VFPxWorkbookXlsx Documentation

Class for Reading/Writing XLSX Format Files

Overview

VFPxWorkbookXlsx class was written to remove the need to automate an installed version of Microsoft Excel 2010 or higher in order to be able to create XLSX format spreadsheets with cell formatting and formulas. Several methods are provided to write a table or grid to a workbook sheet.

Additionally, VFPxWorkbookXlsx class has the ability to read an existing XLSX file and load the workbook into the internal working cursors (all cursors are preceeded with 'xl_'); methods are provided to return the cell values and formatting. The field types are determined from the cell formatting. These cursors can then be queried to be able to extract the worksheet cell information. All cursors are now contained in a private DataSession of the VFPxWorkbookXlsx class.

Support for Visual FoxPro versions 8.0 and 9.0 are provided; earlier versions are not supported due the inclusion of TRY-CATCH commands (if these code locations are refactored then the class should support earlier versions of VFP; note that the use of TRY-CATCH is also a design solution in some cases to catch XML errors due to non-existing nodes or for the datasession determination). The class is implemented as a Label baseclass so that when added to a container such as a form, the class name is provided in the Caption property for visual identification in the Design-time editor of VFP (Visible property is set to False).

VFPxWorkbookXlsx class has the following features:

- Assign values to cells
    - Numeric
    - Boolean
    - Date
    - Date-time
    - Character
- Cell numeric formatting
    - Number
    - Decimal places
    - Currency
    - Date layout
    - Date-time layout
    - Support for custom defined numeric formatting
- Apply formatting to the cells
    - Borders (top, left side, right side, and bottom)
    - Borders (left-to-right slant and right-to-left slant)
    - Border thickness, type (i.e., single, double, etc.)
    - Border color
    - Background color
    - Font name, size, and style
    - Word-wrapping
    - Text/numeric alignment
- Formula assignment to cells

- Merging and unmerging of cells
- Row height and column width
- Multiple sheets
    - Assign/rename sheet names
- Sheet Print setup
    - Orientation
    - Page scaling or sheets to a page count (horizontal and vertical)
    - Paper size (standard and custom)
- Sheet headers and footers for printing
    - First page, odd/even pages
    - Left section, center section and right section texts
    - Font support
- Workbook properties
    - Author/Creator
    - Company Name
- Read existing XLSX workbooks
    - Load into working cursors
    - Set cell datatypes based on cell format
- Write tables or grids directly to XLSX workbooks via a single method
    - Support for multiple sheets
    - Column width of sheets set by column width of grid

Methods / Events / Properties Summary

Method Name	Description
AddAutoFilter	Adds a filter to the column range
AddColumnFilter	Sets the specific filter for a column
AddCustomNumericFormat	Adds a new definition for a numeric format
AddGroupByColumn	Adds a column group level to the selection
AddGroupByRow	Adds a row group level to the selection
AddHyperLinkFile	Adds a new hyperlink to an external file
AddHyperLinkSheet	Adds a new hyperlink to another cell range within the workbook
AddImage	Adds an image to the sheet
AddIndexColor	Adds a new indexed color definition to the workbook
AddInLineFontObject	Adds an in-line character definition to the base in-line font definition object
AddMruColor	Adds a custom defined MRU color to the workbook
AddNamedRange	Adds a new named range of cells
AddNumericFormat	Adds a new definition for a numeric format (full format must be specified) [retained for backward compatibility]
AddSheet	Adds a new sheet to the workbook
AddStyleBorders	Adds to the style definition cell border formatting
AddStyleFill	Adds to the style definition cell fill formatting
AddStyleFont	Adds to the style definition cell font formatting
AddStyleHorizAlignment	Adds to the style definition cell horizontal alignment formatting
AddStyleIndent	Adds to the style definition cell indent formatting
AddStyleNumericFormat	Adds to the style definition cell numeric formatting
AddStyleProtection	Sets the style's protection values (locked and hidden)
AddStyleTextRotation	Adds to the style definition cell text rotation formatting
AddStyleVertAlignment	Adds to the style definition cell vertical alignment formatting
AddStyleWordWrap	Adds to the style definition cell word wrap formatting
CellFormatPainter	Copies the selected cell format to the specified range of cells
ClearAutoFilter	Clears the column filter for the sheet
ClearCellValidation	Removes any cell validations
ClearCellValue	Clears the value from the selected cell
ClearNamedRange	Removes the named range from the workbook
ColumnAsciiToIndex	Converts a Excel notation column reference (ASCII character) to an numeric (integer) column reference

Method Name	Description
ColumnIndexToAscii	Converts a numeric (integer) column reference to an ASCII character column reference
ConvertPixelsToExcelUnits	Converts pixels in VFP to Excel units for column widths
ConvertRangeToColumnRowValues	Converts a given range notation to row and column values
CopyStyle	Copies the style to a new style Id
CreateFormatStyle	Creates a new formatting style definition to be applied to cells
CreateInLineFormatText	Creates the base in-line font object for assigning a text string in a cell to have its characters to be individually formatted
CreateWorkbook	Creates a new workbook
DebugOutput	used for debugging
DeleteAllWorkbooks	Deletes all workbook Ids
DeleteHyperLink	Deletes the selected hyperlink from the sheet
DeleteImage	Deletes an image from the sheet
DeleteSheet	Deletes the workbook sheet
DeleteWorkbook	Deletes the workbook Id
Demo	Demo code examplesof the various features of this class
FreezePanes	Provides for freezing the upper rows and left columns for scrolling
GetCellAlignment	Returns the cell alignment
GetCellBorders	Returns the cell border info
GetCellDataType	Returns the cell data type; this is based on the character expression or the cell format.
GetCellFill	Returns the fill info for the cell
GetCellFont	Returns the cell font settings
GetCellFormula	Returns the cell formula expression
GetCellIndent	Returns the cell indentation
GetCellNumberFormat	Returns the format code for the selected cell
GetCellNumberFormatText	Returns the format text for the selected cell
GetCellStyle	Returns the assigned cell style Id value
GetCellTextRotation	Returns the cell text rotation
GetCellValidation	Gets the cell validation formula settings
GetCellValue	Returns the value from the selected cell
GetCellWordWrap	Returns the cell word wrap setting
GetColumnHidden	Returns the column hidden setting
GetColumnWidth	Returns the width of the selected column
GetCustomNumericFormat	Returns the specified numeric custom format code

Method Name	Description
GetCustomPaperSize	Gets the values for the custom paper size
GetDisplayGridLines	Gets the display setting for showing/hiding grid lines in the sheet
GetImageDimensions	Gets the image height and width dimensions for inserting into a sheet
GetImageRelationshipId	Gets the relationship Id for an image based on the workbook, sheet and position
GetInLineFormatText	Gets the in-line formatted text definition for a cell text string
GetLastColumnInRow	Returns the max column number for a given row in a sheet
GetLastRowNumber	Returns the last row number in the sheet
GetMaxColumnNumber	Returns the max column number for a sheet
GetNumberOfSheets	Returns the number of defined sheets for the given workbook id.
GetPaperSize	Gets the paper size for the selected sheet
GetPrintOrientation	Gets the print orientation for the sheet output
GetRowMaxColumn	Returns the max column number for a given row in a given sheet
GetSheetIndex	Gets the internal sheet index from the sheet name for a given workbook
GetSheetName	Returns the sheet name
GetSheetProtection	Returns the sheet protection settings in an object
GetSheetRowValues	Returns the cell values for the given row
GetSheetScale	Gets the sheet printing scale
GetStyleFormatId	Gets the format style Id for the given numeric format, font format, and fill format. Will dynamically create a new style if it does not exist.
GetValidation	Returns an object with the validation definition
GetValidationList	Returns an object with the list of validations for the workbook/sheet
GetWorkbook	Gets the workbook Id
GetWorkbookProtection	Sets the workbook protection settings
GetWorkbookFileName	Gets the workbook file name
GetWorkbookSheets	Gets the sheet information for a workbook
InsertCell	Inserts a new cell into the sheet
InsertColumn	Inserts a new column into the sheet
InsertRow	Inserts a new row into the sheet
IsCellFormula	Determines if the cell contains a formula
MergeCells	Provides for merging cells into a single cell
OpenCreatedXlsxFile	Opens the selected workbook in the default program via ShellExecute Win API
OpenXlsxWorkbook	Opens the passed XLSX workbook and loads the internal cursors with the content

Method Name	Description
OpenXlsxWorkbookShee	Opens a selected worksheet in a XLXS workbook; always sets the opened sheet as sheet1
ParseString	Replacement for GETWORDNUM function (fixes problem of parsing a string that has a null value for one of the tokens)
RenameSheet	Renames the selected sheet in the workbook
ResetColumnWidth	Resets the column width to the default of Excel
SaveGridToWorkbook	Saves the passed grid to a workbook in xlsx file format. Uses the grid column widths to set the workbook column widths. Adds a new sheet for each passed grid if the same workbook name.
SaveGridToWorkbookEx	Saves the passed grid to a workbook in xlsx file format by writing directly to the XLSX files and does not write to the internal cursors; hence, this is the fastest way to create a XLSX file from a grid.
SaveTableToWorkbook	Saves the passed table to a workbook in xlsx file format. Adds a new sheet for each passed table if the same workbook name.
SaveTableToWorkbookEx	Saves the passed table to a workbook in xlsx file format by writing directly to the XLSX files and does not write to the internal cursors; hence, this is the fastest way to create a XLSX file from a table or cursor. You can also pass an array of the fields that are to be included in the export.
SaveWorkbook	Saves the selected workbook to xlsx file format based on the name set at creation of the workbook
SaveWorkbookAs	Saves the selected workbook to xlsx file format with the supplied file name; resets the workbook file name for future saves
SetCellAlignment	Sets the cell alignment (vertical and horizontal)
SetCellBorder	Sets the cell border; each border is drawed with the same style and color
SetCellBorderEx	
SetCellBorderRange	Sets the cell border for a range of cells; each border is drawed with the same style and color
SetCellFill	Sets the cell fill color (background)
SetCellFillRange	Sets the cell fill color (background) for a range of cells
SetCellFont	Sets the cell format
SetCellFontRange	Sets the cell format for a range of cells
SetCellFormula	Sets the cell formula
SetCellIndent	Sets the cell indentation
SetCellInLineFormatText	Saves an in-line text definition for a text string to a cell
SetCellNumberDecimals	sets the number of decimals to be displayed (used with SetCellNumberFormat)
SetCellNumberFormat	Sets the numeric format for the cell value

Method Name	Description
SetCellNumberFormatRange	Sets the numeric format for a range of cell values
SetCellStyle	Sets the cell style Id to a selected cell
SetCellStyleRange	Sets the cell style Id to a selected cell range of rows/columns
SetCellTextRotation	Sets the cell text rotation
SetCellValidation	Sets cell validation
SetCellValue	Sets the cell value. The data type is set by the data type of the value to be set (determined via VARTYPE() function)
SetCellWordWrap	Sets the cell word-wrap value
SetCellWordWrapRange	Sets the cell word-wrap value for a range of cells
SetColumnBestFit	Sets the column width to best fit (this method is not yet fully working and is not currently saved in the sheet).
SetColumnHidden	Sets the column hidden setting
SetColumnWidth	Sets the selected column width
SetColumnWidthRange	Sets the column width for a range of columns
SetCustomPaperSize	Sets the paper size based on custom dimensions
SetDisplayGridLines	Sets the display setting for showing/hiding grid lines in the sheet
SetHeaderFooterSetup	Sets the properties for the header /footer in the sheet (Align to margins, different first page, different odd/even pages, and scale with print). This method must be set before calling SetHeaderFooterText() method.
SetHeaderFooterText	Sets the header text
SetPaperSize	Sets the paper size for the selected sheet
SetPrintFitToHeight	Number of vertical pages to fit on
SetPrintFitToWidth	Number of horizontal pages to fit on
SetPrintOrientation	Sets the printer orientation for sheet output
SetRowHeight	Sets the selected row height
SetRowHeightRange	Sets the selected row height
SetSheetGroupSettings	Sets the row and column summary settings (roll-up or roll-down)
SetSheetMargins	Sets the margins of the sheet
SetSheetProtection	Sets the sheet protection settings
SetSheetScale	Sets the print scale; must be between 10 and 400; i.e. 10=10%, 50=50%, 100=100%, 175=175%, etc.
SetSheetVisibility	Set the selected sheet visiblity in the workbook
SetTabColor	Sets the tab color of the selected sheet in the workbook
SetWorkbookProtection	Gets the workbook protection settings
UnFreezePanes	Removes all of the panes that are frozen (top and side)

Method Name	Description
UnGroupByColumn	Removes a column group level from the selection
UnGroupByRow	Removes a row group level from the selection
UnMergedCells	Removes the merged cells restoring to individual cells

Event Name	Description
OnDestroy	Called by Destroy Event; for placing user code
OnInit	Called by Init Event; for placing user code
OnShowErrorMessage	Called for displaying a user message when an error occurs. Use BINDEVENTS to bind to this event.
OnShowStatusMessage	Called for displaying a user message during the opening of an existing workbook (xlsx) file. Use BINDEVENTS to bind to this event.

