

SMExport suite

[Registration](#)

Scalabium

Scalable solutions that grow with you...

The SMExport suite for Delphi/C++Builder allows to export a data from linked recordset (any inherited from TDBGrid or TDataset types) into:

1. MS Excel spreadsheet
2. MS Word file
3. MS Access database
4. HTML file
5. XML file
6. Text (fixed or CSV) file
7. PDF (Adobe Acrobat)
8. RichText format
9. SYLK (Symbolic Link) file
10. DIF (Data Interchange Format) file
11. LDAP Data Interchange Format (directory entries)
12. Lotus 1-2-3 spreadsheet
13. QuattroPro file
14. SQL-script with data dump
15. DBase table
16. Paradox table
17. SPSS spreadsheet
18. MS Windows clipboard
19. other dataset.

Also SMExport suite contain the two compaund component with visual dialogs for easy defining of export process settings. With these visual components you can allows for your end-user to setup an export parameters.

Custom data engines allow to export data from any control (TStringGrid, TListView,

TDecisionCube etc). Virtual data engine allow to generate a data in run-time without any linked control.

Data pumping components allow to export a lot of tables from database in a few mouse clicks.

Visit the web site at <http://www.scalabium.com> to see if updated help files is available.

Registration

[By credit card](#) [By mail](#)

Why Register

Thank you for your interest in SMExport suite.

Registered users will receive the latest registered version of SMExport suite, free on-line support, and the source code.

Online registration

You can order a product online at:

1. PayPro: <https://secure.payproglobal.com/orderpage.aspx?products=51379>

Questions & Comments

Please refer questions or comments about SMExport suite to:

<http://www.scalabium.com>
<mailto:mshkolnik@scalabium.com>

TSMEColumnBand component

[Properties](#) [Methods](#)

Unit

[SMEEngine](#)

Description

Represents a band for export component (grouping for columns).

Combined columns can be positioned into several rows within a band.

Alignment property

Applies to

[TSMEColumnBand](#) component

Specifies the band caption text alignment.

Declaration

```
property Alignment: TAlignment;
```

Description

Use the alignment property to specify the band caption text alignment.

Caption property

Applies to

[TSMEColumnBand](#) component

Specifies the text representing a band.

Declaration

```
property Caption: string;
```

Description

Use the Caption property to specify the text representing a band. This text is exported above group of [column titles](#) .

The Caption text position within the band header is determined via the [Alignment](#) property.

Color property

Applies to

[TSMEColumnBand](#) component

Declaration

property Color: TColor;

Description

Use the Color property to specify the background color for band.

Font property

Applies to

[TSMEColumnBand](#) component

Declaration

property Font: TFont;

Description

Use the Font property to specify the font attributes for band.

Visible property

Applies to

[TSMEColumnBand](#) component

Specifies the current band's visibility.

Declaration

property Visible: Boolean;

Description

Use the Visible property to specify whether the current band is exported.

To make the band invisible, set this property value to False. If the band is hidden, then all columns within it are not exported also.

The default value of the Visible property is True.

TSMEColumnBands component

[Properties](#) [Methods](#)

Unit

[SMEEngine](#)

Description

Represents the bands collection of a exported data engine.
You may combine the columns in groups (named as bands)

You may customize any property for every [band](#)

Items property

Applies to

[TSMEColumnBands](#) component

Declaration

property Items[Index: Integer]: [TSMEColumnBand](#);

Description

The value of the Index parameter corresponds to the Index property of [TSMEColumn](#). It represents the position of the band in list of defined items.

Add method

[Example](#)

Applies to

[TSMEColumnBands](#) component

Adds a new band into the current collection.

Declaration

function Add: [TSMEColumnBand](#) ;

Description

Use the Add method to add a new band to the current collection.

After a band is added, you can specify its settings, such as caption, options and colors, etc.

TSMEColumnTitle object

[Properties](#) [Methods](#)

Unit

[SMEEngine](#)

The TSMEColumnTitle that represents the column's title.

Description

Export components uses a [TSMEColumns](#) to maintain a collection of [TSMEColumn](#) objects.

Each [TSMEColumn](#)

has an associated TSMEColumnTitle that holds information about its title. The TSMEColumnTitle instance is stored in the column's Title property.

You can customize any visual property for each caption of any column - font, alignment, color etc.

Alignment property

[See also](#)

Applies to

[TSMEColumnTitle](#) object

Specifies how text is aligned within the column title.

Declaration

property Alignment: TAlignment;

Description

These are the possible values of Alignment:

Value Meaning

taLeftJustify Align text on the left side of the column.

taCenter Center text in the column.

taRightJustify Align text on the right side of the column.

Caption property

[See also](#)

Applies to

[TSMEColumnTitle](#) object

The text that appears at the top of the column.

Declaration

```
property Caption: string;
```

Description

The Caption property contains a text string that identifies the column.

If the FieldName property is set in [TSMEColumn](#), the Caption default value comes from Field.DisplayLabel which itself defaults to FieldName.

Color property

[See also](#)

Applies to

[TSMEColumnTitle](#) object

The background color for the column title.

Declaration

property Color: TColor;

Description

The Color property determines the background color of the column title. You can set Color to one of the constants defined in the Graphics unit (such as clBlue), or to an explicit RGB integer value. The default value is the color of column in linked DBGrid.

Font property

[See also](#)

Applies to

[TSMEColumnTitle](#) object

Controls the font in which the column title displays its caption.

Declaration

```
property Font: TFont;
```

Description

The Font property points to a TFont object that determines typographic attributes of text displayed in the column title. The default value is the font which was assigned to column in linked DBGrid property.

TSMEColumn component

[See also](#) [Properties](#) [Methods](#)

Unit

[SMEEngine](#)

TSMEColumn represents a column in an export component.

Description

Each export component uses the [TSMEColumns](#) to maintain a collection of TSMEColumn objects.

Alignment property

[See also](#)

Applies to

[TSMEColumn](#) component

Specifies how text is aligned within the column.

Declaration

property Alignment: TAlignment;

Description

These are the possible values of Alignment:

Value Meaning

taLeftJustify Align text on the left side of the column.

taCenter Center text in the column.

taRightJustify Align text on the right side of the column.

BandIndex property

[See also](#) [Example](#)

Applies to

[TSMEColumn](#) component

Specifies the band which exports the current column.

Declaration

property BandIndex: Integer;

Description

Use the BandIndex property to define the band exporting the current column. The BandIndex property addresses a band by its index in the Bands collection.

If you set BandIndex to -1, the column will be removed from the band.

Color property

[See also](#)

Applies to

[TSMEColumn](#) component

The background color for the column.

Declaration

property Color: TColor;

Description

The Color property determines the background color of the exported column. You can set Color to one of the constants defined in the Graphics unit (such as clBlue), or to an explicit RGB integer value.

ColumnKind property

Applies to

[TSMEColumn](#) component

Declaration

property ColumnKind: [TSMEColumnKind](#);

Description

This property specify the kind of data in column.

DataType property

Applies to

[TSMEColumn](#) component

Declaration

property DataType: [TCellType](#);

Description

This property specify the type of data in column.

FieldName property

[See also](#)

Applies to

[TSMEColumn](#) component

The name of the field represented by the column.

Declaration

```
property FieldName: string;
```

Description

Setting FieldName changes the Field property so that it points to the dataset field with the same name. If the dataset does not have a field with the same name, Field is set to nil.

For non-dataset sources (ListView, StringGrid etc) the next rules are for FieldName property:

1. TListView/TRZListView/TStringGrid/TdxDBTreeList/TDecisionCube/TFxCube:
Field1, Field2, ... where 1, 2, ... is an index of column

For example, Columns.Add.FieldName := strField + '2'

2. TStrings/TStringList:
Text

For example, Columns.Add.FieldName := strTextFieldName

Font property

[See also](#)

Applies to

[TSMEColumn](#) component

Controls the font in which the column exports its data.

