at Col and Row. Formula is the formula of the cell. Value is the calculated result of the formula. The cell may be formatted by setting FormatIndex to the index of a format. If no format is to be used, set FormatIndex to -1. If there already is a cell at the position, it will be deleted.

See also
AsStrFormulaValue, WriteNumFormula, WriteBoolFormula, Formats

3.11.59 TSheet.WriteBoolFormula

Writes a boolean formula cell.

```
procedure WriteBoolFormula(Col,Row,FormatIndex: integer; Formula: string;
Value: double);
```

Description
Use WriteBoolFormula to write a boolean formula cell at Col and Row. Formula is the formula of the cell. Value is the calculated result of the formula. The cell may be formatted by setting FormatIndex to the index of a format. If no format is to be used, set FormatIndex to -1. If there already is a cell at the

position, it will be deleted.

See also
AsBoolFormulaValue, WriteNumFormula, WriteStrFormula, Formats

3.11.60 TSheet.Zoom

Zoom magnification.

```
property Zoom: word;
```

3.11.61 TSheet.ZoomPreview

Zoom magnification in page break preview.

```
property ZoomPreview: word;
```

3.12 TSheets

3.12.1 TSheets

unit
SheetData

TSheets is an ancestor of TCollection.
TSheets holds the TSheet objects.

3.12.2 TSheets.Items

Provides access to the TSheet objects.

```
property Items[Index: integer]: TSheet; default;
```

Description
Read Items to access a TSheet object.

See also
TSheet

3.13 TWorkbookData

3.13.1 TWorkbookData

Options for the workbook.

units
XLSUtils

Property	Description
Left: word	Horizontal position of the window, in units of 1/20th of a point.
Top: word	Vertical position of the window, in units of 1/20th of a point.
Width: word	Width of the window, in units of 1/20th of a point.

Height: word	Height of the window, in units of 1/20th of a point.
SelectedTab: word	The tab (worksheet) which is selected when the file is opened.
Options: TWorkbookOptions	Options, see below.

```
type TWorkbookOption = (woHidden,woIconized,woHScroll,woVScroll,woTabs);
     TWorkbookOptions = set of TWorkbookOption;

property Options: TWorkbookOptions;
```

Value	Meaning
woHidden	The window is hidden..
woIconized	The window is displayed as an icon.
woHScroll	The horizontal scroll bar is displayed.
woVScroll	The vertical scroll bar is displayed.
woTabs	The workbook tabs are displayed.

3.14 TXLSChart

3.14.1 TXLSChart

A workbook chart object.

unit
XLSChart

Description
TXLSChart represents a chart on the workbook.

See also
TXLSCharts

3.14.2 TXLSChart.CategoryTitle

Title for the charts category (x-axis).

```
property CategoryTitle: string;
```

Description
Use the CategoryTitle property for setting the text for the charts category (x-axis).

See also
Title

3.14.3 TXLSChart.Col1

The first column for the location of the chart.

```
property Col1: integer;
```

Description
Use the Col1 property to set the location for the first column where the chart is placed on the worksheet.

See also
Col2, Row1, Row2

3.14.4 TXLSChart.Col2

The last column for the location of the chart.

```
property Col2: integer;
```

Description
Use the Col2 property to set the location for the last column where the chart is placed on the worksheet.

See also
Col1, Row1, Row2

3.14.5 TXLSChart.Legend

Legend options .

```
property Legend: TChartLegend;
```

Description
Use the Legend property to set options for the charts legend.

TChartLegend properties

Property	Meaning
Include	True if a legend chall be included with the chart.
Placement	Placement for the legend. lpBottom = bottom of the chart. lpCorner = upper right corner. lpTop = top of the chart. lpRight = right of the chart. lpLeft = left of the chart.

3.14.6 TXLSChart.Name

The name of the TXLSChart object.

```
property Name: string;
```

3.14.7 TXLSChart.Row1

The first row for the location of the chart.

```
property Row1: integer;
```

Description
Use the Row1 property to set the location for the first row where the chart is placed on the worksheet.

See also
Col1, Col2, Row2

3.14.8 TXLSChart.Row2

The last row for the location of the chart.

```
property Row2: integer;
```

Description
Use the Row2 property to set the location for the last row where the chart is placed on the worksheet.

See also
Col1, Col2, Row1

3.14.9 TXLSChart.Series

The series that defines the source data for the chart.

```
property Series: TChartSeries;
```

Description
Use the Series property to add one or more series for the chart.