Property Name	Description
AutoTrimSheetName	Indicates whether to auto-trim the sheet name if too long
CodePage	CodePage to use for the Strings cursor
CompanyName	Company name in workbook properties
CreatorName	Creator in workbook properties
Debug	Sets debugging mode
DeclareWinAPI	Boolean to declare the needed Win32 API functions called in Init()
DefaultFont	Default font name
DefaultFontSize	Default font size
DefaultSheetName	Default sheet name
ErrorLevelId	Error level Id that has occurred (see OnErrorMessage() event for id values assigned)
ExcelXlsxRelease	Release version of class
SaveCurrencyAsNumeric	Indicates whether to save a currency value as a currency value or as a numeric value [Boolean]
Subject	Subject in workbook properties
Title	Title in workbook properties
TrueFalseValue	The value to display in the cell for a boolean field type; pipe delimited list of the true value followed by the false value
UserName	Name of person stored in XLSX document as last edit

Creating Workbook Files

The following methods can be used to create a workbook:

- `CreateWorkbook()`
- `SaveGridToWorkbookEx()`
- `SaveTableToWorkbookEx()`
- `SaveGridToWorkbook()`
- `SaveTableToWorkbook()`

The first method, `CreateWorkbook()`, above will create an empty workbook. You have to add sheets and cell values using `AddSheet()` and `SetCellValue()` or `SetCellFormula()` (see the `Demo()` method in the class for examples). This allows for a workbook sheet to be populated as needed by the develper's requirements. Any formatting can also be added as required using the methods available in this class. Once the sheets and cell values and formatting has been assigned, use the method `SaveWorkbook()` to save the workbook as a XLXS file.

The second method, `SaveGridToWorkBook()`, allows for creating a workbook from a VFP grid and saves the grid rows/columns values to the internal xl_* cursor tables. Hidden columns can be optionally omitted in the export by parameter value. This method has a parameter to save the workbook to a XLXS file directly or not (if you do not save directly with the parameter then you must explicity call the `SaveWorkbook()` method to save to a XLSX file).

The third method, `SaveGridToWorkBookEx()`, is similar to the second but only creates a XLSX file. This method does not write to the internal xl_* cursors and instead writes directly to the workbook xml files using FWRITE() command and is very fast. This method also takes the formatting from the grid columns.

The fourth method, `SaveTableToWorkBook()`, allows for creating a workbook from a table or cursor and saves the field values to the internal xl_* cursor tables. This method has a parameter to save the workbook to a XLXS file directly or not (if you do not save directly with the parameter then you must explicity call the `SaveWorkbook()` method to save to a XLSX file).

Using either the `SaveGridToWorkBook()` or `SaveTableToWorkBook()` methods without saving to a XLSX file directly allows you to add more to the workbook since the field values are saved to the internal xl_* cursor tables. You can now use any of the class methods to add formulas, set cell formatting (color, borders, font, etc.), set column/row groupings, add more sheets, and more. Repeatly calling the `SaveTableToWorkBook()` or `SaveGridToWorkBook()` with the same workbook parameter value (`tnWB`), saves each table/grid as a new sheet. Once you have finalized any formatting or adding more rows/sheets, you can now save the workbook with the `SaveWorkbook()` or `SaveWorkbookAs()` methods. This allows you to add multiple sheets and set the formatting as desired/needed.

The fifth method, `SaveTableToWorkBookEx()`, saves the table directly to the workbook. The xl_* cursors are not used. The only formatting is based on column value type and the font/size setting in the class properties. The first row text is set to bold and can also be frozen. No other formatting is possible during the output process. This method saves directly to the xml files using FWRITE() command and is very fast.

Both methods `SaveTableToWorkBook()` and `SaveGridToWorkBook()` first saves to the xl_* cursors and then these same xl_* cursors has to be queried to now save to the xml files.

Defined Numeric Formatting

Most of the standard numeric formatting is supported by this class; the following #DEFINEs are provided for the supported format codes.

#DEFINE Name	Format Code
CELL_FORMAT_INTEGER	0
CELL_FORMAT_FLOAT	0.00
CELL_FORMAT_COMMA_INTEGER	#,##0
CELL_FORMAT_COMMA_FLOAT	#,##0.00
CELL_FORMAT_CURRENCY_PAREN	$#,##0.00;($#,##0.00)
CELL_FORMAT_CURRENCY_RED_PAREN	$#,##0.00;[Red]($#,##0.00)
CELL_FORMAT_PERCENT_INTEGER	###%
CELL_FORMAT_PERCENT_FLOAT	###.00%
CELL_FORMAT_EXPONENT	0.00E+00
CELL_FORMAT_FRACTION_1	# ?/?
CELL_FORMAT_FRACTION_2	# ??/??
CELL_FORMAT_DATE_MMDDYY	mm-dd-yy
CELL_FORMAT_DATE_DMMMYY	d-mmm-yy
CELL_FORMAT_DATE_DMMM	d-mmm
CELL_FORMAT_DATE_MMMYY	mmm-yy
CELL_FORMAT_TIME_HMMAMPM	h:mm AM/PM
CELL_FORMAT_TIME_HMMSSAMPM	h:mm:ss AM/PM
CELL_FORMAT_TIME_HMM	h:mm
CELL_FORMAT_TIME_HMMSS	h:mm:ss
CELL_FORMAT_DATETIME_MDYYHMM	m/d/yy h:mm
CELL_FORMAT_DATETIME_DDMMMYYYY_TTAM	[$-409]dd/mmm/yyyy\ h:mm\ AM/PM;@
CELL_FORMAT_DATETIME_DDMMMYYYY_TT24	dd/mmm/yyyy\ h:mm;@
CELL_FORMAT_DATETIME_MMMDDYYYY_TTAM	[$-409]mmm\ d\,\ yyyy\ h:mm\ AM/PM;@
CELL_FORMAT_DATETIME_MMMDDYYYY_TT24	[$-409]mmm\ d\,\ yyyy\ h:mm;@
CELL_FORMAT_DATETIME_MDYY_TTAM	m/d/yy\ h:mm\ AM/PM;@
CELL_FORMAT_DATETIME_MDYY_TT24	m/d/yy\ h:mm;@
CELL_FORMAT_COMMA_INTEGER_PAREN	#,##0;(#,##0)
CELL_FORMAT_COMMA_INTEGER_RED_PAREN	#,##0;[Red](#,##0)
CELL_FORMAT_COMMA_FLOAT_PAREN	#,##0.00;(#,##0.00)
CELL_FORMAT_COMMA_FLOAT_RED_PAREN	#,##0.00;[Red](#,##0.00)

#DEFINE Name	Format Code
CELL_FORMAT_TIME_MMSS	mm:ss
CELL_FORMAT_TIME_H_MMSS	[h]:mm:ss
CELL_FORMAT_CURRENCY_RED	$#,##0.00;[Red]$#,##0.00

Custom Defined Numeric Formatting

Additional custom numeric formatting can be defined as needed using the method:

```
this.AddNumericFormat(tcFormatCode)
```

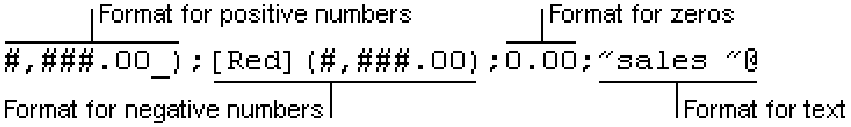
This method will break-down the format into its components in order to support the number of decimals being set for a cell. This allows for the numeric format to be independent of the number of decimals; otherwise, a different numeric format would have to be defined just for a different number of decimals to be shown. However, this method requireds the developer to supply the fully defined format based on all rules for numeric formats.

A newer method is provided that breaks down the format into the various parts and then will construct the numeric format which is hopefully easier to use for creating custom numeric formats. The parameters are:

```
this.AddCustomNumericFormat(tcPosSect, tcNegSect, tcZeroSect,
                            tcTextSect, tlApplyDec)
```

Up to four sections of format codes can be specified. The format codes, separated by semi-colons, define the formats for positive numbers (tcPosSect), negative numbers (tcNegSect), zero values (tcZeroSect), and text (tcTextSect), in that order. If only two sections are specified, the first is used for positive numbers and zeros, and the second is used for negative numbers. If only one section is specified, it is used for all numbers. If a semi-colon is part of the section code it will result in the method considering it an error and will not include the format.

A representation of the numeric format is as follows:


Format for positive numbers
Format for zeros
#,###.00_);[Red](#,###.00);0.00;"sales "@
Format for negative numbers
Format for text

The first section, "tcPosSect - Format for positive numbers", is the format code that applies to the cell when the cell value contains a positive number.

The second section, "tcNegSect - Format for negative numbers", is the format code that applies to the cell when the cell value contains a negative number.

The third section, "tcZeroSect - Format for zeros", is the format code that applies to the cell when the cell value is zero.

The fourth, and last, section, "`tcTextSect` - Format for text", is the format code that applies to the cell when the cell value is text.

The & (ampersand) text operator is used to join, or concatenate, two values.

The following table describes the different symbols that are available for use in custom number formats.

Format Symbol	Description and Result
0	Digit placeholder. [Example: If the value 8.9 is to be displayed as 8.90, use the format #.00]
#	Digit placeholder. This symbol follows the same rules as the 0 symbol. However, the application shall not display extra zeros when the number typed has fewer digits on either side of the decimal than there are # symbols in the format. [Example: If the custom format is #.##, and 8.9 is in the cell, the number 8.9 is displayed]
?	Digit placeholder. This symbol follows the same rules as the 0 symbol. However, the application shall put a space for insignificant zeros on either side of the decimal point so that decimal points are aligned in the column. [Example: The custom format 0.0? aligns the decimal points for the numbers 8.9 and 88.99 in a column]
. (period)	Decimal point.
%	Percentage. If the cell contains a number between 0 and 1, and the custom format 0% is used, the application shall multiply the number by 100 and add the percentage symbol in the cell.
, (comma)	Thousands separator. The application shall separate thousands by commas if the format contains a comma that is enclosed by number signs (#) or by zeros. A comma that follows a placeholder scales the number by one thousand. [Example: If the format is `#.0,,` and the cell value is 12,200,000 then the number 12.2 is displayed]
E- E+ e- e+	Scientific format. The application shall display a number to the right of the "E" symbol that corresponds to the number of places that the decimal point was moved. [Example: If the format is 0.00E+00, and the value 12,200,000 is in the cell, the number 1.22E+07 is displayed. If the number format is #0.0E+0, then the number 12.2E+6 is displayed.]
$-+():space	Displays the symbol. If it is desired to display a character that differs from one of these symbols, precede the character with a backslash (\). Alternatively, enclose the character in quotation marks. [Example: If the number format is (000), and the value 12 is in the cell, the number (012) is displayed]

Format Symbol	Description and Result
/	If this symbol is preceded and followed by a number symbol (0, #, and ?), it is interpreted as the fraction format symbol and will display the number in the format of a fraction. Otherwise, it is interpreted as the forward slash character and is displayed as such.
\	Displays the next character in the format. The application shall not display the backslash. [Example: If the number format is `0\!`, and the value 3 is in the cell, the value `3!` is displayed]
*	Repeats the next character in the format enough times to fill the column to its current width. There shall not be more than one asterisk in one section of the format. If more than one asterisk appears in one section of the format, all but the last asterisk shall be ignored. [Example: if the number format is 0*x, and the value 3 is in the cell, the value 3xxxxxx is displayed. The number of x characters that are displayed in the cell varies based on the width of the column]
_ (underline)	Skips the width of the next character. This is useful for lining up negative and positive values in different cells of the same column. [Example: The number format _(0.0_);(0.0) aligns the numbers 2.3 and -4.5 in the column even though the negative number is enclosed by parentheses]
"text"	Displays whatever text is inside the quotation marks. [Example: The format 0.00 "dollars" displays 1.23 dollars when the value 1.23 is in the cell]
@	Text placeholder. If text is typed in the cell, the text from the cell is placed in the format where the at symbol (@) appears. [Example: If the number format is "Bob "@" Smith" (including quotation marks), and the value "John" is in the cell, the value Bob John Smith is displayed]

Cell Styles

Formatting for a cell that includes font, indentation, borders, fill, etc. in a XLSX file is defined in a style definition internally. This internal style definition is then assigned to individual cells. If one cell is formatted bold and a second cell is formatted non-bold, then there would be two different styles defined. Additionally, there would be two different font definitions defined. Each time a new font definition, border definition, fill definition, etc., is added, a new style has to be defined. Then this style is used to define the formatting for a given cell.

The previous methods for assigning cell formatting took care of when to create a new style definition or when to add to an existing style definition. But this choice of design causes a lot of overhead in the cell formatting assignment process. In order to reduce this overhead, I have added new methods for managing the cell formatting process using the style as the base. This is a similar approach to cell formatting that is used in the Apache Foundation POI Java Classes. The older cell formatting methods will remain in the class but will not be enhanced anymore and should be considered as *depreicated code*.

The first of the style methods is the CreateCellStyle() method. This method creates a base style entry that can be enhanced with the different formatting choices: font, fill, border, etc. A series of methods that begin with AddStyle… are used to assign the different formatting requirements to a style definition. Once a style is defined, it can then be assigned to an individual cell via the SetCellStyle() method or to a series of cells via the SetCellStyleRange() method. Changes to a style definition will automatically be reflected in all cells that reference the style definition.

An example of using style based formatting is as follows (see Demo() method):

```
lnStyle1 = this.CreateFormatStyle(lnWB)        && Create the base style definition
this.AddStyleBorders(lnWB, lnStyle1, 63, BORDER_STYLE_THIN, RGB(16,100,200))
this.AddStyleFont(lnWB, lnStyle1, "Times New Roman", 14, False, False, RGB(0,0,255))
this.SetCellStyleRange(lnWB, lnSh1, 2, 1, 2, 9, lnStyle1)  && Assign formatting style to cells
```

SpreadSheet Headers/Footers

This class supports writing headers and footers for individual spreadsheets which includes different first page, different odd/even pages, and same all pages. The placement of the text can be left section, center section, and/or right section. Font support is also provided. The following method must be first called to set the header/footer properties before assigning any text:

```
this.SetHeaderFooterSetup(tnWB, tnSheet, tlAlignMargin, tlDiffFirstPg,
                          tlDiffOddEven, tlScaleWDoc)
```

After setting the header/footer properties, the following method is called to set the text (see the method below for more details):

```
this.SetHeaderFooterText(tnWB, tnSheet, tnPage, tnSection, tcText, tcFontName,
                         tnFontSize, tnFontEffect, tnFontColor)
```

The default font handling is for the entire section text; there is not direct support for different formatting within a section text. However, this can be encoded within the section text by the developer. Special symbol inclusion (such as page number, number of pages, etc.) in the text is also not directly supported; but, these can be added by the developer into the header text as well. An example of placing the text into a header or footer as:

Page # of ## *Where # is the current page number; ## is total page count*

Can be done with the following text assigned to a header/footer section:

```
"Page &amp;P of &amp;N"
```

Where `&amp;P` is the code for current page, and `&amp;N` is the code for page count.