Declaration

```
property Font: TFont;
```

Description

The Font property points to a TFont object that determines typographic attributes of text exported in the column.

Precision property

Applies to

[TSMEColumn](#) component

Declaration

property Precision: Integer;

Description

Precision determines the precision used in formatting the value in a floating-point column.

Title property

Applies to

[TSMEColumn](#) component

Declaration

property Title: [TSMEColumnTitle](#);

Description

The Title property points to a [TSMEColumnTitle](#) object that determines attributes of the column's title.

If FieldName is set, the value of FieldName becomes the default column title ([TSMEColumnTitle.Caption](#)).

Visible property

Applies to

[TSMEColumn](#) component

Specifies whether the column is exported in the target file.

Declaration

```
property Visible: Boolean;
```

Description

To hide a column in the target file with exported data, set Visible to False. If Visible is True, the column will be exported in the file. This property only determines whether the column is allowed to be exported.

Width property

Applies to

[TSMEColumn](#) component

The width of the column.

Declaration

```
property Width: Integer;
```

Description

The Width property determines the width of the column, in characters.

GetItemCaption method

Applies to

[TSMEColumn](#) component

Declaration

```
function GetItemCaption(FieldDisplayFormat:  
TSMEFieldDisplayFormat): string;
```

Description

This method used internally by export wizard to get the caption of column to display in list of available columns for export.

TSMEColumns component

[See also](#) [Properties](#) [Methods](#)

Unit

[SMEEngine](#)

Description

Each TSMEColumns holds a collection of [TSMEColumn](#) objects in an export component. TSMEColumns maintains an index of the columns in its Items array. The Count property contains the number of columns in the collection.

At design time, use the Columns editor to add, remove, or modify columns.

Items property

Applies to

[TSMEColumns](#) component

An index of the columns in the collection.

Declaration

property Items[Index: Integer]: [TSMEColumn](#);

Description

The value of the Index parameter corresponds to the Index property of TSMEColumn. It represents the position of the column in the exported data file.

Add method

Applies to

[TSMEColumns](#) component

Creates a new TSMEColumn instance and adds it to the Items array.

Declaration

```
function Add: TSMEColumn;
```

Description

Add returns the new column.

At design time, use the Columns editor to add columns to the export component.

ColumnByFieldName method

Applies to

[TSMEColumns](#) component

Locates the column exporting a particular field's values.

Declaration

```
function ColumnByFieldName(const AFieldName: string): Integer;
```

Description

Use the GetColumnByFieldName method to locate the column exporting values from the specified data set field.

The field name is specified via the AFieldName parameter.

If a column corresponding to the specified field name does not exist or the data set contains no field with the specified name, then this method returns **nil**.

MergedColCount method

Applies to

[TSMEColumns](#) component

Declaration

```
function MergedColCount(BandIndex, StartColIndex: Integer):  
Integer;
```

Description

This method returns the number of columns in band (defined by BandIndex parameter)

MergedColStart method

Applies to

[TSMEColumns](#) component

Declaration

```
function MergedColStart(BandIndex: Integer): Integer;
```

Description

This method returns the index of first column in band.

TSMEPageSetup object

[See also](#) [Properties](#) [Methods](#)

Unit

[SMEEngine](#)

Description

Some export formats (MS Word, MS Excel, PDF etc) allow to customize the page settings. This type contains all the page setup attributes (margins, size and so on) as properties.

```
SMExportToWord1.PageSetup.UseDefault := False;  
SMExportToWord1.PageSetup.Measure := emInch;  
SMExportToWord1.PageSetup.LeftMargin := 0.5;  
SMExportToWord1.PageSetup.RightMargin := 0.5;
```

BottomMargin property

[See also](#)

Applies to

[TSMEPageSetup](#) object

Declaration

property BottomMargin: Single;

Description

Returns or sets the distance (in specified [measure](#)) between the bottom edge of the page and the bottom boundary of the body text.

Read/write Single.

LeftMargin property

[See also](#)

Applies to

[TSMEPageSetup](#) object

Declaration

property LeftMargin: Single;

Description

Returns or sets the distance (in specified [measure](#)) between the left edge of the page and the left boundary of the body text.

Read/write Single.

Measure property

Applies to

[TSMEPageSetup](#) object

Declaration

property Measure: [TSMEMeasure](#);

Description

Returns or sets the standard measurement unit for page setup (margins, table width etc)

Orientation property

Applies to

[TSMEPageSetup](#) object

Declaration

property Orientation: [TSMEOrientation](#);

Description

Returns or sets the orientation of the page.

RightMargin property

[See also](#)

Applies to

[TSMEPageSetup](#) object

Declaration

property RightMargin: Single;

Description

Returns or sets the distance (in specified [measure](#)) between the right edge of the page and the right boundary of the body text.

Read/write Single.

TableWidth property

[See also](#)

Applies to

[TSMEPageSetup](#) object

Declaration

property TableWidth: Single;

Description

Returns or sets the table size (except margins) within page.

If not specified then all available width in page will be used.

TopMargin property

[See also](#)

Applies to

[TSMEPageSetup](#) object

Declaration

property TopMargin: Single;

Description

Returns or sets the distance (in specified [measure](#)) between the top edge of the page and the top boundary of the body text.

Read/write Single.

UseDefault property

Applies to

[TSMEPageSetup](#) object

Declaration

property UseDefault: Boolean;

Description

If property is True then custom values for margins are not applied to document and default values from page settings (in MS Windows) used.

TSMECustomDataEngine component

[See also](#) [Properties](#) [Methods](#) [Events](#)

Unit

[SMEEngine](#)

Description

The export engine generates the external file from any data source.

To specify the data source you must use any data engine which is inherited from TSMECustomDataEngine class

By default a lot of standard data engines included in deployment which allows to export data from dataset, dbgrid, string grid, listview, memo, decision cubes etc

Also included a lot of dataa engines that supports the third-party components - DecisionCube, DevExpress grids, TMSSoftware, InfoPower and more

The list of engines included in standard deployment:

Data engine class VCL component (source)

[TSMEVirtualDataEngine](#) export from code (events)

[TSMEDatasetDataEngine](#) any TDataset

[TSMEDBGridDataEngine](#) any TDBGrid

TSMEStringGridDataEngine any TStringGrid

TSMEListViewDataEngine any TListView

TSMEStringsDataEngine TStrings class (memo, listbox, combobox etc)

TSMEDecisionGridDataEngine TDecisionCube

TSMEDOADataEngine DOA (Allround Automations)

TSMEIBODataEngine IBOObjects (Jason Wharton)

TSMEdxTreeListDataEngine TdxDBTreeList (DevExpress, QT v3)

TSMEdxDBTreeListDataEngine TdxTreeList (DevExpress, QT v3)

TSMEdxDBGridDataEngine TdxDBGrid (DevExpress, QT v3)

TSMEcxCustomGridTableViewDataEngine TcxCustomGridTableView (DevExpress, QG v4 and v5)

TSMEwwDBGridDataEngine TwwDBGrid (InfoPower)

TSMEDBAdvStringGridDataEngine TDBAdvStringGridDataEngine (TMS Software)

TSMERZListViewDataEngine TRZListView (Raize)

TSMEsmDBGridDataEngine TSMDBGrid ([Scalabium Software](#))

TSMESpreadsheetDataEngine SMImport suite ([Scalabium Software](#)

)

TSMEFxCubeDataEngine TFCube

TSMEDBGridEHDataEngine TDBGridEH (ehlib.com)

TSMEVolgaDBGridDataEngine TVolgaDBGrid (volgadb.com)

Level property

Applies to

[TSMECustomDataEngine](#) component

Declaration

property Level: Integer;

Description

This readonly property returns the level of data engine in hierarchy of exported data sources (master-detail relations).

By default if no any master sources defined, the Level property returns the 0.

OnCount event

Applies to

[TSMECustomDataEngine](#) component

Declaration

property OnCount: [TSMEGetCount](#);

Description

Using this event you can return the total number of records which are available in data source.