3.14.10 TXLSChart.Style

The style of the chart.

```
property Style: TChartStyle;
```

Description
Use the Style property to change the style of the chart.

TChartStyle values

Value	Meaning
csColumn	Verikal bars.
csColumn3d	Verikal bars, 3D.
csBar	Horizontal bars.
csBar3d	Horizontal bars, 3D.
csLine	Line.
csLineMark	Line with marks at each value.
csLine3d	Line, 3D.
csPie	Pie.
csPie3d	Pie, 3D.
csArea	Area.
csArea3d	Area, 3D.
csSurface	Surface. Requires two series.
csSurface3d	Surface, 3D. Requires two series.
csRadar	Radar.
csRadarArea	Radar with filled areas.

3.14.11 TXLSChart.Title

Title for the chart

```
property Title: string;
```

Description
Use the Title property for setting the text for the charts title.

See also
CategoryTitle

3.15 TXLSCharts

3.15.1 TXLSCharts

unit
XLSChart

TXLSCharts is an ancestor of TCollection.
TXLSCharts is a container for the TXLSChart objects.

3.15.2 TXLSCharts.Add

Add a new TXLSChart to the container.

```
function Add: TXLSChart;
```

Description
Use the Add method to add a new chart to the container.

3.15.3 TXLSCharts.Items

Provides access to the TXLSChart objects.

```
property Items[Index: integer]: TXLSChart; default;
```

Description
Read Items to access a TXLSChart object.

See also
TXLSChart

3.16 TChartSerie

3.16.1 TChartSerie

A serie for the chart.

unit
XLSChart

Description
A serie defines the source data for the chart. A chart needs at least one serie.

See also
TXLSChart

3.16.2 TChartSerie.Area

The source area for the serie.

```
property Area: string;
```

Description
Use the Area property to assign the source values for the serie. The Area property can be any valid Excel expression.

Note
The Area property must contain the sheet name. Example: "Sheet 1!A1:A8".

See also
Categories

3.16.3 TChartSerie.Categories

Defines category labels for a serie.

```
property Categories: string;
```

Description
Use the Categories property to assign category labels for the serie. The Categories property can be any valid Excel expression. If no categories are entered, the serie values will be labeled 1, 2, 3...

Note
The Categories property must contain the sheet name. Example: "Sheet 1!A1:A8".

See also
Area

3.16.4 TChartSerie.Color

The color used to for the serie.

```
property Color: TExcelColor;
```

3.16.5 TChartSerie.Name

The name of the TChartSerie object.

```
property Name: string;
```

3.16.6 TChartSerie.SerieName

The name of the serie.

```
property SerieName: string;
```

Description
Use the SerieName property to assign a name for the serie. Serie names are displayed on the right side of the chart.

3.17 TChartSeries

3.17.1 TChartSeries

unit
XLSChart

TChartSeries is an ancestor of TCollection.
TChartSeries is a container for the TChartSerie objects.

3.17.2 TChartSeries.Add

Add a new TChartSerie to the container.

```
function Add: TChartSerie;
```

Description

Use the Add method to add a new seriet to the container.

3.17.3 TChartSeries.Items

Provides access to the TChartSerie objects.

```
property Items[Index: integer]: ChartSerie; default;
```

Description
Read Items to access a TChartSerie object.

See also
TChartSerie

3.18 TColumnFormat

3.18.1 TColumnFormat

Formats one ore more columns.

unit
CellFormats

Description
Create a column format in order to format, or set the width of, one or more columns. Column formats are much more efficient to use if you want to format entire columns, instead of create individual formats (ie writing blank cells) for each cell. The column formats are placed in the TColumnFormats container.

See also
TColumnFormats

3.18.2 TColumnFormat.Col1

The first column for the format.

```
property Col1: integer;
```

Description
Set the Col1 property to the first column which shall be formatted.

See also
Col2

3.18.3 TColumnFormat.Col2

The last column for the format.

```
property Col2: integer;
```

Description
Set the Col2 property to the last column which shall be formatted.

See also
Col1

3.18.4 TColumnFormat.CollapsedOutline

True if the column range is collapsed in outlining.

```
property CollapsedOutline: boolean;
```

See also
OutlineLevel

3.18.5 TColumnFormat.FormatIndex

The format index used by the column format.

```
property FormatIndex: integer;
```

Description
Set the FormatIndex property to the TCellFormat you want to use for formatting the columns. If no CellFormat is needed, set FormatIndex to zero.