Additional embedded formatting commands are available. These are:

Embedded Code	Explanation / Meaning
`&amp;P`	Code for "current page #"
`&amp;N`	Code for "total pages"
`&amp;font size`	Code for "text font size", where `font size` is a font size in points.
`&amp;K`	Code for "text font color" RGB Color is specified as RRGGBB which is appended to end of code; example red is: `&amp;KFF0000`
`&amp;S`	Code for "text strikethrough" on / off
`&amp;X`	Code for "text super script" on / off
`&amp;Y`	Code for "text subscript" on / off
`&amp;D`	Code for "date"
`&amp;T`	Code for "time"
`&amp;U`	Code for "text single underline"
`&amp;E`	Code for "double underline"
`&amp;Z`	Code for "this workbook's file path"
`&amp;F`	Code for "this workbook's file name"
`&amp;A`	Code for "sheet tab name"
`&amp;+`	Code for add to page #.
`&amp;-`	Code for subtract from page #.
`&amp;"font name,style"`	Code for "text font name" and "text font style", where font name and font style are strings specifying the name and style of the font, separated by a comma. When a hyphen appears in font name, it means "none specified".

Embedded Code	Explanation / Meaning
`&amp;"-,Bold"`	Code for "bold font style"
`&amp;B`	Also means "bold font style".
`&amp;"-,Regular"`	Code for "regular font style"
`&amp;"-,Italic"`	Code for "italic font style"
`&amp;I`	Also means "italic font style"
`&amp;"-,Bold Italic"`	Code for "bold italic font style"

Font formatting will apply to all text following the embedded command until a new embedded font formatting command is encountered.

Properties

AutoTrimSheetName

Description	Indicates whether to auto-trim the sheet name if too long
Default Value	True

CodePage

Description	CodePage to use for the Strings cursor
Default Value	VFP default value

CompanyName

Description	Company name in workbook properties
Default Value	VFPxWorkbookXLSX

CreatorName

Description	Creator in workbook properties
Default Value	VFPxWorkbookXLSX

DeclareWinAPI

Description	Boolean to declare the needed Win32 API functions called in Init()
Default Value	False

Debug

Description	Sets debugging mode
Default Value	False

DefaultFont

Description	Default font name
Default Value	Calibri

DefaultFontSize (new with Release 25)

Description Default font size

Default Value 11

DefaultSheetName

Description Default sheet name

Default Value Sheet

ErrorLevelId

Description Error level Id that has occurred (see OnErrorMessage() event for id values assigned)

Default Value 0 [no errors]

ExcelXlsxRelease

Description Release version of class

Default Value Sheet

SaveCurrencyAsNumeric

Description Indicates whether to save a currency value as a currency value or as a numeric value [Boolean]

Default Value False

Subject

Description Subject in workbook properties

Default Value <none>

TrueFalseValue

Description The value to display in the cell for a boolean field type; pipe delimited list of the true value followed by the false value

Default Value Yes|No

Title

Description	Title in workbook properties
Default Value	<none>

UserName

Description	Name of person stored in XLSX document as last edit
Default Value	VFPxWorkbookXLSX

Events

OnDestroy

Description: Called by Destroy Event; for placing user code

Parameters:

None

OnInit

Description: Called by Init Event; for placing user code

Parameters:

None

OnShowErrorMessage

Description: Called for displaying a user message when an error occurs. Use BINDEVENTS to bind to this event.

Parameters:

tnErrorId Error Id.

The following errors occur during opening of a workbook

1 OpenXlsxWorkbook() - must include file name to open
2 OpenXlsxWorkbook() - error assigned by TRY-CATCH
3 OpenXlsxWorkbook() - missing workbook.xml
4 OpenXlsxWorkbook() - missing workbook.xml.rels
5 OpenXlsxWorkbook() - missing styles.xml
6 OpenXlsxWorkbook() - missing sharedStrings.xml
7 OpenXlsxWorkbook() - error during shared string loading
8 OpenXlsxWorkbook() - missing sheet or invalid sheet <id>
9 OpenXlsxWorkbook() - error reading data; error assigned by TRY-CATCH

The following errors occur during saving of a workbook

10 CreateExcelFile - unable to delete existing file; error assigned by TRY-CATCH
11 CreateExcelFile - Failed to create Zip file
12 CreateExcelFile - Failed to add contents to Zip file
13 CreateExcelFile - Rename failed (changing from zip to xlsx extension)
14 WriteSheetXMLs - Failed to create a sheet; error assigned by TRY-CATCH
15 WriteStringsXML – Failed to create sharedstrings.xml; error assigned by TRY-CATCH
16 WriteRelationshipsXML – Unable to create workbook.xml.rels; error assigned by TRY-CATCH

17 WriteStylesXML - Unable to create styles.xml; error assigned by TRY-CATCH

18 WriteSupportXMLs - Unable to create workbook supporting XMLs; error assigned by TRY-CATCH

The following are general errors

99 Occurs when failure to open the workbook via ShellExecute API command

tcErrMessage — Error message text

OnShowStatusMessage

Description: Called for displaying a user message during the opening of an existing workbook (xlsx) file. Use BINDEVENTS to bind to this event.

Parameters:

tnMode — Mode of the current processing; 1 indicates opening an xlsx file and 2 indicates saving an xlsx file

tnStage — Stage of the process

tnTotStages — Total number of stages to process (passed only on the first call)

Comments: The following is a listing of the values

```
When nMode = 1
nStage = 0; start of open
nStage = 1; reading shared strings XML
nStage = 2; reading styles XML
nStage = 3; reading relationships XML
nStage = 4; reading sheets XML
nStage = 5; reading named ranges
nStage = 6; reading external references
nStage = -1; end of open

When nMode = 2
nStage = 0; start of save
nStage = 1; indicates saving supporting XMLs
nStage = 2; indicates saving strings XML
nStage = 3; indicates saving styles XML
nStage = 4; indicates saving workbook
nStage = 5; indicates saving relationship XML
nStage = 6+; indicates saving sheets
nStage = -1; end of close
```

Methods – Managing Workbooks

CreateWorkbook

Description: Creates a new workbook

Parameters:

tcName Full path and file name of Excel Xlsx Workbook to create

Return Value:

Id of Sheet
0 if failure

DeleteAllWorkbooks

Description: Deletes all workbook Ids

Parameters:

None

Return Value:

None

DeleteWorkbook

Description: Deletes the workbook Id

Parameters:

tnWB Id to workbook

Return Value:

True on success
False on failure

GetNumberOfSheets

Description: Returns the number of defined sheets for the given workbook id.

Parameters:

tnWB	Id to workbook

Return Value:

Number of sheets

GetWorkbook

Description: Gets the workbook Id

Parameters:

tcName	file name of Excel Xlsx Workbook to return

Return Value:

Id of workbook

Zero if failure

GetWorkbookProtection

Description: Gets the workbook protection settings

Parameters:

tnWB	Id to workbook

Return Value:

Boolean – True if protected; False if not. .NULL. is returned if not a valid workbook Id

GetWorkbookFileName

Description: Gets the workbook file name

Parameters:

tnWB	workbook number returned by CreateWorkbook()

Return Value:

File name of the workbook

Empty string if failure

OpenCreatedXlsxFile

Description: Opens the selected workbook in the default program via ShellExecute Win API

Parameters:

txWB	Integer: workbook number returned by CreateWorkbook() String: workbook file name (full path)

Return Value:

False	Failed to open or find workbook
True	Default

OpenXlsxWorkbook

Description: Opens the passed XLSX workbook and loads the internal cursors with the content

Parameters:

tcFileName	File name with full path of the XLSX file to open
tlForceTextFormat	If True, then cell values are forced to Text format [optional]; defaulted to False
tlReadGraphicData	If True, then any graphical data will be loaded [optional]; defaulted to True [New parameter]

Return Value:

Id of workbook; Zero if failure

OpenXlsxWorkbookSheet

Description: Opens a selected worksheet in a XLXS workbook; always sets the opened sheet as sheet1

Parameters:

tcFileName	File name with full path of the XLSX file to open
txSheet	Can be either the sheet name or the sheet Id to open
tlForceTextFormat	If True, then cell values are forced to Text format [optional]; defaulted to False
tlReadGraphicData	If True, then any graphical data will be loaded [optional]; defaulted to True

Return Value:

Id of workbook; Zero if failure

SaveWorkbook

Description: Saves the selected workbook to xlsx file format based on the name set at creation of the workbook

Parameters:

tnWB	Id to workbook

Return Value:

True on success

False on failure

SaveWorkbookAs

Description: Saves the selected workbook to xlsx file format with the supplied file name; resets the workbook file name for future saves

Parameters:

tnWB	Id to workbook
tcWBName	File path and file name to save-as

Return Value:

True on success

False on failure

SetWorkbookProtection

Description: Sets the workbook protection

Parameters:

tnWB	Id to workbook
tlLocked	Boolean, True to lock and False to unlock
tcPassword	The encoded password for the workbook (you have to encode it); defaulted value of '*vfpxworkbookxlsx*'
tcAlgorithm	See the 'ECMA Office Open XML Part 1 - Fundamentals And Markup Language Reference' for types available

Return Value:

Boolean – True if protection set; False if workbook not found

Methods – Managing Sheets

AddAutoFilter

Description: Adds a filter to the column range

Parameters:

tnWB	Id to workbook to add sheet to
tnSheet	Id to sheet in workbook
tnBegCol	Beginning column index for filter
tnEndCol	Ending column index for filter

Return Value:

True on success

False on failure

AddColumnFilter

Description: Sets the specific filter for a column

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook
tnCol	Column to assign filter to
tcOperator	Numeric operator to apply to the filter

txFilterValue	Value for the filter operator; can be any value type (stored as a character); values provided by `#DEFINE`
	`FILTER_OP_EQUAL`
	`FILTER_OP_GREATERTHAN`
	`FILTER_OP_GREATOREQUAL`
	`FILTER_OP_LESSTHAN`
	`FILTER_OP_LESSOREQUAL`
	`FILTER_OP_NOT_EQUAL`

tlAndOperator	Indicates if the column filter for multiple filter conditions is an OR or an AND operation

Return Value:

True on success

False on failure

AddGroupByColumn

Description: Adds a column group level to the selection

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook
tnBegCol	Beginning column index for group
tnEndCol	Ending column index for group

Return Value:

True on success; False on failure

AddGroupByRow

Description: Adds a row group level to the selection

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook
tnBegRow	Beginning row index for group
tnEndRow	Ending row index for group

Return Value:

True on success; False on failure

AddHyperLinkFile

Description: Adds a new hyperlink to the sheet that links to an external file (not contained in the workbook)

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook
tnBegRow	Beginning row index to insert hyperlink
tnBegCol	Beginning column index to insert hyperlink
tnEndRow	Ending row index to insert hyperlink
tnEndCol	Ending column index to insert hyperlink
tcTarget	External file name with full path

Return Value:

True on success; False on failure

AddHyperLinkSheet

Description: Adds a new hyperlink to the sheet that links to another existing sheet in the workbook

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook
tnBegRow	Beginning row index to insert hyperlink
tnBegCol	Beginning column index to insert hyperlink
tnEndRow	Ending row index to insert hyperlink
tnEndCol	Ending column index to insert hyperlink
tnTgtSheet	Sheet Id of the target sheet to hyperlink to
tnTgtBegRow	Beginning row index to hyperlink to
tnTgtBegCol	Beginning column index to hyperlink to
tnTgtEndRow	Ending row index to hyperlink to
tnTgtEndCol	Ending column index to hyperlink to
tcDisplay	Text to display in the hyperlink cell(s); defaults to the current cell value

Return Value:

True on success; False on failure

AddImage

Description: Adds a new image to the sheet

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook
tcImageFile	File name of the image with full path
tcAnchorType	Type of anchoring to be used; values provided by #DEFINE

```
IMAGE_ANCHOR_TYPE_ABS   && Positioned by absolute
IMAGE_ANCHOR_TYPE_ONE   && Positioned by one cell
IMAGE_ANCHOR_TYPE_TWO   && Positioned by two cells
```

tcImgMove	Positioning setting for image; values provided by #DEFINE

```
IMAGE_ANCHOR_MOVE_ABS
IMAGE_ANCHOR_MOVE_ONE
IMAGE_ANCHOR_MOVE_TWO
```

tnBegCol	Beginning column index
tnBegColOff	Offset from beginning column; value given in centimeters
tnBegRow	Beginning row index
tnBegRowOff	Offset from beginning row; value given in centimeters

tnEndCol Ending column index

tnEndColOff Offset from ending column; value given in centimeters

tnEndRow Ending column index

tnEndRowOff Offset from ending column; value given in centimeters

Return Value:

Id of image; 0 if failed

AddSheet (parameter change with Release 16)

Description: Adds a new sheet to the workbook

Parameters:

tnWB Id to workbook to add sheet to

tcSheetName Name of the sheet to be added; limited to 30 characters

tnState Visibility of sheet [optional parameter, defaults to Visible]; select value from #DEFINEs

```
VISIBLE_SHEET_STATE
HIDDEN_SHEET_STATE
VERYHIDDEN_SHEET_STATE
```

Return Value:

Id of Sheet; 0 if failure

ClearAutoFilter

Description: Removes auto-filter from sheet

Parameters:

tnWB Id to workbook to add sheet to

tnSheet Id to sheet in workbook

Return Value:

True on success

False on failure

DeleteImage

Description: Deletes an image from the worksheet

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook
tnRelId	Image relationship Id (value returned by AddImage method)

Return Value:

True on success; False on failure

DeleteHyperLink

Description: Deletes a hyperlink from the worksheet

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook
tnBegRow	Beginning row index to hyperlink
tnBegCol	Beginning column index to hyperlink
tnEndRow	Ending row index to hyperlink
tnEndCol	Ending column index to hyperlink

Return Value:

True on success; False on failure

DeleteSheet

Description: Deletes the workbook sheet

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook

Return Value:

True on success

False on failure

GetColumnHidden

Description: Returns the hidden setting of the selected column

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnColumn	Column index to get the hidden setting

Return Value:

Hidden setting; True if hidden, False if not hidden

NULL on failure or sheet does not exist

GetColumnWidth

Description: Returns the width of the selected column

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnColumn	Column index to reset width

Return Value:

Width of column; -1 is returned if a column width is not explicitly set

NULL on failure or sheet does not exist

GetDisplayGridLines

Description: Gets the display setting for showing/hiding grid lines in the sheet

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook

Return Value:

Boolean – True if displayed, False if not displayed

GetLastColumnInRow

Description: Returns the max column number for a given row in a sheet

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnRow	Row number

Return Value:

Integer value of maximum column number in row; zero if none.