Note that you must also include the number of rows for headers/footers/grouping etc

OnGetValue event

Applies to

[TSMECustomDataEngine](#) component

Declaration

property OnGetValue: [TSMEGetValue](#);

Description

This event called during export process for each column of every exported record
There you may return the value for every cell.

OnNext event

Applies to

[TSMECustomDataEngine](#) component

Declaration

property OnNext: [TSMEGetNext](#);

Description

In this event you can move the current position in your data source on the next record.

SelectedRecords property

[See also](#)

Applies to

[TSMECustomDataEngine](#) component

Declaration

property SelectedRecords: Boolean;

Description

This property specify if data engines will return the selected records only, not all available records.

Note that this property used only if data engine supports the feature to select the records (visual grid, for example)

DataTypeByColumn method

Applies to

[TSMECustomDataEngine](#) component

Returns the field data type for column.

Declaration

```
function DataTypeByColumn(Column: TSMEColumn): TFieldType; virtual;
```

Description

Use the DataTypeByColumn method to get the field type of column.

Eof method

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
function Eof: Boolean; virtual;
```

Description

Override this method to implement the custom algorithm for EOF (end of file)

FindFieldByColumn method

[See also](#)

Applies to

[TSMECustomDataEngine](#) component

Locates the field instance linked to column.

Declaration

```
function FindFieldByColumn(Column: TSMEColumn): TField; virtual;
```

Description

Use the FindFieldByColumn method to locate the field.

If a field corresponding to the specified column does not exist, then this method returns nil.

GetFieldValue method

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
function GetFieldValue(Column: TSMEColumn): Variant; virtual;
```

Description

Override this method to implement the custom algorithm to get the value for column in current record

IsDataRow method

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
function IsDataRow(Index: Integer): Boolean; virtual;
```

Description

Override this method to implement the custom algorithm to detect if row (specified by Index parameter) is the record with data.

For example, any "additional" rows (footers, headers, groupings etc) must return the False.

By default, all rows will return the True.

ReadOnlyByColumn method

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
function ReadOnlyByColumn(Column: TSMEColumn): Boolean; virtual;
```

Description

Use the ReadOnlyByColumn method to detect if column is readonly

RecordCount method

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
function RecordCount: Integer; virtual;
```

Description

Override this method to implement the custom algorithm to calculate the total record count

RequiredByColumn method

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
function RequiredByColumn(Column: TSMEColumn): Boolean; virtual;
```

Description

Use the RequiredByColumn method to detect if column is unique/primary key

SelectedRecordIsSupported method

[See also](#)

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
function SelectedRecordIsSupported: Boolean; virtual;
```

Description

Override this method if your custom data engine supports the multiselect and can export the selected records only.

Usually this method overridden by engines for visual grids.

By default, the parent method returns the False.

ApplyCellColors method

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
procedure ApplyCellColors(ARow, ACol: Integer; var al: TAlignment;  
var fnt: TFont; var AColor: TColor); virtual;
```

Description

You must override this method to implement some custom algorithm to apply the font/colors settings for cell (defined by ARow and ACol parameters)

For example, engine for DecisionCube reads the custom colors and fonts for values, subtotals, groupings etc

DisableControls method

[See also](#)

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
procedure DisableControls; virtual;
```

Description

You must override this method if your data engine supports the feature to disable the repainting during record exporting.

For example, disable the dbgrid repainting during dataset export.

This method executed only if soDisableControls flag included in Options property for TSMECustomBaseComponent

EnableControls method

[See also](#)

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
procedure EnableControls; virtual;
```

Description

You must override this method if your data engine supports the feature to enable the repainting after export.

For example, enable the dbgrid repainting when export finished.

This method executed only if soDisableControls flag included in Options property for TSMECustomBaseComponent

FillColumns method

[See also](#)

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
procedure FillColumns(Columns: TSMEColumns ; Bands: TSMEColumnBands  
; RightToLeft: Boolean); virtual;
```

Description

Override this method to implement the custom algorithm to create the default columns and bands to export.

For example, for export from dataset you may add all fields or for grid engine you may add all columns in grid.

First method

[See also](#)

Applies to

[TSMECustomDataEngine](#) component

Declaration

procedure First; **virtual**;

Description

In this method you may initialize the current position in data engine before export.

Next method

[See also](#)

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
procedure Next; virtual;
```

Description

In this method you may change the current position in your data engine on the next record.

RestorePosition method

[See also](#)

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
procedure RestorePosition; virtual;
```

Description

If your data engine supports the bookmarks, you may restore the previous position in list after export finished (saved in [SavePosition](#) method).

For example, you may restore the current record in dataset

SavePosition method

[See also](#)

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
procedure SavePosition; virtual;
```

Description

If your data engine supports the bookmarks, you may save the current position in list. When export process will be finished, this position will be restored in [RestorePosition](#) method.

OnAfterExecute event

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
property OnAfterExecute: TNotifyEvent;
```

Description

This event called when the export process finished. There you may destroy/release here any own internal structures/variables etc

OnBeforeExecute event

Applies to

[TSMECustomDataEngine](#) component

Declaration

property OnBeforeExecute: TNotifyEvent;

Description

This event called when the export process started. You may initialize/allocate there any own internal structures/variables etc

OnFillColumns event

Applies to

[TSMECustomDataEngine](#) component

Declaration

```
property OnFillColumns: TNotifyEvent;
```

Description

This event executed before export started if no any custom columns defined for control.
You may define the default columns with all required attributes

OnFirst event

Applies to

[TSMECustomDataEngine](#) component

Declaration

property OnFirst: TNotifyEvent;

Description

In this event you can set the current position in data source in beginning.

TSMEVirtualDataEngine component

[Methods](#)

Unit

[SMEEngine](#)

Description

This class inherited from [TSMECustomDataEngine](#) and implements the virtual/event-handled engine without any real control with data.

The full control (exported data, navigations etc) are in events.

You may generate your data on fly and return in corresponded [events](#)

TSMECustomDBDataEngine component

[Properties](#) [Methods](#)

Unit

[SMEEngDB](#)

Description

The type for basic db-aware engine.

Inherited from [TSMECustomDataEngine](#) class.

Dataset property

Applies to

[TSMECustomDBDataEngine](#) component

Declaration

property Dataset: TDataset;

Description

The Dataset property publishes the dataset which records will be exported

TSMEDatasetDataEngine component

[Properties](#) [Methods](#)

Unit

[SMEEngDB](#)

Description

This component allow to export records from standard TDataset

Any database engine supported - BDE, ADO, IBX, dbExpress, DBISAM, DOA, Halcyon etc

Dataset property

Applies to

[TSMEDatasetDataEngine](#) component

Declaration

property Dataset: TDataset;

Description

The Dataset property publishes the dataset which records will be exported

TSMEDBGridDataEngine component

[Properties](#) [Methods](#)

Unit

[SMEEngDB](#)

Description

This component allow to export records from standard TDBGrid

Any grids which are inherited from TDBGrid class - standard TDBgrid, TRxDBGrid, TSMDBGrid etc

DBGrid property

Applies to

[TSMEDBGridDataEngine](#) component

Declaration

```
property DBGrid: TCustomDBGrid;
```

Description

The DBGrid property publishes the db-aware grid with records to export.

The any dbgrid inherited from TDBGrid is supported.

TSMExportStyle object

[See also](#) [Properties](#) [Methods](#)

Unit

[ExportDS](#)

Description

This type allow to customize the color scheme for exported data.

EvenColor property

[See also](#)

Applies to

[TSMExportStyle](#) object

Declaration

```
property EvenColor: TColor;
```

Description

This property defines a default color scheme of exported data for even rows.

When you change the Style property, the EvenColor property will have a correspondence value for this scheme.

If you want to customize colors, you can assign any available color to this property.

OddColor property

[See also](#)

Applies to

[TSMExportStyle](#) object

Declaration