See also
TCellFormat

3.18.6 TColumnFormat.Hidden

If the column range shall be hidden.

```
property Hidden: boolean;
```

Description
Set the Hidden property to true if the column range shall be hidden.

3.18.7 TColumnFormat.OutlineLevel

Outline level of column range.

```
property OutlineLevel: integer;
```

See also
CollapsedOutline

3.18.8 TColumnFormat.Width

The width of each column in the column range.

```
property Width: integer;
```

Description
Set the Width property to the width for the columns in the column range. The width is in units of 1/256s of a character width.

3.19 TColumnFormats

3.19.1 TColumnFormats

unit
CellFormats

TColumnFormats is an ancestor of TCollection.
TColumnFormats is a container for the TColumnFormat objects.

3.19.2 TColumnFormats.Add

Add a new TColumnFormat to the container.

```
function Add: TColumnFormat; overload;
function Add(Col1,Col2,FormatIndex: integer): TColumnFormat; overload;
```

Description
Use the Add method to add a new column format to the container. Col1 and Col2 are the first and last columns. FormatIndex is the format index to be used.

3.19.3 TColumnFormats.ByColumn

Returns a column format by a column.

```
function ByColumn(Col: integer): TColumnFormat;
```

Description
Use the ByColumn method to find a column format associated with the Col parameter. If no format is found, -1 is returned.

3.19.4 TColumnFormats.ColWidth

Returns a column width for a column.

```
function ColWidth(Col: integer): integer;
```

Description
Use the ColWidth method to find the column width for the Col parameter column. If no format (width) is found, -1 is returned. This means that the column width is set to the default column width.

See also

DefaultColWidth, AutoColWidth

3.19.5 TColumnFormats.Items

Provides access to the TColumnFormat objects.

```
property Items[Index: integer]: TColumnFormat; default;
```

Description
Read Items to access a TColumnFormat object.

See also
TColumnFormat

3.20 TMergedCell

3.20.1 TMergedCell

Merges a cell area into one cell.

unit
SheetData

Description
TMergedCell represents a merged cell area, which acts like one large cell.

See also
TMergedCells

3.20.2 TMergedCell.Col1

The first column for the merged cells.

```
property Col1: integer;
```

Description
Set the Col1 property to the first column which shall be included in the merged cells area.

See also
Col2, Row1, Row2, Valid

3.20.3 TMergedCell.Col2

The last column for the merged cells.

```
property Col2: integer;
```

Description
Set the Col2 property to the last column which shall be included in the merged cells area.

See also
Col1, Row1, Row2, Valid

3.20.4 TMergedCell.Row1

The first row for the merged cells.

```
property Row1: integer;
```

Description
Set the Row1 property to the first row which shall be included in the merged cells area.

See also
Col1, Col2, Row2, Valid

3.20.5 TMergedCell.Row2

The last row for the merged cells.

```
property Row2: integer;
```

Description
Set the Row2 property to the last row which shall be included in the merged cells area.

See also
Col1, Col2, Row1, Valid

3.20.6 TMergedCell.Valid

Checks if the cell is valid.

```
function Valid: boolean;
```

Description
Use the Valid method to check if the cell area which makes up the merged cell not overlaps other merged cells. It the cell area not overlaps, Valid returns True.

See also
Col1, Col2, Row1, Row2

3.21 TMergedCells

3.21.1 TMergedCells

unit
SheetData

TMergedCells is an ancestor of TCollection.
TMergedCells is a container for the TMergedCell objects.

3.21.2 TMergedCells.Add

Add a new TMergedCell to the container.

```
function Add: TMergedCell;
```

Description
Use the Add method to add a new merged cell to the container.

3.21.3 TMergedCells.IsInMerged

Check if a column and row is in a merged cell.

```
function IsInMerged(Col,Row: integer): integer;
```

Description
Use the IsInMerged method to find a merged cell which the Col and Row parameters are in. The returned value is the index into the Items property. If no merged cell can be found,
-1 is returned.

See also
Items

3.21.4 TMergedCells.Items

Provides access to the TMergedCell objects.

```
property Items[Index: integer]: TMergedCell; default;
```

Description
Read Items to access a TMergedCell object.