GetLastRowNumber

Description: Returns the last row number in the sheet

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook

Return Value:

Integer value of last row number; zero if none.

GetMaxColumnNumber

Description: Returns the max column number for a sheet

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook

Return Value:

Integer value of maximum column number in sheet across all rows; zero if none.

GetRowMaxColumn

Description: Returns the max column number for a given row in a given sheet

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnCellRow	Row number to return

Return Value:

Integer value of maximum column number in row; zero if none.

GetSheetIndex

Description: Gets the internal sheet index from the sheet name for a given workbook

Parameters:

tnWB	Id to workbook
tcShName	Name of sheet

Return Value:

Id to sheet in workbook

GetSheetName

Description: Returns the sheet name

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook

Return Value:

Name of sheet or empty string if not found

GetSheetProtection

Description: Gets the sheet protection for a workbook

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook

Return Value:

Sheet Protecton object:

loProtection.Locked
loProtection.AlgorithmName
loProtection.Password
loProtection.AutoFilter
loProtection.DeleteColumns
loProtection.DeleteRows
loProtection.FormatCells
loProtection.FormatColumns
loProtection.FormatRows
loProtection.InsertColumns
loProtection.InsertRows
loProtection.InsertHyperlinks
loProtection.PivotTables
loProtection.SelectLockedCells
loProtection.SelectUnlockedCells
loProtection.Sort
loProtection.Objects
loProtection.Scenarios

GetWorkbookSheets

Description: Gets the sheet information for a workbook

Parameters:

tnWB	Id to workbook

Return Value:

Sheet list object:

loSheets.Count	Count of sheets
loSheets.List[n, 1]	Sheet Id
loSheets.List[n, 2]	Sheet Name

InsertCell

Description: Inserts a new cell into the sheet

Parameters:

tnWB	Id to workbook
tnSheet	Sheet Id
tnCellRow	Numeric cell value for row
tnCellCol	Numeric cell value for column
tnShift	Shift direction for the cell insertion; select value from #DEFINEs
	`INSERT_LEFT`
	`INSERT_RIGHT`
	`INSERT_BEFORE`
	`INSERT_AFTER`

Return Value:

True on success

False on failure

InsertColumn

Description: Inserts a new column into the sheet

Parameters:

tnWB	Id to workbook
tnSheet	Sheet Id
tnCellCol	Numeric cell value for column
tnShift	Shift direction for the cell insertion; select value from #DEFINEs
	`INSERT_LEFT`
	`INSERT_RIGHT`

Return Value:

True on success

False on failure

InsertRow

Description: Inserts a new row into the sheet

Parameters:

tnWB	Id to workbook
tnSheet	Sheet Id
tnCellRow	Numeric cell value for row
tnShift	Shift direction for the cell insertion; select value from #DEFINEs INSERT_BEFORE INSERT_AFTER

Return Value:

True on success
False on failure

RenameSheet

Description: Renames the selected sheet in the workbook

Parameters:

tnWB	Id to workbook
txSheet	Sheet to remove; can be either the sheet Id or the sheet name
tcSheetName	New name for the sheet; limited to 30 characters

Return Value:

True on success
False on failure

SetDisplayGridLines

Description: Sets the display setting for showing/hiding grid lines in the sheet

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook
tlGridLines	True if displayed, False if not displayed

Return Value:

True on success
False on failure

SetSheetGroupSettings

Description: Sets the row and column summary settings (roll-up or roll-down)

Parameters:

tnWB	Id to workbook
tnSheet	Sheet Id
tlSummaryBelow	Boolean – True for summary below, False for above
tlSummaryRight	Boolean – True for summay right, False for left

Return Value:

True on success

False on failure

SetSheetProtection

Description: Sets the sheet protection settings

Parameters:

tnWB	Id to workbook
tnSheet	Sheet Id
toProtection	Object with protection settings assigned (see GetSheetProtection method for object properties).

Return Value:

True on success

False on failure

SetSheetVisibility

Description: Set the selected sheet visiblity in the workbook

Parameters:

tnWB	Id to workbook
txSheet	Sheet to remove; can be either the sheet Id or the sheet name

tnState	Visibility of sheet; select value from #DEFINEs
	`SHEET_STATE_VISIBLE`
	`SHEET_STATE_HIDDEN`
	`SHEET_STATE_VERYHIDDEN`

Return Value:

True on success; False on failure

SetTabColor

Description: Sets the selected sheet tab color in the workbook

Parameters:

tnWB	Id to workbook
tnSheet	Sheet Id
tnRBGColor	The RBG color value as returned by RGB() function

Return Value:

True on success; False on failure

UnGroupByColumn

Description: Removes a column group level to the selection

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook
tnBegCol	Beginning column index for group
tnEndCol	Ending column index for group

Return Value:

True on success
False on failure

UnGroupByRow

Description: Removes a row group level to the selection

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook
tnBegRow	Beginning row index for group
tnEndRow	Ending row index for group

Return Value:

True on success
False on failure

Methods – Assigning Cell Values

ClearCellValue

Description: Clears the value from the selected cell

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Numeric cell value for row
tnCellCol	Numeric cell value for column

Return Value:

True on success

False on failure

SetCellFormula

Description: Sets the cell formula

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)
tcCellFormula	Formula to add; you must format the formula with cell references and preceeded with an equals sign; i.e., =SUM(A1:A10)

Return Value:

True on success

False on failure

SetCellValue

Description: Sets the cell value. The data type is set by the data type of the value to be set (determined via VARTYPE() function)

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)

txCellValue	Value to set; supported data types include (#DEFINEs):
	`DATA_TYPE_CHAR`
	`DATA_TYPE_DATE`
	`DATA_TYPE_DATETIME`
	`DATA_TYPE_CURRENCY`
	`DATA_TYPE_FLOAT`
	`DATA_TYPE_INT`
	`DATA_TYPE_GENERAL (this is set to an empty string)`

Return Value:

True on success

False on failure

Methods – Returning Cell Values

GetCellDataType

Description: Returns the cell data type; this is based on the character expression or the cell format.

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Numeric cell value for row
tnCellCol	Numeric cell value for column

Return Value:

Data type for the cell; see SetCellValue() method for a list of data type #DEFINEs.

GetCellFormula

Description: Returns the cell formula expression

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Numeric cell value for row
tnCellCol	Numeric cell value for column

Return Value:

Formula expression for the cell

GetCellValue

Description: Returns the value from the selected cell

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Numeric cell value for row
tnCellCol	Numeric cell value for column

Return Value:

Cell value set to the data type of the cell

GetSheetRowValues

Description: Returns the cell values for the given row

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Numeric cell value for row

Return Value:

Return object:

loRow.Count	Number of columns returned in row
loRow.Values[nCol, 1]	Cell value set to data type of the cell
loRow.Values[nCol, 2]	Cell data type

A NULL value for a column indicates a value is not set. If a failure occurs (sheet or column does not exist, then a NULL is returned).

IsCellFormula

Description: Determines if the cell contains a formula

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row number
tnCellCol	Cell column number

Return Value:

True if the cell contains a formula; otherwise false.

Methods – Style Formatting

AddStyleBorders

Description: Adds to the style definition cell border formatting

Parameters:

tnWB	Id to workbook
tnCellXfsId	Id to the format style

tnBorders	Cell Border to draw; this is a addition of the appropriate border side to set; to set all sides:
	`BORDER_LEFT +`
	`BORDER_RIGHT +`
	`BORDER_TOP +`
	`BORDER_BOTTOM +`
	`BORDER_DIAGDOWN +`
	`BORDER_DIAGUP`

tcBorderStyle	Style of border to draw; the following styles are available:
	`BORDER_STYLE_THIN`
	`BORDER_STYLE_HAIR`
	`BORDER_STYLE_DOTTED`
	`BORDER_STYLE_DASHDOTDOT`
	`BORDER_STYLE_DASHDOT`
	`BORDER_STYLE_DASHED`
	`BORDER_STYLE_THIN`
	`BORDER_STYLE_MEDIUMDASHDOTDOT`
	`BORDER_STYLE_SLANTDASHDOT`
	`BORDER_STYLE_MEDIUMDASHDOT`
	`BORDER_STYLE_MEDIUMDASHED`
	`BORDER_STYLE_MEDIUM`
	`BORDER_STYLE_THICK`
	`BORDER_STYLE_DOUBLE`

tnBorderColor	The color to draw the border in RGB() value

Return Value:

True on success; false on failure to assign

AddStyleFill

Description: Adds to the style definition cell fill formatting. Cell fill patterns operate with two colors: a background color and a foreground color. These combine together to make a patterned cell fill (the foreground color sets the pattern color). *The foreground color of the cell does not affect the text foreground color; text foreground color is set in the AddStyleFont() method.*

Parameters:

tnWB	Id to workbook
tnCellXfsId	Id to the format style
tnFColor	Fill foreground color; RGB(N,N,N)
tnBColor	Fill background color; RGB(N,N,N)
tcPatternType	Fill pattern type; based on #DEFINEs
	`FILL_STYLE_NONE`
	`FILL_STYLE_SOLID`
	`FILL_STYLE_GRAY125`

Return Value:

True on success; false on failure to assign

AddStyleFont

Description: Adds to the style definition cell font formatting

Parameters:

tnWB	Id to workbook
tnCellXfsId	Id to the format style
tcFName	Font name
tnFSize	Font size
tlBold	Boolean to indicate bold font
tlItalic	Boolean to indicate italic font
tnFColor	Font foreground color; RGB(N,N,N)
tcULine	Boolean to indicate underline
tlStrikThr	Boolean to indicate strikethrough
tcVPos	Verical position of text (from #DEFINEs)
	`FONT_VERTICAL_BASELINE`
	`FONT_VERTICAL_SUBSCRIPT`
	`FONT_VERTICAL_SUPERSCRIPT`

Return Value:

True on success; false on failure to assign

AddStyleHorizAlignment

Description: Adds to the style definition cell horizontal alignment formatting

Parameters:

tnWB	Id to workbook
tnCellXfsId	Id to the format style
tcHorizAlign	Assigned by the following #DEFINEs

```
CELL_HORIZ_ALIGN_LEFT
CELL_HORIZ_ALIGN_RIGHT
CELL_HORIZ_ALIGN_CENTER
```

Return Value:

True on success; false on failure to assign

AddStyleIndent

Description: Adds to the style definition cell indent formatting

Parameters:

tnWB	Id to workbook
tnCellXfsId	Id to the format style
tnIndent	Amount of indent to apply

Return Value:

True on success; false on failure to assign

AddStyleNumericFormat

Description: Adds to the style definition cell numeric formatting

Parameters:

tnWB	Id to workbook
tnCellXfsId	Id to the format style
tnNumFmtId	Value of numeric format (from #DEFINEs)

```
CELL_FORMAT_INTEGER
CELL_FORMAT_FLOAT
CELL_FORMAT_COMMA_INTEGER
CELL_FORMAT_COMMA_FLOAT
CELL_FORMAT_CURRENCY_PAREN
CELL_FORMAT_CURRENCY_RED_PAREN
CELL_FORMAT_CURR_EURO_RED
CELL_FORMAT_CURR_POUNDS_RED
CELL_FORMAT_PERCENT_INTEGER
```

```
CELL_FORMAT_PERCENT_FLOAT
CELL_FORMAT_EXPONENT
CELL_FORMAT_FRACTION_1
CELL_FORMAT_FRACTION_2
CELL_FORMAT_DATE_MMDDYY
CELL_FORMAT_DATE_DMMMYY
CELL_FORMAT_DATE_DMMM
CELL_FORMAT_DATE_MMMYY
CELL_FORMAT_TIME_HMMAMPM
CELL_FORMAT_TIME_HMMSSAMPM
CELL_FORMAT_TIME_HMM
CELL_FORMAT_TIME_HMMSS
CELL_FORMAT_DATETIME_MDYYHMM
CELL_FORMAT_DATETIME_DDMMMYYYY_TTAM
CELL_FORMAT_DATETIME_DDMMMYYYY_TT24
CELL_FORMAT_DATETIME_MMMDDYYYY_TTAM
CELL_FORMAT_DATETIME_MMMDDYYYY_TT24
CELL_FORMAT_DATETIME_MDYY_TTAM
CELL_FORMAT_DATETIME_MDYY_TT24
CELL_FORMAT_COMMA_INTEGER_PAREN
CELL_FORMAT_COMMA_INTEGER_RED_PAREN
CELL_FORMAT_COMMA_FLOAT_PAREN
CELL_FORMAT_COMMA_FLOAT_RED_PAREN
CELL_FORMAT_TEXT
CELL_FORMAT_TIME_MMSS
CELL_FORMAT_TIME_H_MMSS
CELL_FORMAT_CURRENCY_RED
```

Return Value:

True on success; false on failure to assign

AddStyleProtection

Description: Sets the style’s protection values (locked and hidden)

Parameters:

tnWB	Id to workbook
tnCellXfsId	Id to the format style
tnLocked	Boolean – True to lock, False for unlock
tnHidden	Boolean – True to Hide, False for Visible

Return Value:

True on success; false on failure to assign

AddStyleTextRotation

Description: Adds to the style definition cell text rotation formatting

Parameters:

tnWB	Id to workbook
tnCellXfsId	Id to the format style
tnRotation	Rotation angle to set the text (value between -90 and 90 degrees)

Return Value:

True on success; false on failure to assign

AddStyleVertAlignment

Description: Adds to the style definition cell vertical alignment formatting

Parameters:

tnWB	Id to workbook
tnCellXfsId	Id to the format style
tcVertAlign	Assigned by the following #DEFINEs CELL_VERT_ALIGN_TOP CELL_VERT_ALIGN_BOTTOM CELL_VERT_ALIGN_CENTER

Return Value:

True on success; false on failure to assign

AddStyleWordWrap

Description: Adds to the style definition cell word wrap formatting

Parameters:

tnWB	Id to workbook
tnCellXfsId	Id to the format style
tlWordWrap	True - set wordwrapping on; False - set wordwrapping off

Return Value:

True on success; false on failure to assign

CreateFormatStyle

Description: Creates a new formatting style definition to be applied to cells

Parameters:

tnWB	Id to workbook

Return Value:

Id value of new style

CopyStyle

Description: Copies the style to a new style Id

Parameters:

tnWB	Id to workbook
tnCellXfsId	Id to the format style

Return Value:

Id value of new copied style; -1 if passed style Id not valid

GetCellStyle

Description: Returns the assigned cell style Id value

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)

Return Value:

Id value of new style

GetStyleFormatId

Description: Gets the format style Id for the given numeric format, font format, and fill format. Will dynamically create a new style if it does not exist.