```
property OddColor: TColor;
```

Description

This property defines a default color scheme of exported data for odd rows.

When you change the Style property, the OddColor property will have a correspondence value for this scheme.

If you want to customize colors, you can assign any available color to this property.

Style property

[See also](#)

Applies to

[TSMExportStyle](#) object

Declaration

property Style: [TSMStyle](#);

Description

This property defines a default color scheme for exported data.

The default value is esNormal.

TSMEDataFormats object

[See also](#) [Properties](#) [Methods](#)

Unit

[ExportDS](#)

Description

This type allow to customize the presentation of exported values.

By default any data will be formatted by regional settings of current logged user in Control Panel of MS Windows but this type allow to customize an external presentation of data and define a custom decimal separator or date format.

BooleanFalse property

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

```
property BooleanFalse: string;
```

Description

This property allow to define a custom text for exported logical (False) values.

The default value is 'False' but you can define it as 'F', 'No', 'N' or any other desired text.

BooleanTrue property

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

```
property BooleanTrue: string;
```

Description

This property allow to define a custom text for exported logical (True) values.

The default value is 'True' but you can define it as 'Y', 'Yes', 'T' or any other desired text.

CurrencyString property

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

```
property CurrencyString: string;
```

Description

CurrencyString property defines the currency symbol (or characters) used in floating-point to decimal conversions.

The initial value is fetched from CurrencyString variable (SysUtils.pas unit).

CustomDateTimeFormat property

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

property CustomDateTimeFormat: **string**;

Description

You may specify any custom format string for date/time values.

For example, 'MMMM DD, YYYY time: HH:SS' which can't be constructed automatically from other properties ([DateOrder](#))

DateOrder property

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

property DateOrder: [TSMEDateOrder](#);

Description

This property allow to define a custom order of exported date/time values.

DateSeparator property

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

property DateSeparator: Char;

Description

This property allow to define a custom date separator for exported date/time values.

The default value is a value from regional settings in Control Panel of MS Windows.

DecimalSeparator property

[See also](#)

Applies to

[TSMEDDataFormats](#) object

Declaration

property DecimalSeparator: Char;

Description

DecimalSeparator property allow to define a custom numeric separator for exported numeric and currency values.

The default value is a value from regional settings in Control Panel of MS Windows (DecimalSeparator variable variable (SysUtils.pas unit).

FourDigitYear property

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

property FourDigitYear: Boolean;

Description

This property allow to define a custom year presentation for exported date/time values.

The default value is a value from regional settings in Control Panel of MS Windows. For example, if short date format is 'mm/dd/yyyy' then FourDigitYear is True.

LeadingZerosInDate property

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

property LeadingZerosInDate: Boolean;

Description

This property allow to define a custom date presentation for exported date/time values.

The default value is a value from regional settings in Control Panel of MS Windows. For example, if short date format is 'm/d/yy' then LeadingZerosInDate is False but if short format is 'mm/dd/yy' then LeadingZerosInDate is True.

ThousandSeparator property

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

property ThousandSeparator: Char;

Description

ThousandSeparator property is the character used to separate thousands in numbers with more than three digits to the left of the decimal separator.

The initial value is fetched from ThousandSeparator variable (SysUtils.pas unit).

TimeSeparator property

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

property TimeSeparator: Char;

Description

This property allow to define a custom time separator for exported date/time values.

The default value is a value from regional settings in Control Panel of MS Windows.

UseRegionalSettings property

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

property UseRegionalSettings: Boolean;

Description

Specifies whether format settings are updated automatically before export process started.

If property value is False, then values stored in dfm-file (or applied in run-time code) used.

GetDateFormat method

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

```
function GetDateFormat: string;
```

Description

This method returns the format string generated for date values.

GetDateTimeFormat method

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

```
function GetDateTimeFormat: string;
```

Description

This method returns the format string generated for date/time values.

GetTimeFormat method

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

```
function GetTimeFormat: string;
```

Description

This method returns the format string generated for time values.

LoadRegionalSettings method

[See also](#)

Applies to

[TSMEDataFormats](#) object

Declaration

procedure LoadRegionalSettings;

Description

LoadRegionalSettings method allow to load the current regional settings in run-time

TSMESendTo object

[See also](#) [Properties](#) [Methods](#)

Unit

[ExportDS](#)

Description

This type allow to customize the settings for emailing (generate the exported file and send by email).

The Simple MAPI protocol used for emailing and default mail client (Outlook, Outlook Express, Eudora, The Bath etc) must be correctly configured in MS Windows.

EEmailBCC property

[See also](#)

Applies to

[TSMESendTo](#) object

Declaration

property EMailBCC: **string**;

Description

You may specify the list of BCC:-recipients

SMExportToPDF1.SendTo.EmailBCC := 'admin@domain.com;boss@domain.com';

Please note that some mail-servers requires the 'smtp:' prefix for correct name resolving:

SMExportToPDF1.SendTo.EmailBCC := 'smtp:admin@domain.com';

EmailBody property

[See also](#)

Applies to

[TSMESendTo](#) object

Declaration

property EmailBody: **string**;

Description

You may customize the body for generated message

```
SMExportToPDF1.SendTo.EmailBody := 'Hello friends,' + #13#10 + 'I attach the requested document' + #13#10 + 'Peter';
```


EMailCC property

[See also](#)

Applies to

[TSMESendTo](#) object

Declaration

property EMailCC: **string**;

Description

You may specify the list of CC:-recipients

SMExportToPDF1.SendTo.EmailCC := 'peter@domain.com;mary@domain.com';

Please note that some mail-servers requires the 'smtp:' prefix for correct name resolving:

SMExportToPDF1.SendTo.EmailCC := 'smtp:peter@domain.com';

EmailOpenBeforeSend property

Applies to

[TSMESendTo](#) object

Declaration

property EmailOpenBeforeSend: Boolean;

Description

Set this property to True if you want to display the standard message dialog before send where your end-user can change the subject or body etc

Set to False if you want to send the message in "hidden" mode

EmailRecipient property

[See also](#)

Applies to

[TSMESendTo](#) object

Declaration

property EmailRecipient: **string**;

Description

You may specify the list of recipients for your message.

```
SMExportToPDF1.SendTo.EmailRecipient := 'user1@domain.com;user2@domain.com';
```

Please note that some mail-servers requires the 'smtp:' prefix for correct name resolving:

```
SMExportToPDF1.SendTo.EmailRecipient := 'smtp:user1@domain.com';
```

EmailSubject property

[See also](#)

Applies to

[TSMESendTo](#) object

Declaration

property EmailSubject: **string**;

Description

You may customize the subject for generated message

```
SMExportToPDF1.SendTo.EmailSubject := 'account balance for current year';
```

TSMEDataLevel component

[See also](#) [Properties](#) [Methods](#)

Unit

[ExportDS](#)

Description

<<< Description of TSMEDataLevel component >>>

Bands property

[See also](#) [Example](#)

Applies to

[TSMEDataLevel](#) component

Declaration

property Bands: TSMEColumnBands;

Description

<<< Description of Bands property >>>

Caption property

[See also](#) [Example](#)

Applies to

[TSMEDataLevel](#) component

Declaration

property Caption: **string**;

Description

<<< Description of Caption property >>>

Columns property

[See also](#) [Example](#)

Applies to

[TSMEDataLevel](#) component

Declaration

property Columns: TSMEColumns;

Description

<<< Description of Columns property >>>

ColumnSource property

[See also](#) [Example](#)

Applies to

[TSMEDataLevel](#) component

Declaration

property ColumnSource: TColumnSource;

Description

<<< Description of ColumnSource property >>>

DataEngine property

[See also](#) [Example](#)

Applies to

[TSMEDataLevel](#) component

Declaration

property DataEngine: TSMECustomDataEngine;

Description

<<< Description of DataEngine property >>>

DataSet property

[See also](#) [Example](#)

Applies to

[TSMEDataLevel](#) component

Declaration

property DataSet: TDataSet;

Description

<<< Description of DataSet property >>>

DBGrid property

[See also](#) [Example](#)

Applies to

[TSMEDataLevel](#) component

Declaration