See also
TMergedCell

3.22 TNote

3.22.1 TNote

TNote describes a cell note associated with a cell.

unit
Notes

See also
TNotes

3.22.2 TNote.Col1

The first column for the cell note.

```
property Col1: integer;
```

Description
Set the Col1 property to the first column which is used when the note is displayed.

See also
Col1Offset, Col2, Col2Offset, Row1, Row1Offset, Row2, Row2Offset, CellCol, CellRow

3.22.3 TNote.Col1Offset

The offset into the first column for the cell note.

```
property Col1Offset: integer;
```

Description
Set the Col1Offset property to the offset into the first column which is used when the note is displayed. The Col1Offset is in units of a 1/1024s of the cell width.

See also
Col1, Col2, Col2Offset, Row1, Row1Offset, Row2, Row2Offset, CellCol, CellRow

3.22.4 TNote.Col2

The last column for the cell note.

```
property Col2: integer;
```

Description
Set the Col2 property to the last column which is used when the note is displayed.

See also
Col1, Col1Offset, Col2Offset, Row1, Row1Offset, Row2, Row2Offset, CellCol, CellRow

3.22.5 TNote.Col2Offset

The offset into the last column for the cell note.

```
property Col2Offset: integer;
```

Description
Set the Col2Offset property to the offset into the first column which is used when the note is displayed. The Col2Offset is in units of a 1/1024s of the cell width.

See also
Col1, Col1Offset, Col2, Row1, Row1Offset, Row2, Row2Offset, CellCol, CellRow

3.22.6 TNote.Row1

The first row for the cell note.

```
property Row1: integer;
```

Description
Set the Row1 property to the first row which is used when the note is displayed.

See also
Col1, Col1Offset, Col2, Col2Offset, Row1Offset, Row2, Row2Offset, CellCol, CellRow

3.22.7 TNote.Row1Offset

The offset into the first row for the cell note.

```
property Row1Offset: integer;
```

Description
Set the Row1Offset property to the offset into the first row which is used when the note is displayed. The Row1Offset is in units of a 1/256s of the row height.

See also
Col1, Col1Offset, Col2, Col2Offset, Row1, Row2, Row2Offset, CellCol, CellRow

3.22.8 TNote.Row2

The last row for the cell note.

```
property Row2: integer;
```

Description
Set the Row2 property to the last row which is used when the note is displayed.

See also
Col1, Col1Offset, Col2, Col2Offset, Row1, Row1Offset, Row2Offset, CellCol, CellRow

3.22.9 TNote.Row2Offset

The offset into the first column for the cell note.

`property Row2Offset: integer;`

Description
Set the Row2Offset property to the offset into the last row which is used when the note is displayed. The Row2Offset is in units of a 1/256s of the row height.

See also
Col1, Col1Offset, Col2, Col2Offset, Row1, Row1Offset, Row2, CellCol, CellRow

3.22.10 TNote.CellCol

The column for the cell associated with the note.

`property CellCol: integer;`

Description
Set the CellCol property to the column of the cell associated with the note.

See also
CellRow

3.22.11 TNote.CellRow

The row for the cell associated with the note.

`property CellRow: integer;`

Description
Set the CellRow property to the row of the cell associated with the note.

See also
CellCol

3.22.12 TNote.Text

The text of the note.

`property Text: TStringList;`

Description
Use the Text property to read or write the strings of the note.

3.23 TNotes

3.23.1 TNotes

unit
Notes

TNotes is an ancestor of TCollection.
TNotes is a container for the TNote objects.

3.23.2 TNotes.Add

Add a new TNote to the container.

```
function Add: TNote;
```

Description
Use the Add method to add a new merged cell to the container.

3.23.3 TNotes.Items

Provides access to the TNote objects.

```
property Items[Index: integer]: TNote; default;
```

Description
Read Items to access a TNote object.

See also
TNote

3.24 TSheetPicture

3.24.1 TSheetPicture

TSheetPicture defines where a TXLSPicture is inserted on the worksheet.

unit
Picture

Description
Add SheetPictures in order to insert pictures on the worksheet. A TSheetPicture object links to a TXLSPicture trough the PictureFilename property.

See also
TSheetPictures

3.24.2 TSheetPicture.Col

The column where the picture shall be inserted.

```
property Col: integer;
```

Description
Set the Col property to the column where the picture shall be inserted on the worksheet.