Parameters:

tnWB	Id to workbook
tnNumFmtId	Numeric format Id
tnFillColor	Cell fill RGB() color value
tcFontName	Font name
tnFontSize	Font size
tlFontBold	Font bold setting (true / false)
tlFontItalic	Font italic setting (true / false)
tnFontColor	Font foreground RGB() color value
tcFontULine	Font underline setting (true / false)
tlFontStrikThr	Font strike-thru setting (true / false)
tcFontVPos	Font vertical positioning; set by defines: `FONT_VERTICAL_BASELINE`, `FONT_VERTICAL_SUBSCRIPT`, `FONT_VERTICAL_SUPERSCRIPT`

Return Value:

Id value of style

SetCellStyle

Description: Sets the cell style Id to a selected cell

Parameters:

tnWB	Id to workbook

Return Value:

Id value of new style

SetCellStyleRange

Description: Sets the cell style Id to a selected cell range of rows/columns

Parameters:

tnWB	Id to workbook

Return Value:

Id value of new style

Methods – Returning Cell Formatting

GetCellAlignment

Description: Returns the cell alignment

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)

Return Value:

Return object:

loReturn.HorzAlign	Horizontal alignment value
loReturn.VertAlign	Vertical alignment value

See method SetCellAlignment() for the #DEFINE values

GetCellBorders

Description: Returns the cell border info

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)

Return Value:

Return object:

loBdrInfo.LeftStyle	Left border style
loBdrInfo.LeftColor	Left border color (integer)
loBdrInfo.Index	Left border color index (integer) [changed]
loBdrInfo.Tint	Left border color tint (integer) [changed]
loBdrInfo.Theme	Left border color theme (integer) [changed]
loBdrInfo.RightStyle	Right border style
loBdrInfo.RightColor	Right border color (integer)
loBdrInfo.RightIndex	Right border color index (integer) [changed]
loBdrInfo.RightTint	Right border color tint (integer) [changed]
loBdrInfo.RightTheme	Right border color theme (integer) [changed]

loBdrInfo.TopStyle	Top border style
loBdrInfo.TopColor	Top border color (integer)
loBdrInfo.TopIndex	Top border color index (integer) [changed]
loBdrInfo.TopTint	Top border color tint (integer) [changed]
loBdrInfo.TopTheme	Top border color theme (integer) [changed]
loBdrInfo.BotStyle	Bottom border style
loBdrInfo.BotColor	Bottom border color (integer)
loBdrInfo.BotIndex	Bottom border color index (integer) [changed]
loBdrInfo.BotTint	Bottom border color tint (integer) [changed]
loBdrInfo.BotTheme	Bottom border color theme (integer) [changed]
loBdrInfo.DiagStyle	Diagonal style
loBdrInfo.DiagColor	Diagonal color (integer)
loBdrInfo.DiagIndex	Diagonal border color index (integer) [changed]
loBdrInfo.DiagTint	Diagonal border color tint (integer) [changed]
loBdrInfo.DiagTheme	Diagonal border color theme (integer) [changed]
loBdrInfo.DiagDn	Integer value for down setting [changed]
loBdrInfo.DiagUp	Integer value for up setting [changed]

See method SetCelBorder() for the #DEFINE values

GetCellFill

Description: Returns the fill info for the cell

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)

Return Value:

loFillInfo.FgColor	Fill foreground color (integer)
loFillInfo.BgColor	Fill background color (integer)
loFillInfo.PatType	Fill pattern type
loFillInfo.Theme	Fill color theme (integer)
loFillInfo.Tint	Fill color tint (integer)
loFillInfo.FgIndexed	Fill foreground color index value (integer)
loFillInfo.BgIndexed	Fill background color index value (integer)

NULL if cell fill is not defined.

GetCellFont

Description: Returns the cell font settings

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)

Return Value:

Return object:

loFontInfo.FontName	Font name
loFontInfo.FontSize	Font size (integer)
loFontInfo.FontBold	Boolean; True bold is set, False bold is not set
loFontInfo.FontItalic	Boolean; True italic is set, False italic is not set
loFontInfo.ForeColor	Font forecolor (integer)
loFontInfo.FontUnderline	Boolean; True underline is set, False underline is not set
loFontInfo.FontStrikeThr	Boolean; True strike-through is set, False strike-through is not set
loFontInfo.FontVerticalPos	Verical position of text (set SetCellFont() method for #DEFINE values)

NULL if cell is not defined.

GetCellIndent

Description: Returns the cell indentation

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)

Return Value:

Indentation amount; returns -1 if cell does not exist

GetCellTextRotation

Description: Returns the cell text rotation

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)

Return Value:

Text rotation amount (value between -90 and 90 degrees); returns 99 if incorrect parameters are sent.

GetCellWordWrap

Description: Returns the cell word wrap setting

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)

Return Value:

Boolean value; True wordwrap is set, False wordwrap is not set.

Methods – Assigning Cell Formatting

AddIndexColor

Description: Adds a new indexed color definition to the workbook

Parameters:

tnWB	Id to workbook
tnRGBColor	RGB() color value to add

Return Value:

Index value assigned to color

AddMruColor

Description: Adds a custom defined MRU color to the workbook

Parameters:

tnWB	Id to workbook
tnRGBColor	RGB() color value to add

Return Value:

MRU index value assigned to color

CellFormatPainter

Description: Copies the selected cell format to the specified range of cells

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnSrcRow	Row of cell containing the format that is to be copied
tnSrcCol	Column of cell containing the format that is to be copied
tnBegRow	Row to begin the cell format copy to
tnBegCol	Column to begin the cell format copy to
tnEndRow	Row to end the cell format copy to
tnEndCol	Column to end the cell format copy to

Return Value:

True on success

False on failure

MergeCells

Description: Provides for merging cells into a single cell

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnBegRow	Row to begin the cell merge
tnBegCol	Column to begin the cell merge
tnEndRow	Row to end the cell merge
tnEndCol	Column to end the cell merge

Return Value:

True on success

False on failure

SetCellAlignment (deprecated with Release 18)

Description: Sets the cell alignment (vertical and horizontal)

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)
tcHorizAlign	Horizontal alignment (from #DEFINEs) CELL_HORIZ_ALIGN_LEFT CELL_HORIZ_ALIGN_RIGHT CELL_HORIZ_ALIGN_CENTER
tcVertAlign	Vertical alignment (from #DEFINEs) CELL_VERT_ALIGN_TOP CELL_VERT_ALIGN_BOTTOM CELL_VERT_ALIGN_CENTER

Return Value:

True on success

False on failure

SetCellBorder (deprecated with Release 18)

Description: Sets the cell border; each border is drawed with the same style and color

Parameters:

tnWB Id to workbook

tnSheet Id to sheet in workbook

tnCellRow Cell row (integer)

tnCellCol Cell column (integer)

tnBorders Cell Border to draw; this is a addition of the appropriate border side to set; to set all sides:

```
BORDER_LEFT +
BORDER_RIGHT +
BORDER_TOP +
BORDER_BOTTOM +
BORDER_DIAGDOWN +
BORDER_DIAGUP
```

tcBorderStyle Style of border to draw; the following styles are available:

```
BORDER_STYLE_THIN
BORDER_STYLE_HAIR
BORDER_STYLE_DOTTED
BORDER_STYLE_DASHDOTDOT
BORDER_STYLE_DASHDOT
BORDER_STYLE_DASHED
BORDER_STYLE_THIN
BORDER_STYLE_MEDIUMDASHDOTDOT
BORDER_STYLE_SLANTDASHDOT
BORDER_STYLE_MEDIUMDASHDOT
BORDER_STYLE_MEDIUMDASHED
BORDER_STYLE_MEDIUM
BORDER_STYLE_THICK
BORDER_STYLE_DOUBLE
```

tnBorderColor The color to draw the border in RGB() value

Return Value:

True on success

False on failure

SetCellBorderEx (deprecated with Release 18)

Description: Sets the cell border; each border can have a different style or color

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)
tcLeftStyle	Left border style
tnLeftColor	Left border color
tcRightStyle	Right border style
tnRightColor	Right border color
tcTopStyle	Top border style
tnTopColor	Top border color
tcBotStyle	Bot border style
tnBotColor	Bot border color
tcDiagStyle	Diag border style
tnDiagColor	Diag border color
tnDiagDownUp	Diag border drawn down/up

Return Value:

True on success

False on failure

SetCellBorderRange (deprecated with Release 18)

Description: Sets the cell border for a range of cells; each border is drawed with the same style and color

Parameters:

tnWB Id to workbook

tnSheet Id to sheet in workbook

tnBegRow Cell beginning row (integer)

tnBegCol Cell beginning column (integer)

tnEndRow Cell ending row (integer)

tnEndCol Cell ending column (integer)

tnBorders Border to draw; this is a combination of the following by adding:

BORDER_LEFT +
BORDER_RIGHT +
BORDER_TOP +
BORDER_BOTTOM +
BORDER_DIAGDOWN +
BORDER_DIAGUP

tcBorderStyle Style of border to draw; the following styles are available:

BORDER_STYLE_THIN
BORDER_STYLE_HAIR
BORDER_STYLE_DOTTED
BORDER_STYLE_DASHDOTDOT
BORDER_STYLE_DASHDOT
BORDER_STYLE_DASHED
BORDER_STYLE_THIN
BORDER_STYLE_MEDIUMDASHDOTDOT
BORDER_STYLE_SLANTDASHDOT
BORDER_STYLE_MEDIUMDASHDOT
BORDER_STYLE_MEDIUMDASHED
BORDER_STYLE_MEDIUM
BORDER_STYLE_THICK
BORDER_STYLE_DOUBLE

tnBorderColor The color to draw the border in RGB() value

Return Value:

True on success

False on failure

SetCellFill (deprecated with Release 18)

Description: Sets the cell fill color (background)

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)
tnFColor	Fill foreground color; RGB(N,N,N) [changed]
tnBColor	Fill background color; RGB(N,N,N) [changed]
tcPatternType	Fill pattern type [changed]

Return Value:

True on success

False on failure

SetCellFillRange (deprecated with Release 18)

Description: Sets the cell fill color (background) for a range of cells

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnBegRow	Cell beginning row (integer)
tnBegCol	Cell beginning column (integer)
tnEndRow	Cell ending row (integer)
tnEndCol	Cell ending column (integer)
tnFColor	Fill foreground color; RGB(N,N,N) [changed]
tnBColor	Fill background color; RGB(N,N,N) [changed]
tcPatternType	Fill pattern type [changed]

Return Value:

True on success

False on failure

SetCellFont (deprecated with Release 18)

Description: Sets the cell format

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)
tcFName	Font name
tnFSize	Font size
tlBold	Boolean to indicate bold font
tlItalic	Boolean to indicate italic font
tnFColor	Font foreground color; RGB(N,N,N)
tcULine	Boolean to indicate underline
tlStrikThr	Boolean to indicate strikethrough

tcVPos	Verical position of text (from #DEFINEs)
	`FONT_VERTICAL_BASELINE`
	`FONT_VERTICAL_SUBSCRIPT`
	`FONT_VERTICAL_SUPERSCRIPT`

Return Value:

True on success
False on failure

SetCellFontRange (deprecated with Release 18)

Description: Sets the cell format for a range of cells

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnBegRow	Cell beginning row (integer)
tnBegCol	Cell beginning column (integer)
tnEndRow	Cell ending row (integer)
tnEndCol	Cell ending column (integer)
tcFName	Font name
tnFSize	Font size
tlBold	Boolean to indicate bold font
tlItalic	Boolean to indicate italic font
tnFColor	Font foreground color; RGB(N,N,N)
tcULine	Boolean to indicate underline
tlStrikThr	Boolean to indicate strikethrough
tcVPos	Verical position of text (see SetCellFont() method for values)

Return Value:

True on success

False on failure

SetCellIndent (deprecated with Release 18)

Description: Sets the cell indentation

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Row to begin the cell merge
tnCellCol	Column to begin the cell merge
tnIndent	Cell indentation value

Return Value:

True if set; False if not set

SetCellNumberDecimals (deprecated with Release 18)

Description: sets the number of decimals to be displayed (used with SetCellNumberFormat)

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)
tnNumDecimals	Number of decimals to be displayed

Return Value:

True on success

False on failure

SetCellNumberFormat (deprecated with Release 18)

Description: Sets the numeric format for the cell value

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)
tnNumFormat	Value of numeric format (from #DEFINEs)

```
CELL_FORMAT_INTEGER
CELL_FORMAT_FLOAT
CELL_FORMAT_COMMA_INTEGER
CELL_FORMAT_COMMA_FLOAT
CELL_FORMAT_CURRENCY_PAREN
CELL_FORMAT_CURRENCY_RED_PAREN
CELL_FORMAT_PERCENT_INTEGER
CELL_FORMAT_PERCENT_FLOAT
CELL_FORMAT_EXPONENT
CELL_FORMAT_FRACTION_1
CELL_FORMAT_FRACTION_2
CELL_FORMAT_DATE_MMDDYY
CELL_FORMAT_DATE_DMMMYY
CELL_FORMAT_DATE_DMMM
CELL_FORMAT_DATE_MMMYY
CELL_FORMAT_TIME_HMMAMPM
CELL_FORMAT_TIME_HMMSSAMPM
CELL_FORMAT_TIME_HMM
CELL_FORMAT_TIME_HMMSS
CELL_FORMAT_DATETIME_MDYYHMM
```

```
CELL_FORMAT_DATETIME_DDMMMYYYY_TTAM
CELL_FORMAT_DATETIME_DDMMMYYYY_TT24
CELL_FORMAT_DATETIME_MMMDDYYYY_TTAM
CELL_FORMAT_DATETIME_MMMDDYYYY_TT24
CELL_FORMAT_DATETIME_MDYY_TTAM
CELL_FORMAT_DATETIME_MDYY_TT24
CELL_FORMAT_COMMA_INTEGER_PAREN
CELL_FORMAT_COMMA_INTEGER_RED_PAREN
CELL_FORMAT_COMMA_FLOAT_PAREN
CELL_FORMAT_COMMA_FLOAT_RED_PAREN
CELL_FORMAT_TIME_MMSS
CELL_FORMAT_TIME_H_MMSS
CELL_FORMAT_CURRENCY_RED
```

Return Value:

True on success

False on failure

SetCellNumberFormatRange (deprecated with Release 18)

Description: Sets the numeric format for a range of cell values

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnBegRow	Cell beginning row (integer)
tnBegCol	Cell beginning column (integer)
tnEndRow	Cell ending row (integer)
tnEndCol	Cell ending column (integer)
tnNumFormat	Value of numeric format (see SetCellNumberFormat() method for list of values)

Return Value:

True on success

False on failure

SetCellTextRotation (deprecated with Release 18)

Description: Sets the cell text rotation

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Row to set the cell text
tnCellCol	Column to set the cell text
tnRotation	Rotation angle to set the text (value between -90 and 90 degrees)

Return Value:

True on success
False on failure

SetCellWordWrap (deprecated with Release 18)

Description: Sets the cell word-wrap value

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)
tlWordWrap	True - set wordwrapping on; False - set wordwrapping off

Return Value:

True on success
False on failure

SetCellWordWrapRange (deprecated with Release 18)

Description: Sets the cell word-wrap value for a range of cells

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnBegRow	Cell beginning row (integer)
tnBegCol	Cell beginning column (integer)
tnEndRow	Cell ending row (integer)
tnEndCol	Cell ending column (integer)
tlWordWrap	True - set wordwrapping on; False - set wordwrapping off

Return Value:

True on success

False on failure

UnMergedCells

Description: Removes the merged cells restoring to individual cells

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnBegRow	Row to begin the cell merge
tnBegCol	Column to begin the cell merge
tnEndRow	Row to end the cell merge
tnEndCol	Column to end the cell merge

Return Value:

True on success

False on failure

Methods – In-Line Text Formatting

AddInLineFontObject

Description: Adds an in-line character definition to the base in-line font definition object

Parameters:

toInline	In-Line Text object
tnBeg	Beginning position for text format in text string
tnLen	Length of text for format in text string
tcFontName	Font name for in-line text
tnFontSize	Font size for in-line text
tnFontColor	Font color for in-line text
tlFontBold	Font bold for in-line text
lFontItalic	Font italic for in-line text
tcULine	Font underline for in-line text
tlStrkThru	Font strike-through for in-line text
tlSubscript	Font subscript for in-line text
tlSuperscript	Font superscript for in-line text

Return Value:

In-Line Character format object added to the In-Line Text object:

loCharacter.BegPos
loCharacter.Length
loCharacter.FontName
loCharacter.FontSize
loCharacter.FontBold
loCharacter.FontItalic
loCharacter.FontColor
loCharacter.Underline
loCharacter.StrikeThru
loCharacter.SubScript
loCharacter.SuperScript

CreateInLineFormatText

Description: Creates the base in-line font object for assigning a text string in a cell to have its characters to be individually formatted

Parameters:

tnWB	Id to workbook
tcCellText	Full text for the cell value

Return Value:

loInline.Workbook	Id to workbook
loInline.StringId	Internal String Id for text string (initially set to NULL)
loInline.StringValue	String value to be assigned to cell
loInline.Count	Count of in-line character format expressions (initially zero)
loInline.Characters[1]	Array of in-line character format expressions (set to NULL)

Null value if cell text not assigned.

GetInLineFormatText

Description: Gets the in-line formatted text definition for a cell text string

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row number
tnCellCol	Cell column number

Return Value:

loInline.Workbook	Id to workbook
loInline.StringId	Internal String Id for text string
loInline.StringValue	String value assigned to cell
loInline.Count	Count of in-line character format expressions
loInline.Characters[n]	Array of in-line character format expressions
loInline.Characters[n].BegPos	nth Beginning position of in-line character format
loInline.Characters[n].Length	nth Length of of in-line character format
loInline.Characters[n].FontName	nth Font name of in-line character format
loInline.Characters[n].FontSize	nth Font size of in-line character format
loInline.Characters[n].FontBold	nth Font bold setting of in-line character format
loInline.Characters[n].FontItalic	nth Font italic setting of in-line character format
loInline.Characters[n].FontColor	nth Font color setting of in-line character format

loInline.Characters[n].Underline	nth Font underline setting of in-line character format
loInline.Characters[n].StrikeThru	nth Font Strike Through setting of in-line character format
loInline.Characters[n].SubScript	nth Font sub-script setting of in-line character format
loInline.Characters[n].SuperScript	nth Font super-script setting of in-line character format

Null value if cell text is not assigned to an in-line format.

SetCellInLineFormatText

Description: Saves an in-line text definition for a text string to a cell

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row number
tnCellCol	Cell column number
toInline	In-Line Text object

Return Value:

True on success; False on failure (this value will be returned if the toInline.Workbook value does not match the tnWB value)

Comments:

If the toInline.Workbook value does not match the tnWB value, False will be returned (no assignment). You can use the same loInLine object to assign the same in-line formatted text to multiple spreadsheet cells within the same workbook (i.e., different sheets).

Methods – Numeric Cell Formats

AddCustomNumericFormat

Description: Adds a new definition for a numeric format

Parameters:

tcPosSect	Format for positive numbers; is the format code that applies to the cell when the cell value contains a positive number. [required]
tcNegSect	Format for negative numbers; is the format code that applies to the cell when the cell value contains a negative number. [optional]
tcZeroSect	Format for zeros; is the format code that applies to the cell when the cell value is zero. [optional]
tcTextSect	Format for text; is the format code that applies to the cell when the cell value is text. [optional]
tlApplyDec	Flag to set the number of decimals as determined by the method SetCellNumberDecimals(); defaults to False [optional]

Return Value:

Id of format

0 on failure

AddNumericFormat

Description: Adds a new definition for a numeric format (full format must be specified) [retained for backward compatibility]

Parameters:

tcFormatCode	Numeric format to be added

Return Value:

Id of format

0 on failure

GetCellNumberFormat

Description: Returns the format code for the selected cell

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)

Return Value:

Number format code

Zero if none or failure

GetCellNumberFormatText

Description: Returns the format text for the selected cell

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row (integer)
tnCellCol	Cell column (integer)

Return Value:

Number format text string

Empty string if none or failure

GetCustomNumericFormat

Description: Returns the specified numeric custom format code

Parameters:

tnWB	Id to workbook [changed]
tnFormatCode	Format Id to return

Return Value:

Numeric Format code; empty string if none.

Methods – Cell Validations and Named Ranges

AddNamedRange

Description: Adds a new named range of cells

Parameters:

tnWB	Id to workbook
tnSheet	Sheet index of the named range
tcName	Range name
tnScope	Scope of named range; use value from #DEFINEs SCOPE_WB_NAMED_RANGE SCOPE_SH_NAMED_RANGE
tcComment	Comment for named range
tnBegRow	Named range cell beginning row number
tnBegCol	Named range cell beginning column number
tnEndRow	Named range cell ending row number
tnEndCol	Named range cell ending column number

Return Value:

Range name (replaces spaces with underscore character)

Empty string on failure

ClearCellValidation

Description: Removes any cell validations

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row number
tnCellCol	Cell column number

Return Value:

True on success

False on failure

ClearNamedRange

Description: Removes the named range from the workbook

Parameters:

tnWB	Id to workbook
tcName	Range name

Return Value:

True on success

False on failure

GetCellValidation

Description: Gets the cell validation formula settings

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnCellRow	Cell row number
tnCellCol	Cell column number

Return Value:

Validation Object, loValidation with the following properties:

loValiation.Type
loValiation.Style
loValiation.Operator
loValiation.AllowBlank
loValiation.ShowInputMsg
loValiation.ShowErrMsg
loValiation.ErrMsg
loValiation.ErrTitle
loValiation.Prompt
loValiation.Formula1
loValiation.Formula2

GetValidation

Description: Returns an object with the validation definition

Parameters:

tnValidNdx	Validation index

Return Value:

Validation Object, loValidation with the following properties:

loValiation.Type
loValiation.Style
loValiation.Operator
loValiation.AllowBlank
loValiation.ShowInputMsg
loValiation.ShowErrMsg
loValiation.ErrMsg
loValiation.ErrTitle
loValiation.Prompt
loValiation.Formula1
loValiation.Formula2

GetValidationList

Description: Returns an object with the list of validations for the workbook/sheet

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook

Return Value:

Validation Object, loValidation with the following properties:

loValiation.Count
loValiation.List[1, 1] = Validation Type
loValiation.List[1, 2] = Validation Index

SetCellValidation

Description: Sets cell validation

Parameters:

tnWB Id to workbook

tnSh Id to sheet in workbook

tnCellRow Cell row number

tnCellCol Cell column number

tnType Cell validation type; use #DEFINEs for value

```
NONE_VALID_TYPE            DATE_VALID_TYPE
WHOLE_VALID_TYPE           TIME_VALID_TYPE
DECIMAL_VALID_TYPE         TXTLEN_VALID_TYPE
LIST_VALID_TYPE            CUSTOM_VALID_TYPE
```

tnStyle Cell validation style [optional; defaults to none] ; use #DEFINEs for value

```
STOP_VALID_STYLE
WARN_VALID_STYLE
INFO_VALID_STYLE
```

tnOperator Cell validation operator [optional; defaults to none] ; use #DEFINEs for value

```
BETWEEN_VALID_OPER            LESSTHAN_VALID_OPER
NOTBETW_VALID_OPER            LESSOREQUAL_VALID_OPER
EQUAL_VALID_OPER              GREATTHAN_VALID_OPER
NOTEQUAL_VALID_OPER           GREATOREQUAL_VALID_OPER
```

tlAllowBlank Boolean to indicate if cell value can be blank [default true]

tShowInputMsg Boolean to show input message [default true]

tlShowErrMsg Boolean to show error message [default true]

tcErrMsg Cell error message to display to user; limited to 100 characters [optional; defaults to none]

tcErrTitle Cell error title on message displayed; limited to 100 characters [optional; defaults to none]

tcPrompt Cell prompt information to user; limited to 100 characters [optional; defaults to none]

tcFormula Cell validation formula; limited to 254 characters; a list of allowed values is separated by commas

Return Value:

True on success

False on failure

Methods – Set Sheet Formatting

FreezePanes

Description: Provides for freezing the upper rows and left columns for scrolling

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnTopRowCount	Number of rows to freeze at the top
tnSideColCount	Number of columns to freeze at the left

Return Value:

True on success

False on failure

ResetColumnWidth

Description: Resets the column width to the default of Excel

Parameters:

tnWB	Id to workbook
tnSh	Id to sheet in workbook
tnColumn	Column index to reset width

Return Value:

True on success

False on failure

SetColumnBestFit

Description: Sets the column width to best fit (this method is not yet fully working and is not currently saved in the sheet).

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnColumn	Column index (integer) to set to best fit
tlBestFit	Boolean value; True set to best fit, False do not set

Return Value:

True on success

False on failure

SetColumnHidden

Description: Sets the selected column hidden setting

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnColumn	Column index (integer) to set the width of
tlHidden	True to set to hidden; False to set to not hidden

Return Value:

True on success

False on failure

SetColumnWidth

Description: Sets the selected column width

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnColumn	Column index (integer) to set the width of
tnWidth	Value to set the column width to

Return Value:

True on success

False on failure

SetColumnWidthRange

Description: Sets the column width for a range of columns

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnBegCol	Beginning column index (integer) to set the width of
tnEndCol	Ending column index (integer) to set the width of
tnWidth	Value to set the column width to

Return Value:

True on success

False on failure

SetHeaderFooterSetup

Description: Sets the properties for the header /footer in the sheet (Align to margins, different first page, different odd/even pages, and scale with print). This method must be set before calling SetHeaderFooterText() method.