```
property DBGrid: TCustomControl;
```

Description

<<< Description of DBGrid property >>>

ShiftCount property

[See also](#) [Example](#)

Applies to

[TSMEDataLevel](#) component

Declaration

```
property ShiftCount: Integer;
```

Description

<<< Description of ShiftCount property >>>

CreateSourceDataEngine method

[See also](#) [Example](#)

Applies to

[TSMEDataLevel](#) component

Declaration

```
procedure CreateSourceDataEngine;
```

Description

<<< Description of CreateSourceDataEngine method >>>

DestroySourceDataEngine method

[See also](#) [Example](#)

Applies to

[TSMEDataLevel](#) component

Declaration

```
procedure DestroySourceDataEngine;
```

Description

<<< Description of DestroySourceDataEngine method >>>

TSMEDataLevels component

[See also](#) [Properties](#) [Methods](#) [Tasks](#)

Unit

[ExportDS](#)

Description

<<< Description of TSMEDataLevels component >>>

Items property

[See also](#) [Example](#)

Applies to

[TSMEDataLevels](#) component

Declaration

property Items: TSMEDataLevel;

Description

<<< Description of Items property >>>

Run-time only

SMEControl property

[See also](#) [Example](#)

Applies to

[TSMEDataLevels](#) component

Declaration

property SMEControl: TSMECustomBaseComponent;

Description

<<< Description of SMEControl property >>>

Run-time only

Read-only

Create method

[See also](#) [Example](#)

Applies to

[TSMEDataLevels](#) component

Declaration

constructor Create(SMECtrl: TSMECustomBaseComponent);

Description

<<< Description of Create method >>>

Add method

[See also](#) [Example](#)

Applies to

[TSMEDataLevels](#) component

Declaration

```
function Add: TSMEDataLevel;
```

Description

<<< Description of Add method >>>

TSMEStatistic object

[See also](#) [Properties](#) [Methods](#)

Unit

[ExportDS](#)

Description

<<< Description of TSMEStatistic object >>>

CurrentCol property

[See also](#) [Example](#)

Applies to

[TSMStatistic](#) object

Declaration

```
property CurrentCol: LongInt;
```

Description

<<< Description of CurrentCol property >>>

Run-time only

CurrentDataLevel property

[See also](#) [Example](#)

Applies to

[TSMEStatistic](#) object

Declaration

property CurrentDataLevel: Integer;

Description

<<< Description of CurrentDataLevel property >>>

Run-time only

CurrentRow property

[See also](#) [Example](#)

Applies to

[TSMEStatistic](#) object

Declaration

property CurrentRow: LongInt;

Description

<<< Description of CurrentRow property >>>

Run-time only

CurrentSection property

[See also](#) [Example](#)

Applies to

[TSMStatistic](#) object

Declaration

property CurrentSection: TSMSection;

Description

<<< Description of CurrentSection property >>>

Run-time only

Result property

[See also](#) [Example](#)

Applies to

[TSMStatistic](#) object

Declaration

property Result: TSMEResult;

Description

<<< Description of Result property >>>

TotalCount property

[See also](#) [Example](#)

Applies to

[TSMEStatistic](#) object

Declaration

```
property TotalCount: LongInt;
```

Description

<<< Description of TotalCount property >>>

TotalErrors property

[See also](#) [Example](#)

Applies to

[TSMEStatistic](#) object

Declaration

property TotalErrors: LongInt;

Description

<<< Description of TotalErrors property >>>

UpdateStep property

[See also](#) [Example](#)

Applies to

[TSMEStatistic](#) object

Declaration

```
property UpdateStep: LongInt;
```

Description

<<< Description of UpdateStep property >>>

TSMECustomBaseComponent component

[See also](#) [Properties](#) [Methods](#) [Events](#)

Unit

[ExportDS](#)

Description

TSMExportBaseComponent is the basic export component. All other export components inherited from this basic class

About property

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property About: [TSMEAbout](#);

Description

This is design-time property - in Object Inspector you can read a basic information about SMExport suite.

AddTitle property

Applies to

[TSMECustomBaseComponent](#) component

AddTitle indicates whether the field titles is exporting in output file.

Declaration

property AddTitle: Boolean;

Description

Set AddTitle to True if you want to export a captions for each dataset column before data.
Set AddTitle to False if you want to export a data without field titles.

The AddTitle must be set before export starting.

AnimatedStatus property

Applies to

[TSMECustomBaseComponent](#) component

AnimatedStatus indicates whether the status dialog is displaying during export process.

Declaration

property AnimatedStatus: Boolean;

Description

Set AnimatedStatus to True if you want to show an animated status dialog during the export process.

Set AnimatedStatus to False if you want to export a data without animated status dialog.

AutoFitColumns property

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property AutoFitColumns: Boolean;

Description

<<< Description of AutoFitColumns property >>>

Run-time only

BlankIfZero property

Applies to

[TSMECustomBaseComponent](#) component

BlankIfZero indicates whether the empty values is exported as zeros.

Declaration

```
property BlankIfZero: Boolean;
```

Description

Set BlankIfZero to True if you want to export the empty value (NULL) for zero values.
Set BlankIfZero to False if you want to export the empty values as NULL.

CharacterSet property

Applies to

[TSMECustomBaseComponent](#) component

You can define the fished code page for data in result file.

Declaration

property CharacterSet: [TCharacterSet](#);

Description

You may select the destination code page for file.

csANSI_WINDOWS save in ANSI

csASCII_MSDOS save OEM

csEBCDIC save file for IBM mainframe

DataFormats property

[Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Specifies the custom formats which you want to apply to exported data.

Declaration

property DataFormats: [TSMEDataFormats](#) ;

Description

Use this property you can define a desired formats for data of any data type.
For example you can customize a date separator or True/False meanings.

Note that these colors will be affected as default but you can customize any exported value in the OnGetCellParams event.

ExportedRecordCount property

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property ExportedRecordCount: Integer;

Description

This is readonly property which returns a total number of exported records. You can read a correct value in run-time after export finishing only.

ExportIfEmpty property

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

```
property ExportIfEmpty: Boolean;
```

Description

<<< Description of ExportIfEmpty property >>>

ExportResult property

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property ExportResult: TSMEResult;

Description

<<< Description of ExportResult property >>>

Run-time only

Read-only

ExportStyle property

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property ExportStyle: TSMExportStyle;

Description

<<< Description of ExportStyle property >>>

Run-time only

Fixed property

[See also](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property Fixed: Boolean;

Description

This property allows to switch a mode of export type into text file.

When Fixed is True, the target text file is "Fixed Width": fields are aligned in columns with separators between each field.

When Fixed is False, the target text is CSV: characters such as comma or tab (separator symbols) separate each field.

KeyGenerator property

[See also](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property KeyGenerator: **string**;

Description

This property allows to define the value which will be used as value for meta-value of export process.

For example, this value will be used in TSMExportToHTML component as meta name of generator in HTML-header or in TSMExportToExcel component as name of spreadsheet.

You can change it for each export component but the default value stores in GeneratorVer constant.

OnGetFileName property

Applies to

[TSMECustomBaseComponent](#) component

OnGetFileName occurs if you defined RowsPerFile property and SMExport engine must create a new file for next batch.

Declaration

property OnGetFileName: [TSMEGetFileName](#) ;

Description

Use the OnGetFileName event handler to define a custom file names for each batch with exported records.

By default the file names will be generated as FileName001.ext, FileName002.ext etc where FileName.ext is a value of your FileName property.

OnProgress property

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property OnProgress: TSMEProgressadd;

Description

<<< Description of OnProgress property >>>

Options property

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property Options: TSMOptions;

Description

<<< Description of Options property >>>

PageSetup property

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property PageSetup: TSMEPageSetup;

Description

<<< Description of PageSetup property >>>

Run-time only

RecordSeparator property

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property RecordSeparator: **string**;

Description

<<< Description of RecordSeparator property >>>

Run-time only

RightToLeft property

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property RightToLeft: Boolean;

Description

This property can be used for middle-east users. When RightToLeft is True, then all columns will be exported in opposite direction (from right side to left).

This is useful for jewish or arabic languages.

SelectedRecord property

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property SelectedRecord: Boolean;

Description

Using this property you can export the selected records from linked DBGrid.

PS: this property is used when:

1. [ColumnSource](#) = csDBGrid
2. DBGrid component inherits from TCustomDBGrid only

If you use some third-party grid or you want to control a list of exported records in own code, you can use [OnGetSelectedCount](#) and [OnGetNextSelected](#) events.

SendTo property

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property SendTo: [TSMESendTo](#);

Description

<<< Description of SendTo property >>>

Separator property

[See also](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property Separator: Char;

Description

Changing the value of this property you can define the separator of fields in target text file.

The popular values are Tab (#9), Semicolon (;), Comma (,), Space (#32) but you can define the any other character too.

Statistic property

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