See also
Row, Zoom

3.24.3 TSheetPicture.PictureName

The name of the TXLSPicture object which holds the actual picture.

```
property PictureName: string;
```

Description
Set the PictureName property to the name of the TXLSPicture object which holds the picture.

See also
TXLSPicture

3.24.4 TSheetPicture.Row

The row where the picture shall be inserted.

```
property Row: integer;
```

Description
Set the Row property to the Row where the picture shall be inserted on the worksheet.

See also
Col, Zoom

3.24.5 TSheetPicture.Zoom

Sets the magnification of the picture.

```
property Zoom: integer;
```

Description
Change the Zoom property in order to display the picture larger or smaller than it's actual size. Zoom is the percent change of the picture size. The default value is 100, which means that the picture is displayed in its original size.

See also
Col, Row, Zoom

3.25 TSheetPictures

3.25.1 TSheetPictures

unit
Picture

TSheetPictures is an ancestor of TCollection.
TSheetPictures is a container for the TSheetPicture objects.

3.25.2 TSheetPictures.Add

Add a new TSheetPicture to the container.

function Add: TSheetPicture;

Description
Use the Add method to add a picture to the container.

3.25.3 TSheetPictures.Items

Provides access to the TSheetPicture objects.

`property Items[Index: integer]: TSheetPicture; default;`

Description
Read Items to access a TSheetPicture object.

See also
TSheetPicture

3.25.4 TSheetPictures.Name

The name of the TXLSPicture object.

`property Name: string;`

Description
Change the Name property in order to set the name of the object. Name is used by TSheetPicture objects when refereeing to the object.

3.26 TDataValidation

3.26.1 TDataValidation

Defines a validation for a range of cells.

unit
Validate

Description
TDataValidation controls an input criteria for a cell or range of cells.

See also
TDataValidations

3.26.2 TDataValidation.Areas

The areas for which the validation shall be applied.

`property Areas: TAreas;`

Description
TAreas is a container for TArea objects. Each TArea object has four properties, Col1, Col2, Row1 and Row2 which defines an area for where the validation shall be applied.

3.26.3 TDataValidation.ErrorMsg

Message to be displayed when an invalid value is entered.

```
function ErrorMsg: string;
```

Description
Use ErrorMsg to define the error message which is displayed when a value is entered that not meets the validation criteria. If no value is defined, a default message is shown.

See also
ErrorTitle, InputMsg, InputTitle

3.26.4 TDataValidation.ErrorTitle

The title (caption) of the message box of the error message.

```
function ErrorTitle: string;
```

Description
Use ErrorTitle to set the caption of the message box for which the error message is shown in. f no value is defined, a default title is shown.

See also
ErrorMsg, InputMsg, InputTitle

3.26.5 TDataValidation.Expression1

Expression of value used by the validation criteria.

```
property Expression1: string;
```

Description
Use Expression1 to define the value or formula used to limit the input range. If two values are needed, the other is in Expression2. The expression may be any valid Excel value or formula.

See also
ValidationOperator, ValidationStyle, ValidationType, Expression2

3.26.6 TDataValidation.Expression2

Expression of value used by the validation criteria.

```
property Expression2: string;
```

Description
Use Expression2 to define the value or formula used to limit the input range. Expression2 is only used when two values are needed for the criteria. The expression may be any valid Excel value or formula.

See also

ValidationOperator, ValidationStyle, ValidationType, Expression1

3.26.7 TDataValidation.InputMsg

Hint text which is shown when the cursor is in the cell area(s).

```
function InputMsg: string;
```

Description
Use InputMsg to define a hint text which is shown when the cursor is in one of the areas for which the validation applies. Of no text is defined, no text is shown.

See also
ErrorMsg, ErrorTitle, InputTitle

3.26.8 TDataValidation.InputTitle

Title of input message.

```
function InputTitle: string;
```

Description
Use InputTitle to define a title (firs row in bold) in the hint box shown for the validation.

See also
ErrorMsg, ErrorTitle, InputMsg

3.26.9 TDataValidation.ValidationOperator

What kind of validation that is to be used.

```
type TValidationOperator =
(voBetween,voNotBetween,voEqual,voNotEqual,voGreater,voLess,voGreateEqual,
voLessEqual);

property ValidationOperator: TValidationOperator;
```

Description
Use ValidationOperator in order to define what kind of values that may be entered.