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tlAlignMargin	Boolean; True – align with margins, False – fixed position
tlDiffFirstPg	Boolean; True – different first page, False – same as odd page
tlDiffOddEven	Boolean; True – different odd/even pages, False – same as odd page
tlScaleWDoc	Boolean; True – scale size with sheet scalling factor; False – fixed

Return Value:

True on success

False on failure

SetHeaderFooterText

Description: Sets the header text

Parameters:

tnWB Id to workbook

tnSheet Id to sheet in workbook

tnPage Page to apply header/footer text; i.e., first page, odd page, or even page; use #DEFINEs values (use same page for same odd and even pages)

HEADERFOOTER_FIRST_PAGE HEADERFOOTER_EVEN_PAGE
HEADERFOOTER_ODD_PAGE HEADERFOOTER_SAME_PAGE

tnSection Position of the text (i.e., Left, Center, or Right); use #DEFINEs values

HEADERFOOTER_POS_FTR_LEFT HEADERFOOTER_POS_HDR_LEFT
HEADERFOOTER_POS_FTR_CENTER HEADERFOOTER_POS_HDR_CENTER
HEADERFOOTER_POS_FTR_RIGHT HEADERFOOTER_POS_HDR_RIGHT

tcText Header text

tcFontName Font name of header/footer text [optional]

tnFontSize Font size of header/footer text [optional]

tnFontStyle Font effect of header/footer text [optional]; i.e., normal, italic, or bold; use #DEFINEs values

HEADERFOOTER_FONT_STYLE_NORMAL HEADERFOOTER_FONT_STYLE_ITALIC
HEADERFOOTER_FONT_STYLE_BOLD HEADERFOOTER_FONT_STYLE_BOLDITALIC

Return Value:

True on success

False on failure

SetRowHeight

Description: Sets the selected row height

tnWB Id to workbook

tnSheet Id to sheet in workbook

tnRow Row index (integer) to set the height of

tnHeight Value to set the row height to

Return Value:

True on success

False on failure

SetRowHeightRange

Description: Sets the selected row height

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnBegRow	Beginning row index (integer) to set the height of
tnEndRow	Ending row index (integer) to set the height of
tnHeight	Value to set the row height to

Return Value:

True on success

False on failure

UnFreezePanes

Description: Removes all of the panes that are frozen (top and side)

tnWB	Id to workbook
tnSheet	Id to sheet in workbook

Return Value:

True on success

False on failure

Methods – Sheet Printer Setup

GetCustomPaperSize

Description: Gets the values for the custom paper size

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook

Return Value:

Return object:

loReturn.PaperWidth	Paper width value
loReturn.PapeHeight	Paper height value
loReturn.PaperDimen	Paper width/height unit of measurement (in or mm)

GetPaperSize

Description: Gets the paper size for the selected sheet

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook

Return Value:

Paper size value (see SetPaperSize() method for a list of values)

-1 on failure or none set

GetPrintOrientation

Description: Gets the print orientation for the sheet output

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook

Return Value:

Printer orientation; numeric value (see the #DEFINE list of values)

Zero on failure or none set

GetSheetScale

Description: Gets the sheet printing scale

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook

Return Value:

Printer scale value (numeric)

-1 on failure or none set

SetCustomPaperSize

Description: Sets the paper size based on custom dimensions

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnWidth	Paper width (numeric value)
tnHeight	Paper height (numeric value)
tcDimen	Unit of measurement (in or mm)

Return Value:

True on success

False on failure

SetPaperSize

Description: Sets the paper size for the selected sheet

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnPaperSize	The paper size to set (see the #DEFINEs list of values)

```
PAPERSIZE_LTR
PAPERSIZE_LTR_SMALL
PAPERSIZE_TABLOID
PAPERSIZE_LEDGER
PAPERSIZE_LEGAL
PAPERSIZE_STATEMENT
PAPERSIZE_EXECUTIVE
PAPERSIZE_A3
PAPERSIZE_A4
PAPERSIZE_A4_SMALL
PAPERSIZE_A5
PAPERSIZE_B4
PAPERSIZE_B5
PAPERSIZE_FOLIO
PAPERSIZE_QUARTO
PAPERSIZE_STD10X14
PAPERSIZE_STD11X17
PAPERSIZE_NOTE
PAPERSIZE_9ENV
PAPERSIZE_10ENV
PAPERSIZE_11ENV
PAPERSIZE_12ENV
PAPERSIZE_14ENV
PAPERSIZE_C
PAPERSIZE_D
PAPERSIZE_E
PAPERSIZE_DL_ENV
PAPERSIZE_C5_ENV
PAPERSIZE_C3_ENV
PAPERSIZE_C4_ENV
PAPERSIZE_C6_ENV
PAPERSIZE_C65_ENV
PAPERSIZE_B4_ENV
PAPERSIZE_B5_ENV
PAPERSIZE_B6_ENV
PAPERSIZE_ITALY_ENV
PAPERSIZE_MONARCH_ENV
PAPERSIZE_6_3_4_ENV
PAPERSIZE_US_STD_FANFOLD
PAPERSIZE_GERMAN_STD_FANFOLD
PAPERSIZE_GERMAN_LGL_FANFOLD
PAPERSIZE_ISO_B4
PAPERSIZE_JPN_DBL_POSTCARD
PAPERSIZE_STD_PAPER9X11
PAPERSIZE_STD_PAPER10X11
PAPERSIZE_STD_PAPER15X11
PAPERSIZE_INVITE_ENV
PAPERSIZE_A5_TRANSVERSE
PAPERSIZE_JIS_B5_TRANSVERS
PAPERSIZE_A3_EXTRA
PAPERSIZE_A5_EXTRA
PAPERSIZE_ISO_B5_EXTRA
PAPERSIZE_A2
PAPERSIZE_A3_TRANSVERSE
PAPERSIZE_A3_EXTRA_TRANSVE
PAPERSIZE_JPN_DOUBLE
PAPERSIZE_A6
PAPERSIZE_JPN_ENV_KAKU1
PAPERSIZE_JPN_ENV_KAKU2
PAPERSIZE_JPN_ENV_CHOU3
PAPERSIZE_JPN_ENV_CHOU4
PAPERSIZE_LTR_ROT
PAPERSIZE_A3_ROT
PAPERSIZE_A4_ROT
PAPERSIZE_A5_ROT
PAPERSIZE_B4_JIS_ROT
PAPERSIZE_B5_JIS_ROT
PAPERSIZE_JPN_POSTCARD
PAPERSIZE_DOUBLE_JPN
PAPERSIZE_A6_ROT
PAPERSIZE_JPN_ENV_KAKU2_ROT
PAPERSIZE_JPN_ENV_KAKU3_ROT
PAPERSIZE_JPN_ENV_CHOU3_ROT
PAPERSIZE_JPN_ENV_CHOU4_ROT
PAPERSIZE_B6_JIS
PAPERSIZE_B6_JIS_ROT
PAPERSIZE_12X11
PAPERSIZE_JPN_ENV_YOU4
PAPERSIZE_JPN_ENV_YOU4_ROT
PAPERSIZE_PRC_16K
PAPERSIZE_PRC_32K
PAPERSIZE_PRC_32K_BIG
PAPERSIZE_PRC_ENV_1
PAPERSIZE_PRC_ENV_2
PAPERSIZE_PRC_ENV_3
PAPERSIZE_PRC_ENV_4
PAPERSIZE_PRC_ENV_5
PAPERSIZE_PRC_ENV_6
PAPERSIZE_PRC_ENV_7
PAPERSIZE_PRC_ENV_8
PAPERSIZE_PRC_ENV_9
PAPERSIZE_PRC_ENV_10
PAPERSIZE_PRC_16K_ROT
PAPERSIZE_PRC_32K_ROT
```

PAPERSIZE_LTR_XTRA_PAPER
PAPERSIZE_LEGAL_XTRA_PAPER
PAPERSIZE_TABLOID_XTRA_PAPER
PAPERSIZE_A4_XTRA_PAPER
PAPERSIZE_LTR_TRANSVERSE
PAPERSIZE_A4_TRANSVERSE
PAPERSIZE_LTR_XTRA_TRANSV
PAPERSIZE_SUPERA_A4
PAPERSIZE_SUPERB_A3
PAPERSIZE_LTR_PLUS
PAPERSIZE_A4_PLUS
PAPERSIZE_PRC_32K_BIG_ROT
PAPERSIZE_PRC_ENV_1_ROT
PAPERSIZE_PRC_ENV_2_ROT
PAPERSIZE_PRC_ENV_3_ROT
PAPERSIZE_PRC_ENV_4_ROT
PAPERSIZE_PRC_ENV_5_ROT
PAPERSIZE_PRC_ENV_6_ROT
PAPERSIZE_PRC_ENV_7_ROT
PAPERSIZE_PRC_ENV_8_ROT
PAPERSIZE_PRC_ENV_9_ROT
PAPERSIZE_PRC_ENV_10_ROT

Return Value:

True on success
False on failure

SetPrintFitToHeight

Description: Number of vertical pages to fit on

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnFitToHeight	Number of pages to fit to height

Return Value:

True on success
False on failure

SetPrintFitToWidth

Description: Number of horizontal pages to fit on

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnFitToWidth	Number of pages to fit to width

Return Value:

True on success
False on failure

SetPrintOrientation

Description: Sets the printer orientation for sheet output

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnOrientation	The printer orientation to set PORTRAIT_PRINT_ORIENTATION LANDSCAPE_PRINT_ORIENTATION

Return Value:

True on success; False on failure

SetSheetMargins

Description: Sets the margins of the sheet

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook
tnLeft	Value for left margin
tnRight	Value for right margin
tnTop	Value for top margin
tnbot	Value for bot margin
tnHeader	Value for header margin
tnFooter	Value for footer margin

Return Value:

True on success; False on failure

SetSheetScale

Description: Sets the print scale; must be between 10 and 400; i.e. 10=10%, 50=50%, 100=100%, 175=175%, etc.

Parameters:

tnWB	Id to workbook
tnSheet	Id to sheet in workbook

Return Value:

True on success; False on failure

Methods – Direct VFP Table Support

SaveGridToWorkbook (parameter change with Release 28)

Description: Saves the passed grid to a workbook in xlsx file format. Uses the grid column widths to set the workbook column widths. Adds a new sheet for each passed grid if the same workbook name. Uses the grid formatting to determine the xlsx cell format properties; including the dynamic column properties.

Parameters:

toGrid	Object reference to the grid to be saved
txWB	Integer value: Workbook integer value as returned by CreateWorkbook() method; String value: Workbook file name to be created
tlFreeze	[optional] Boolean to set the FreezePanes on the first row; defaults to True
tlSaveWB	[optional] Boolean to save the workbook to file; defaults to True
tcSheetName	[optional] Name of sheet to add; defaults to table alias
tlInclHiddenCols	[optional] Indicates whether to include hidden columns during export; True – hidden columns are exported, False – hidden columns are not exported. Default is True.
tlShowGridLines	[optional] Indicates whether to hide or show the gridlines. True shows grid lines and False hides grid lines; defaulted from Grid property settings [new parameter]

Return Value:

Return object:

loReturn.Workbook	Workbook Id; zero on failure
loReturn.Sheet	Sheet Id; zero on failure

SaveGridToWorkbookEx (parameter change with Release 28)

Description: Saves the passed grid to a workbook in xlsx file format by writing directly to the XLSX files and does not write to the internal cursors; hence, this is the fastest way to create a XLSX file from a grid.

Parameters:

toGrid	Object reference to the grid to be saved
tcFileName	String value: Workbook file name to be created
tlFreeze	[optional] Boolean to set the FreezePanes on the first row; defaults to True
tcSheetName	[optional] Name of sheet to add; defaults to table alias
tlInclHiddenCols	[optional] Indicates whether to include hidden columns during export; True – hidden columns are exported, False – hidden columns are not exported. Default is True.
tlShowGridLines	[optional] Indicates whether to hide or show the gridlines. True shows grid lines and False hides grid lines; defaulted from Grid property settings [new parameter]

Return Value:

True on success

False on failure

SaveTableToWorkbook

Description: Saves the passed table to a workbook in xlsx file format. Adds a new sheet for each passed table if the same workbook name.

Parameters:

tcAlias	This can be the table alias (table already opened) or this can be the full path and name to a table
txWB	Integer value: Workbook integer value as returned by CreateWorkbook(); String value: Workbook file name to be created
tlFreeze	[optional] Boolean to set the FreezePanes on the first row; defaults to True
tlSaveWB	[optional] Boolean to save the workbook to file; defaults to True
tcSheetName	[optional] Name of sheet to add; defaults to table alias

Return Value:

Return object:

loReturn.Workbook	Workbook Id; zero on failure
loReturn.Sheet	Sheet Id; zero on failure

SaveTableToWorkbookEx

Description: Saves the passed table to a workbook in xlsx file format by writing directly to the XLSX files and does not write to the internal cursors; hence, this is the fastest way to create a XLSX file from a table or cursor. You can also pass an array of the fields that are to be included in the export.