```
property Statistic: TSMEStatistic;
```

Description

<<< Description of Statistic property >>>

TableName property

[See also](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property TableName: **string**;

Description

This property is used in two export components:

TSMEExportToAccess name of table in database in which will export any data

TSMEExportToSQL name of table which will be used for generation of INSERT INTO-statement

TextQualifier property

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property TextQualifier: Char;

Description

<<< Description of TextQualifier property >>>

Run-time only

Extension method

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

```
function Extension: string; virtual;
```

Description

<<< Description of Extension method >>>

GetTitleRowCount method

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

```
function GetTitleRowCount: Integer;
```

Description

<<< Description of GetTitleRowCount method >>>

SendMail method

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

```
function SendMail(const AFileName, body: string): Integer;
```

Description

<<< Description of SendMail method >>>

AboutSME method

Applies to

[TSMECustomBaseComponent](#) component

Declaration

```
procedure AboutSME;
```

Description

To call this method if you want to see a dialog with short information about SMExport suite.

OnAfterExecute event

[See also](#)

Applies to

[TSMECustomBaseComponent](#) component

OnAfterExecute occurs after an application completes exporting a data.

Declaration

```
property OnAfterExecute: TNotifyEvent;
```

Description

Write an OnAfterExecute event handler to take specific action immediately after an application exports the data. OnAfterExecute is immediately after finishing of export process. For example, an OnAfterExecute event handler might write a message to the log system or show a total time which was spent on export process.

OnAfterRecord event

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property OnAfterRecord: TSMEAfterRecordEvent;

Description

<<< Description of OnAfterRecord event >>>

OnBeforeExecute event

[See also](#)

Applies to

[TSMECustomBaseComponent](#) component

OnBeforeExecute occurs before an application executes a request to export a data.

Declaration

property OnBeforeExecute: TNotifyEvent;

Description

Write a OnBeforeExecute event handler to take specific action before an application exports a data. OnBeforeExecute is triggered when an application calls Execute method.

OnBeforeRecord event

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property OnBeforeRecord: TSMEBeforeRecordEvent;

Description

<<< Description of OnBeforeRecord event >>>

OnGetCellParams event

[See also](#) [Example](#)

Applies to

[TSMECustomBaseComponent](#) component

OnGetCellParams occurs for each exported cell (for each column of each record in source of data).

Declaration

property OnGetCellParams: [TGetCellParamsEvent](#) ;

Description

Use the OnGetCellParams event handler to write a code that allow to change an exported data (text, font, alignment or background color).

OnGetLanguageString event

[Example](#)

Applies to

[TSMECustomBaseComponent](#) component

Declaration

property OnGetLanguageString: [TSMEGetLanguageStringEvent](#);

Description

This event must be used for development of multilingual applications. Each string resource have the own integer identifier - so in this event you can return a desired text for any ID.

TSMExportBaseComponent component

[See also](#) [Properties](#) [Methods](#) [Events](#)

Unit

[ExportDS](#)

Description

<<< Description of TSMExportBaseComponent component >>>

ActionAfterExport property

Applies to

[TSMExportBaseComponent](#) component

Declaration

property ActionAfterExport: [TActionAfterExport](#);

Description

You can define the default action which will be processed after export complete.

For example, to open the target export file in view/edit mode or send this exported file by e-mail as attachment.

Bands property

[See also](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

property Bands: [TSMColumnBands](#);

Description

<<< Description of Bands property >>>

Columns property

[See also](#)

Applies to

[TSMExportBaseComponent](#) component

Columns describes the display attributes and field bindings of the columns in the [TSMExportBaseComponent](#) object.

Declaration

property Columns: [TSMEColumns](#) ;

Description

Use Columns to read or set the field bindings and display attributes of the columns in the exported file. Columns is an indexed collection of TSMEColumn objects. Use the properties of the TSMEColumn objects to specify the display attributes or field bindings of individual columns in the exported file.

Columns can be set at design time through the Columns editor, or programmatically at runtime.

ColumnSource property

Applies to

[TSMExportBaseComponent](#) component

This property specifies the source of data for exporting.

Declaration

property ColumnSource: [TColumnSource](#) ;

Description

To set the ColumnSource to csDBGrid if you want to export a data from linked [DBGrid](#) component. All grid attributes (width, font, alignment, titles etc) for each column will be copied to export component.

To set the ColumnSource to csDataSet if you want to export a data from linked [DataSet](#) component.

To set the ColumnSource to csDataEngine if you want to export from custom or third-party [DataEngine](#) component.

DataEngine property

[See also](#)

Applies to

[TSMEExportBaseComponent](#) component

Declaration

property DataEngine: [TSMECustomDataEngine](#);

Description

<<< Description of DataEngine property >>>

DataSet property

[See also](#)

Applies to

[TSMExportBaseComponent](#) component

Specifies the dataset object component used as source to export a data.

Declaration

```
property DataSet: TDataSet;
```

Description

Use the DataSet property to specify a source for data exporting. If you defined ColumnSource as csDataset, the DataSet property is required and you must define it with available dataset component. Else if you exports a data from DBGrid (ColumnSource = csDBGrid), the DataSet property is not required and can be omitted.

You can specify the any TDataSet successor.

For example, the BDE's TTable, TQuery, the multi-tier TClientDataSet, ADO's TADODataset, TADOTable, TADOQuery, IBX's TIBTable, TIBQuery or third-party dataset components (Titan for Btrieve, MemoryTable, OracleDataset etc).

DBGrid property

[See also](#)

Applies to

[TSMExportBaseComponent](#) component

Specifies the dbgrid object component used as source to export a data.

Declaration

```
property DBGrid: TDBGrid;
```

Description

Use the DBGrid property to specify a source for data exporting. If you defined ColumnSource as csDBGrid, the DBGrid property is required and you must define it with available grid component. Else if you exports a data from Dataset (ColumnSource = csDataset), the DBGrid property is not required and can be omitted.

You can specify the any TCustomGrid successor which have the next properties:

1. DataSource: TDataSource;
- 2.

property Columns: TCollection and each item have the property FieldName: string;
or