Value	Meaning
voBetween	Value is between Expression1 and Expression2.
voNotBetween	Value is not between Expression1 and Expression2.
voEqual	Value is equal to Expression1.
voNotEqual	Value is not equal to Expression1.
voGreater	Value is greater than Expression1.
voLess	Value is less than Expression1.
voGreateEqual	Value is greater or equal to Expression1.
voLessEqual	Value is less or equal to Expression1.

See also
ValidationStyle, ValidationType, Expression1, Expression2

3.26.10 TDataValidation.ValidationStyle

What kind of action that takes place when the entered value is not met by the validation criteria.

```
type TValidationStyle = (vsStop,vsWarning,vsInfo);
```

```
property ValidationStyle: TValidationStyle;
```

Description
Use ValidationStyle to define what kind of action that takes place when the entered value falls outside the validation criteria.

Value	Meaning
vsStop	The value is not accepted.
vsWarning	A warning is shown, but the value is accepted.
vsInfo	An information box is shown, but the value is accepted.

See also
ValidationOperator, ValidationType, Expression1, Expression2

3.26.11 TDataValidation.ValidationType

What kind of values that may be entered.

```
type TValidationType =
(vtAny,vtInteger,vtNumber,vtList,vtDate,vtTime,vtTextLength,vtCustom);
```

```
property ValidationType: TValidationType;
```

Description
Use ValidationType to set what kind of values the user may enter in order to meet the validation criteria.

Value	Meaning
vtAny	Any value. This is this is the same as no validation, but any defined messages are shown.
vtInteger	An integer value.
vtNumber	Any number.
vtList	A list of values. Expression1 defines the source area of the list.
vtDate	A date.
vtTime	A time.
vtTextLength	Limited text length.
vtCustom	A formula defined by Expression1.

See also
ValidationOperator, ValidationStyle, Expression1, Expression2

3.27 TDataValidations

3.27.1 TDataValidations

unit
Validate

TDataValidations is an ancestor of TCollection.
TDataValidations is a container for the TDataValidation objects.

3.27.2 TDataValidations.Add

Add a new TDataValidations to the container.

```
function Add: TDataValidation;
```

Description
Use the Add method to add a new validation to the container.

3.27.3 TDataValidations.Items

Provides access to the TDataValidation objects.

```
property Items[Index: integer]: TDataValidation; default;
```

Description
Read Items to access a TDataValidation object.

See also
TDataValidation

3.28 TVertPagebreak

3.28.1 TVertPagebreak

Defines a vertical page break.

unit
SheetData

Description
TVertPagebreak defines a vertical pagebreak when printing the worksheet.

See also
TVertPagebreaks

3.28.2 TVertPagebreak.Col

Column for vertical pagebreak.

```
property Col: integer;
```

Description
Set Col to the column for the vertical page break.

See also
Row1, Row2

3.28.3 TVertPagebreak.Row1

First row for vertical pagebreak.

```
property Row1: integer;
```

Description
Set Row1 to the first row for the vertical page break.

See also
Col, Row2

3.28.4 TVertPagebreak.Row2

Last row for vertical pagebreak.

```
property Row2: integer;
```

Description
Set Row2 to the last row for the vertical page break. If all rows shall be included, set it to 65535.

See also
Col, Row1

3.29 TVertPagebreaks

3.29.1 TVertPagebreaks

unit
SheetData

TVertPagebreaks is an ancestor of TCollection.
TVertPagebreaks is a container for the TVertPagebreak objects.

3.29.2 TVertPagebreaks.Add

Add a new TVertPagebreak to the container.

`function` `Add:` TVertPagebreak;

Description
Use the Add method to add a new page break to the container.

3.29.3 TVertPagebreaks.Items

Provides access to the TVertPagebreak objects.

`property` `Items[Index: integer]:` TVertPagebreak; **`default`**;

Description
Read Items to access a TVertPagebreak object.

See also
TVertPagebreak

3.30 THorizPagebreak

3.30.1 THorizPagebreak

Defines a horizonyal page break.

unit
SheetData

Description
THorizPagebreak defines a horizontal pagebreak when printing the worksheet.

See also
THorizPagebreaks

3.30.2 THorizPagebreak.Col1

First column for vertical pagebreak.

`property` `Col1:` **`integer`**;

Description
Set Col1 to the first column for the hirizontal page break.

See also
Col2, Row

3.30.3 THorizPagebreak.Col2

Last column for vertical pagebreak.

property `Col2:` **integer**;

Description
Set Col2 to the last column for the vertical page break. If all columns shall be included, set it to 255.