Parameters:

tcAlias This can be the table alias (table already opened) or this can be the full path and name to a table

tcXlsxName String value: Workbook file name to be created

taFields [optionall] Array that has at least two columns. The first array column is the field name to export and the second array column is the field title to be displayed in the first row of the spreadsheet.

tlFreeze [optional] Boolean to set the FreezePanes on the first row; defaults to True

tcSheetName [optional] Name of sheet to add; defaults to table alias

Return Value:

True on success

False on failure

Methods – Support

ColumnAsciiToIndex

Description: Converts a Excel notation column reference (ASCII character) to an numeric (integer) column reference

Parameters:

tcCol	ASCII value of column

Return Value:

Integer of column index

ColumnIndexToAscii

Description: Converts a numeric (integer) column reference to an ASCII character column reference

Parameters:

tnCol	Integer value of column to convert to ASCII

Return Value:

ASCII equilvalent of column index

ConvertPixelsToExcelUnits

Description: Converts pixels in VFP to Excel units for column widths

Parameters:

tnCol	Pixel value

Return Value:

Excel value

ConvertRangeToColumnRowValues

Description: Converts a given range notation to row and column values

Parameters:

tcCellRange Cell range notation; i.e., “A1:B34”

Return Value:

Range object;

loRange.BegCol
loRange.BegRow
loRange.EndCol
loRange.EndRow

DebugOutput

Description: used for debugging

Parameters:

None

Return Value:

None

Demo

Description: Demo code examplesof the various features of this class

Parameters:

None

Return Value:

None

GetImageDimensions

Description: Gets the image height and width dimensions for inserting into a sheet

Parameters:

tcImageFile	File name and full path of the image file

Return Value:

Image object:

loDimens.Width

loDimens.Height

GetImageRelationshipId

Description: Gets the relationship Id for an image based on the workbook, sheet and position

Parameters:

tnWB	Id to workbook to add sheet to
tnSh	Id to sheet in workbook
tnBegRow	Beginning row index for image
tnBegCol	Beginning column index for image
tnEndRow	Ending row index for image
tnEndCol	Ending column index for image

Return Value:

Image relationship Id (assigned by AddImage method); 0 if not found

ParseString

Description: Replacement for GETWORDNUM function (fixes problem of parsing a string that has a null value for one of the tokens)

Parameters:

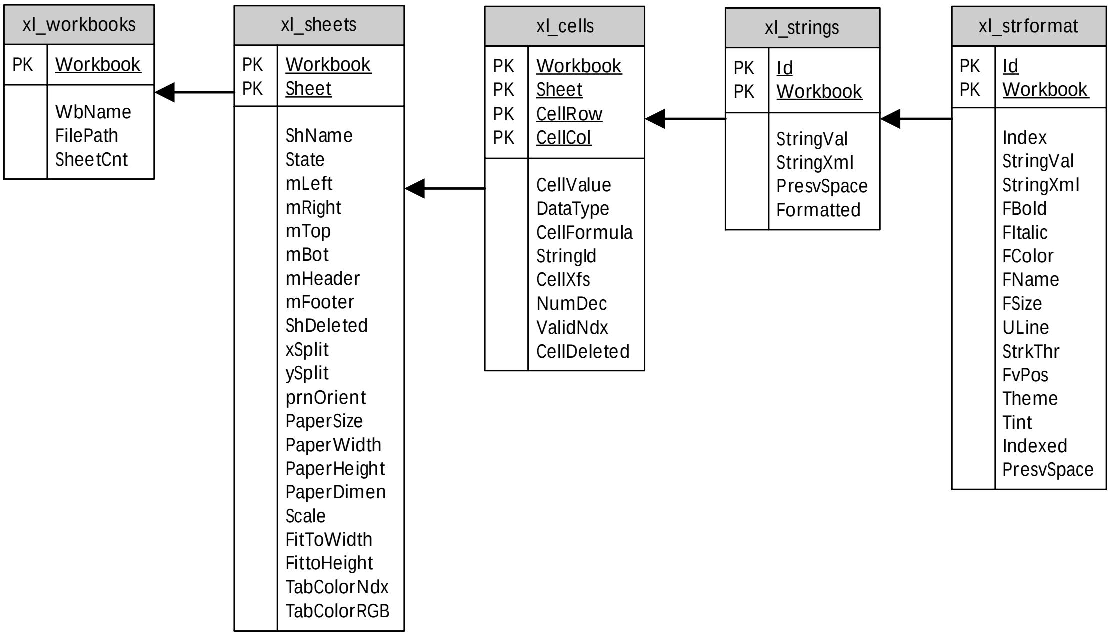
tcText	Text string to parse
tnPos	The token to be returned in the string
tcDelimiter	The delimiter for the string

Return Value:

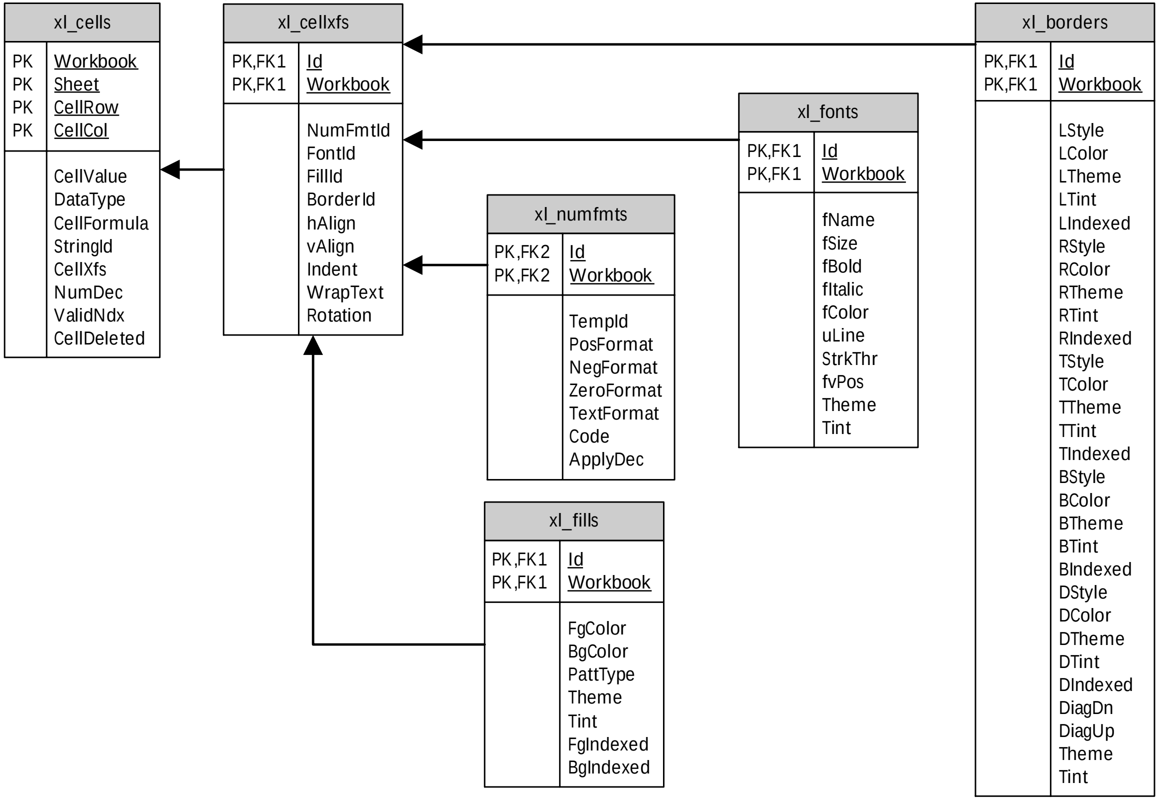
The text token.

Entity Diagrams

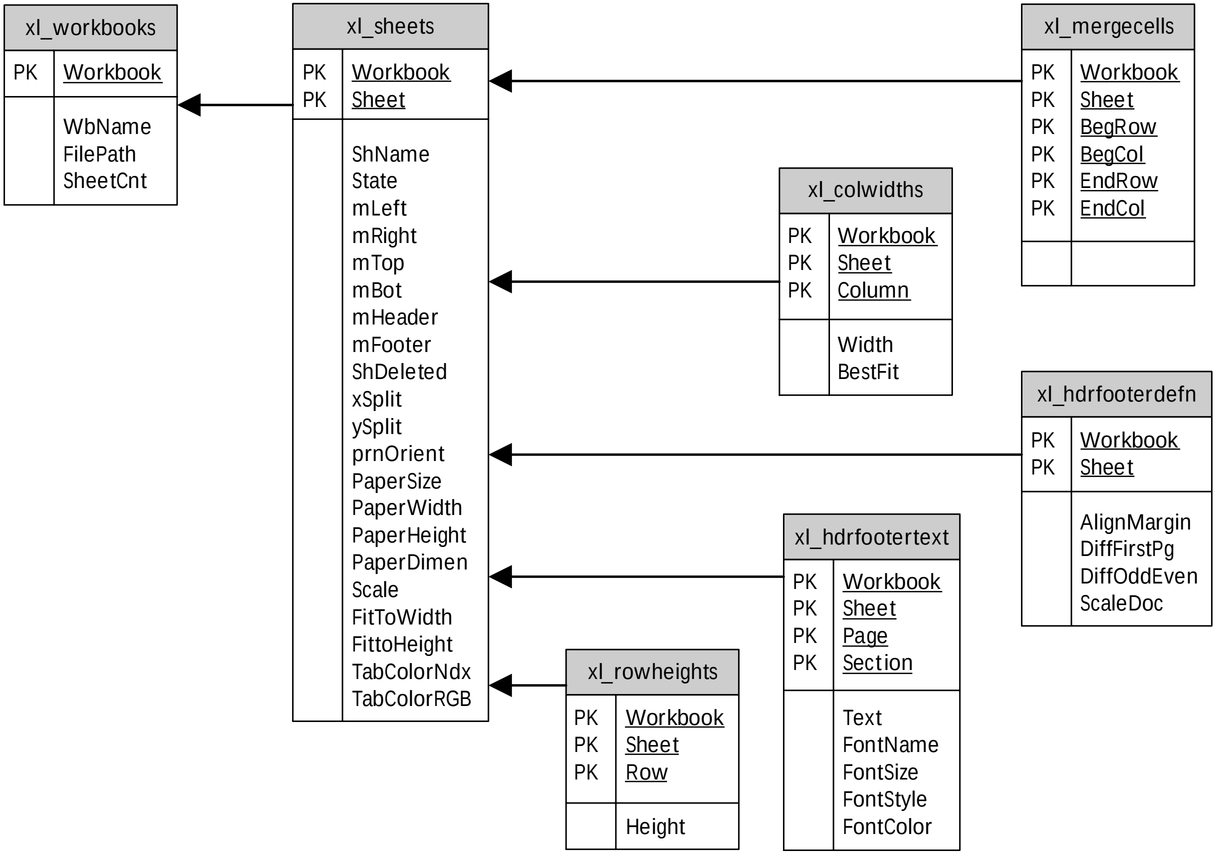
Sheet/Cell Data Schema



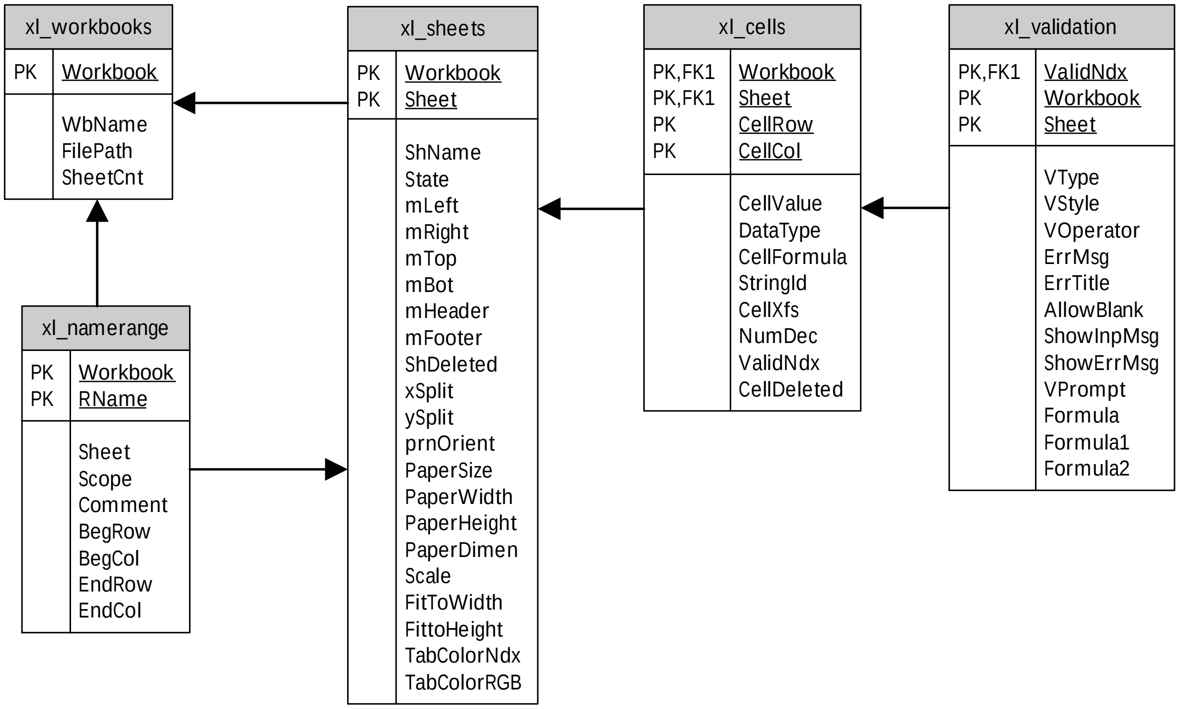
Cell Formatting Schema



Sheet Formatting Schema



Validation Schema


xl_workbooks
PK Workbook
WbName
FilePath
SheetCnt
xl_sheets
PK Workbook
PK Sheet
ShName
State
mLeft
mRight
mTop
mBot
mHeader
mFooter
ShDeleted
xSplit
ySplit
prnOrient
PaperSize
PaperWidth
PaperHeight
PaperDimen
Scale
FitToWidth
FittoHeight
TabColorNdx
TabColorRGB
xl_cells
PK,FK1 Workbook
PK,FK1 Sheet
PK CellRow
PK CellCol
CellValue
DataType
CellFormula
StringId
CellXfs
NumDec
ValidNdx
CellDeleted
xl_validation
PK,FK1 ValidNdx
PK Workbook
PK Sheet
VType
VStyle
VOperator
ErrMsg
ErrTitle
AllowBlank
ShowInpMsg
ShowErrMsg
VPrompt
Formula
Formula1
Formula2
xl_namerange
PK Workbook
PK RName
Sheet
Scope
Comment
BegRow
BegCol
EndRow
EndCol