```
property Fields[i: Integer]: TField;
```

For example, you can select the standard TDBGrid or or third-party DBGrid components (the RXLib's TRxDBGrid, the SM's TSMDBGrid, InfoPower's TwwDBGrid etc).

DetailSources property

[See also](#) [Example](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

property DetailSources: TSMDataLevels;

Description

<<< Description of DetailSources property >>>

Run-time only

FileName property

Applies to

[TSMExportBaseComponent](#) component

Declaration

property FileName: **string**;

Description

This property defines the file name of target file with exported data.

LastExportColumnIndex property

Applies to

[TSMEExportBaseComponent](#) component

Declaration

property LastExportColumnIndex: Integer;

Description

This run-time property will return the total number of visible columns

OnGetNextSelected property

[See also](#) [Example](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

property OnGetNextSelected: [TSMGetNextSelected](#) ;

Description

If you use some third-party grid or you want to control a list of exported records in own code, you must use OnGetSelectedCount and OnGetNextSelected events.

In this event you must return the pointer to the next selected record which must be exported.

OnGetSelectedCount property

[See also](#) [Example](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

property OnGetSelectedCount: [TSMEGetSelectedCount](#) ;

Description

If you use some third-party grid or you want to control a list of exported records in own code, you must use OnGetSelectedCount and OnGetNextSelected events.

In this event you must return a total number of records which must be exported in target file.

SpecificationDir property

[See also Example](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

property SpecificationDir: **string**;

Description

You may specify the folder where SMLExport will search the available specifications.

By default, specifications located in application directory but you may define any other directory.

LoadSpecification method

[See also](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

```
class procedure LoadSpecification(strFileName: string);
```

Description

Using this method you can load the saved specification from external file which contain all settings for export.

The parameter is the file name with full path and drive.

GetDefExt method

[See also](#) [Example](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

```
function GetDefExt(intIndex: Integer): string;
```

Description

\$IFDEF CLR

GetDS method

[See also](#) [Example](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

```
function GetDS: TDataSet;
```

Description

```
procedure FillColumns;
```

BuildDefaultColumns method

Applies to

[TSMEExportBaseComponent](#) component

Declaration

```
procedure BuildDefaultColumns;
```

Description

If no any custom columns are not defined in [Columns](#) property, then SMExport will create the columns by default from dbgrid or dataset or data engine (depends from [ColumnSource](#) property).

Execute method

[See also](#)

Applies to

[TSMExportBaseComponent](#) component

Execute performs the export operation specified by properties.

Declaration

```
procedure Execute;
```

Description

After setting properties to indicate what export operation should be performed and how, call Execute to perform the operation. As a minimum, the ColumnSource, DBGrid and/or Dataset properties must be defined.

After calling Execute, the file with exported data will be created. Also the result of operation will give an indication of what happened as a result of the call to Execute.

LoadSpecificationFromStream method

[See also](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

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

Description

Using this method you can load the saved specification from any your stream (BLOB-field, for example).

SaveSpecification method

[See also](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

```
procedure SaveSpecification(SpecName, FileName: string; ShowDialog:  
Boolean);
```

Description

Using this method you can save a specification into external file which will contain all defined export settings.

The first parameter is description of specification (will be displayed in TSMEWizardDlg for end-user), the second parameter is the file name with full path.

If parameters are empty, there will be displayed a dialog where user can define parameters.

SaveSpecificationToStream method

[See also](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

```
procedure SaveSpecificationToStream(Stream: TStream; const
SpecName: string);
```

Description

Using this method you can save a specification into any stream (BLOB-field, for example). The first parameter is the destination stream, the second is the description of specification (will be displayed in TSMEWizardDlg for end-user).

OnAfterLoadSpecification event

[See also](#) [Example](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

property OnAfterLoadSpecification: TSMESpecificationEventadd;

Description

<<< Description of OnAfterLoadSpecification event >>>

OnAfterSaveSpecification event

[See also](#) [Example](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

property OnAfterSaveSpecification: TSMESpecificationEventadd;

Description

<<< Description of OnAfterSaveSpecification event >>>

OnBeforeLoadSpecification event

[See also](#) [Example](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

property OnBeforeLoadSpecification: TSMESpecificationEventadd;

Description

<<< Description of OnBeforeLoadSpecification event >>>

OnBeforeSaveSpecification event

[See also](#) [Example](#)

Applies to

[TSMExportBaseComponent](#) component

Declaration

property OnBeforeSaveSpecification: TSMESpecificationEventadd;

Description

<<< Description of OnBeforeSaveSpecification event >>>

Registration By credit card

Visa/Discover/MasterCard/AmericanExpress 50 EUR for full package with sources

For technical support or comments about this program, you may contact Mike Shkolnik at: <mailto:mshkolnik@scalabium.com>

For your convenience we have contracted another companies (registrators), PayPro to process any orders you may wish to place with your PayPal, Visa, Discover, MasterCard or other credit cards.

Registrators can be easily contacted **for orders only** via any of the following methods:

ONLINE ORDERS

You may register SMExport suite via an online order form by pointing your browser to:

1. PayPro: <https://secure.payproglobal.com/orderpage.aspx?products=51379>

Registration By mail

Select **Print Topic...** from the **File** menu to print this form.

Item:

SMExport suite for Delphi/C++Builder

Price: 50EUR with sources or 35EUR without sources

Please register my copy of SMExport suite for Delphi/C++Builder,
I am sending a check or money order for \$_____

Name: _____

Company: _____

Address1: _____

Address2: _____

City: _____

State: _____ **Zip:** _____

Country: _____

Phone: _____ optional

Email: _____

Please send completed form with payment to:

ul.Pragskaya 4, kv.39

Kiev, 02090

Ukraine UA

Properties

► Run-time only ■ Key properties

■ [Alignment](#)

■ [Caption](#)

■ [Color](#)

■ [Font](#)

■ [Visible](#)

Methods

■ Key methods

~~Create~~{linkDelphi=Create_Method}

~~Destroy~~{linkDelphi=Destroy_Method}

~~Assign~~{linkDelphi=Assign_Method}

SMEEngine unit

[See also](#)

This is the unit where a few basic types, components and procedures are declared.

Components

[TSMEColumnBand](#)

[TSMEColumnBands](#)

[TSMEColumn](#)

[TSMEColumns](#)

[TSMECustomDataEngine](#)

[TSMEVirtualDataEngine](#)

Objects

[TSMEColumnTitle](#)

[TSMEPageSetup](#)

Types

[TCellType](#)

[TSMEColumnKind](#)

[TSMEFieldDisplayFormat](#)

[TSMEGetCount](#)

[TSMEGetNext](#)

[TSMEGetValue](#)

[TSMEMeasure](#)

[TSMEOrientation](#)

Routines

[CellType2FieldType](#)

[GetValueType](#)

Properties

▶ Run-time only ■ Key properties

▶ ■ [Items](#)

Methods

- Key methods
- [Add](#)

Add method example

```
with SMExportToCSV1.Bands.Add do
begin
Caption := 'General information';
Alignment := taCenter;
Visible := True;
Font.Style := [fsBold, fsUnderline];
Font.Color := clRed;
Color := clYellow;
end;
```

Properties

► Run-time only ■ Key properties

■ [Alignment](#)

■ [Caption](#)

■ [Color](#)

■ [Font](#)

Methods

- Key methods

~~Destroy~~{linkDelphi=Destroy_Method}

~~Assign~~{linkDelphi=Assign_Method}

See also

[Caption](#)

See also

[TSMEColumn.FieldName](#)

[TSMEColumn.Title](#)

See also

[TSMEColumn.Color](#)

See also

[TSMEColumn.Font](#)

See also

[TSMEColumnBand](#)

[TSMEColumnBands](#)

[TSMEColumnTitle](#)

Properties

► Run-time only ■ Key properties

- [Alignment](#)
- [BandIndex](#)
- [Color](#)
- [ColumnKind](#)
- [DataType](#)
- [FieldName](#)
- [Font](#)
- [Precision](#)
- [Title](#)
- [Visible](#)
- [Width](#)

Methods

- Key methods

~~Create~~{linkDelphi=Create_Method}

~~Destroy~~{linkDelphi=Destroy_Method}

- [GetItemCaption](#)

~~Assign~~{linkDelphi=Assign_Method}

See also

[TSMEColumnTitle.Alignment](#)

See also

[TSMEColumnBand](#)

[TSMEColumnBands](#)

BandIndex property example