See also
Col1, Row

3.30.4 THorizPagebreak.Row

Row for horizontal pagebreak.

property `Row:` **integer**;

Description
Set Row to the row for the horizontal page break.

See also
Col1, Col2

3.31 THorizPagebreaks

3.31.1 THorizPagebreaks

unit
SheetData

THorizPagebreaks is an ancestor of TCollection.
THorizPagebreaks is a container for the THorizPagebreak objects.

3.31.2 THorizPagebreaks.Add

Add a new THorizPagebreak to the container.

function `Add:` THorizPagebreak;

Description
Use the Add method to add a new page break to the container.

3.31.3 THorizPagebreaks.Items

Provides access to the THorizPagebreak objects.

property `Items[Index: integer]:` THorizPagebreak; **default**;

Description
Read Items to access a THorizPagebreak object.

See also
THorizPagebreak

3.32 TCell

3.32.1 TCell

TCell is a storage for a cell value of a worksheet.
The only property of interest is the FormatIndex property of the TCell object. This may be used as an index into the Formats property in order to check how a cell is formatted.

3.33 Utilities

unit
XLSUtils

Utilitie functions.

function `ColRowToRefStr(ACol,ARow: integer; AbsCol,AbsRow: boolean):` **string**;

Converts a cell reference, ACol and ARow, to a reference string. Set AbsCol or AbsRow to True if the reference shall be absolute, i.e. preceded with then '$' characther.
Example: `ColRowToRefStr(1,2,False,False)` will return the string "B3".
Example: `ColRowToRefStr(1,2,True,True)` will return the string "$B$3".

function `AreaToRefStr(Col1,Row1,Col2,Row2: integer; AbsCol1,AbsRow1,AbsCol2,AbsRow2: boolean):` **string**;

Converts an area reference, Col1, Row1,Col2 and Row2, to a reference string. Set AbsXXX to True if the reference shall be absolute, i.e. preceded with then '$' characther.
Example: `AreaToRefStr(1,2,5,9,False,False,False,False)` will return the string "B3:F10".
Example: `AreaToRefStr(1,2,5,9,True,True,True,True)` will return the string "$B$3:$F$10".

3.34 Excel Number Format Strings

Excel Number Format Strings are in the same style as in Excel, but with the following important exception. The strings always uses the characters that is used in the English Excel versions, even if they originates from a non-English Excel version. Format strings written with TXLSReadWriteII will be shown correct in there nationalized format in Excel. This translation is handled by Excel. For more details on Format Strings, see the Excel documentation.

Example of format strings:

```
'0.00',
'#,##0',
'(#,##0.00_);[Red](#,##0.00)',
'_(* #,##0_);_(* (#,##0);_(* "-"_);_(@_)',
```

Desctription of format charcters.

Specifier	**Displays**
. (dot)	Decimal separator
, (comma)	Thousands separator
/	Date separator
:	Time separator
yy	

yyyy	Year
m mm mmm mmmm	Month
d dd ddd dddd	Day
h hh	Hour
m mm	Minute
s ss	Second

3.35 TExcelColor

Excel colors.

```
type TExcelColor = (xc0,xc1...xc63);
```

Description
Excel uses an common color palette with 64 colors. TExcelColor is the index into this color palette. The value xcAutomatic means automatic color (or no color at all).

Use the function XColorToTColor to convert a TExcelColor value to it's TColor value.

```
function XColorToTColor(XC: TExcelColor): TColor;
```

3.36 TCellBorderStyle

```
type TCellBorderStyle =
(cbsNone,cbsThin,cbsMedium,cbsDashed,cbsDotted,cbsThick,
                             cbsDouble,cbsHair,cbsMediumDashed,cbsDashDot,cbsM
ediumDashDot,
                             cbsDashDotDot,cbsMediumDashDotDot,cbsSlantedDashD
ot);
```

3.37 TCellError

type TCellError = (errError,errNull,errDiv0,errValue,errRef,errName,errNum,errNA);

Value	Description
errError	Internal, no error defined.
errNul	#NULL!
errDiv0	#DIV/0!
errValue	#VALUE!
errRef	#REF!
errName	#NAME!
errNum	#NUM!

errNA	N/A

3.38 TCellType

```
type TCellType =
(ctNone,ctBlank,ctRK,ctInteger,ctFloat,ctString,ctBoolean,ctError,ctNumberFormula,ctStringFormula,ctBooleanFormula);
```

Top Level Intro

This page is printed before a new
top-level chapter starts

Part IV

4 TXLSDbRead

4.1 TXLSDbRead

TXLSDbRead is a component for importing any Dataset into TXLSReadWriteII.

Unit
XLSDbRead

4.2 TXLSDbRead.Column

The column where the table is inserted.

```
property Column: integer;
```

Description
Column is the column in TXLSReadWriteII where the first field from the dataset is inserted.

See also
Row, Sheet

4.3 TXLSDbRead.Dataset

The dataset from where the data is read.

```
property Dataset: TDataset;
```

Description
Assign to the Dataset property the dataset from where the data shall be read.

See also
Read

4.4 TXLSDbRead.ExcludeFields

Fields which shall be excluded from the data read.

```
property ExcludeFields: TStrings;
```

Description
Add to the ExcludeFields property names of fields which shall be excluded when reading the dataset.

See also
IncludeFields

4.5 TXLSDbRead.IncludeFields

Fields which shall be included from the data read.

```
property IncludeFields: TStrings;
```

Description
Add to the IncludeFields property the names of the fields which shall be included when reading the dataset. If no field names are given, all fields from the dataset are included.

See also
ExcludeFields

4.6 TXLSDbRead.IncludeFieldnames

If fieldnames shall be included.

```
property IncludeFieldnames: boolean;
```

Description
Set the IncludeFieldnames property to true if the names of the fields from the dataset shall be inserted on the first row.

4.7 TXLSDbRead.Row

The row where the table is inserted.

```
property Row: integer;
```

Description
Row is the row in TXLSReadWriteII where the first record from the dataset is inserted.

See also
Column, Sheet

4.8 TXLSDbRead.Sheet

The worksheet where the table is inserted.

```
property sheet: integer;
```

Description
Sheet is the worksheet in TXLSReadWriteII where the records from the dataset are inserted.

See also
Column, Row

4.9 TXLSDbRead.XLS

The TXLSReadWriteII object where the data is inserted.

```
property XLS: TXLSReadWriteII;
```

4.10 TXLSDbRead.Read

Reads the data from the dataset.

```
procedure Read;
```

Description
Call the Read method in order to read the data from the dataset.

See also
Dataset

Top Level Intro

This page is printed before a new
top-level chapter starts

Part V

5 TXLSExportHTML

5.1 TXLSExportHTML

TXLSExportHTML is a component for exporting the cells in a TXLSReadWriteII object to a HTML file.

Unit
XLSExportHTML

5.2 TXLSExportHTML.Col1

The first column from where to export.

```
property Col1: integer;
```

Description
Set Col1 to the first column of the area from where to export. Set Col1 to -1 for the first column with data.

See also
Col2, Row1, Row2

5.3 TXLSExportHTML.Col2

The last column from where to export.

```
property Col2: integer;
```

Description
Set Col2 to the last column of the area from where to export. Set Col1 to -1 for the last column with data.

See also
Col1, Row1, Row2

5.4 TXLSExportHTML.Filename

The name of the exported file.

```
property Filename: string;
```

5.5 TXLSExportHTML.Row1

The first row from where to export.

```
property Row1: integer;
```

Description
Set Row1 to the first row of the area from where to export. Set Row1 to -1 for the first row with data.

See also
Col1, Col2, Row2

5.6 TXLSExportHTML.Row2

The last row from where to export.

```
property Row2: integer;
```

Description
Set Row2 to the last row of the area from where to export. Set Row2 to -1 for the last row with data.

See also
Col1, Col2, Row1

5.7 TXLSExportHTML.SaveToStream

Saves the data to a stream.

```
procedure SaveToStream(Stream: TStream);
```

Description
Call the SaveToStream method in order to write the HTML data a stream object.

See also
Write

5.8 TXLSExportHTML.TABLE

Containes properties for the TABLE tags in the HTML file.

The following properties can be defined:
BORDER, CELLPADDING and CELLSPACING.

5.9 TXLSExportHTML.XLS

The TXLSReadWriteII object where the data is read.

```
property XLS: TXLSReadWriteII;
```

5.10 TXLSExportHTML.Write

Writes the data to the file.

```
procedure Write;
```

Description
Call the Write method in order to write the data to the HTML file.

See also
Filename

Index

Endnotes 2... (after index)

Back Cover