```
with SMExportToPDF1.Bands.Add do
begin
Caption := 'General information';
Visible := True;
Font.Style := [fsBold];
Font.Color := clRed;
Color := clWindow;
end;
with SMExportToPDF1.Bands.Add do
begin
Caption := 'Details';
Visible := True;
Font.Style := [fsBold];
Font.Color := clBlue;
Color := clWindow;
end;
```

{link columns to bands}

```
SMExportToPDF1.Columns[0].BandIndex := 0;
SMExportToPDF1.Columns[1].BandIndex := 0;
SMExportToPDF1.Columns[2].BandIndex := 1;
SMExportToPDF1.Columns[3].BandIndex := 1;
SMExportToPDF1.Columns[4].BandIndex := -1;
```

See also

[TSMEColumnTitle.Color](#)

TSMEColumnKind type

[See also](#)

Unit

[SMEEngine](#)

Declaration

```
type TSMEColumnKind = (ckField, ckConstant, ckSysVar);
```

Description

This enumerated type is the available kinds for data in exported columns.

TCellType type

[See also](#)

Unit

[SMEEngine](#)

Declaration

```
type TCellType = (ctBlank, ctInteger, ctDouble, ctString,  
ctDateTime, ctDate, ctCurrency, ctTime, ctMEMO, ctGraphic,  
ctBoolean);
```

Description

This enumerated type is internal type which described the type of column. Used by any "spreadsheet" component.

See also

[TSMEColumn.Title](#)

[TSMEColumnTitle.Caption](#)

See also

[TSMEColumnTitle.Font](#)

TSMEFieldDisplayFormat type

[See also](#)

Unit

[SMEEngine](#)

Declaration

```
type TSMEFieldDisplayFormat = (dfFull, dfFieldCaption,  
dfFieldName);
```

Description

This enumerated type used to list all available types to view the column list in export wizard

dfFull will be displayed the caption and field name for every column

dfFieldCaption will be displayed the caption only

dfFieldName will be displayed the field name only

See also

[TSMEColumnBand](#)

[TSMEColumnBands](#)

[TSMEColumn](#)

[TSMEColumnTitle](#)

Properties

▶ Run-time only ■ Key properties

▶ ■ [Items](#)

Methods

- Key methods
- [Add](#)
- [ColumnByFieldName](#)
- [MergedColCount](#)
- [MergedColStart](#)

See also

[TSMEMeasure](#)

[TSMEOrientation](#)

Properties

► Run-time only ■ Key properties

■ [BottomMargin](#)

■ [LeftMargin](#)

■ [Measure](#)

■ [Orientation](#)

■ [RightMargin](#)

■ [TableWidth](#)

■ [TopMargin](#)

■ [UseDefault](#)

Methods

- Key methods

~~Assign~~{linkDelphi=Assign_Method}

See also

- [LeftMargin](#)
- [RightMargin](#)
- [TopMargin](#)

See also

- [BottomMargin](#)
- [RightMargin](#)
- [TopMargin](#)

TSMEMeasure type

[See also](#)

Unit

[SMEEngine](#)

Declaration

```
type TSMEMeasure = (emPoint, emInch, emCentimeters, emPicas);
```

Description

This enumerated type used to list all available measures to calculate the coordinates for page size and margins

TSMEOrientation type

[See also](#)

Unit

[SMEEngine](#)

Declaration

```
type TSMEOrientation = (emDefault, emPortrait, emLandscape);
```

Description

This enumerated type used to list all available orientations for page size.

See also

- [BottomMargin](#)
- [LeftMargin](#)
- [TopMargin](#)

See also

- [BottomMargin](#)
- [LeftMargin](#)
- [RightMargin](#)
- [TopMargin](#)

See also

- [BottomMargin](#)
- [LeftMargin](#)
- [RightMargin](#)

See also

<<< See also of TSMECustomDataEngine component >>>

Properties

▶ Run-time only ■ Key properties

▶ ■ [Level](#)

▶ ■ [SelectedRecords](#)

Methods

- Key methods
 - [DataTypeByColumn](#)
 - [FindFieldByColumn](#)
 - [IsDataRow](#)
 - [ReadOnlyByColumn](#)
 - [RequiredByColumn](#)
 - [SelectedRecordIsSupported](#)
 - [ApplyCellColors](#)
- [FillColumns](#)
- [Eof](#)
- [RecordCount](#)
- [GetFieldValue](#)
- [First](#)
- [Next](#)
- [DisableControls](#)
- [EnableControls](#)
- [RestorePosition](#)
- [SavePosition](#)

Events

- Key events
- [OnAfterExecute](#)
- [OnBeforeExecute](#)
- [OnCount](#)
- [OnFillColumns](#)
- [OnFirst](#)
- [OnGetValue](#)
- [OnNext](#)

TSMEGetCount type

[See also](#)

Unit

[SMEEngine](#)

Declaration

```
type TSMEGetCount = procedure(Sender: TObject; var Count: LongInt);  
of object;
```

Description

This procedure type defined for [OnCount](#) event to return the total row count for data exporting.

TSMEGetValue type

[See also](#)

Unit

[SMEEngine](#)

Declaration

```
type TSMEGetValue = procedure(Sender: TObject; Column: TSMEColumn;  
var Value: Variant); of object;
```

Description

This procedure type defined for [OnGetValue](#) event to return the value for column from current record

TSMEGetNext type

[See also](#)

Unit

[SMEEngine](#)

Declaration

```
type TSMEGetNext = procedure(Sender: TObject; var Abort: Boolean);  
of object;
```

Description

This procedure type defined for [OnNext](#) event to navigate thru records in data engine.

See also

- [SelectedRecordIsSupported](#)

See also

[TSMEColumns.ColumnByFieldName](#)

See also

►  [SelectedRecords](#)

See also

- [EnableControls](#)

See also

- [DisableControls](#)

See also

- [OnFillColumns](#)

See also

- [OnFirst](#)

See also

- [OnNext](#)

See also

- [SavePosition](#)

See also

- [RestorePosition](#)

Methods

- Key methods

~~Eof~~{linkDelphi=Eof_Method}

~~GetFieldValue~~{linkDelphi=GetFieldValue_Method}

~~First~~{linkDelphi=First_Method}

~~Next~~{linkDelphi=Next_Method}

Properties

▶ Run-time only ■ Key properties

▶ ■ [Dataset](#)

Methods

■ Key methods

~~DataTypeByColumn~~{linkDelphi=DataTypeByColumn_Method}
~~Eof~~{linkDelphi=Eof_Method}
~~FindFieldByColumn~~{linkDelphi=FindFieldByColumn_Method}
~~GetFieldValue~~{linkDelphi=GetFieldValue_Method}
~~ReadOnlyByColumn~~{linkDelphi=ReadOnlyByColumn_Method}
~~RecordCount~~{linkDelphi=RecordCount_Method}
~~RequiredByColumn~~{linkDelphi=RequiredByColumn_Method}
~~DisableControls~~{linkDelphi=DisableControls_Method}
~~EnableControls~~{linkDelphi=EnableControls_Method}
~~FillColumns~~{linkDelphi=FillColumns_Method}
~~First~~{linkDelphi=First_Method}
~~Next~~{linkDelphi=Next_Method}
~~RestorePosition~~{linkDelphi=RestorePosition_Method}
~~SavePosition~~{linkDelphi=SavePosition_Method}

SMEEngDB unit

In this unit a few standard db-aware engines declared.

Components

[TSMECustomDBDataEngine](#)

[TSMEDatasetDataEngine](#)

[TSMEDBGridDataEngine](#)

Properties

▸ Run-time only ▣ Key properties

▣ [Dataset](#)

Methods

■ Key methods

~~Eof~~{linkDelphi=Eof_Method}

~~RecordCount~~{linkDelphi=RecordCount_Method}

~~SelectedRecordsSupported~~{linkDelphi=SelectedRecordsSupported_Method}

~~First~~{linkDelphi=First_Method}

~~Next~~{linkDelphi=Next_Method}

Properties

▸ Run-time only ▣ Key properties

▣ [DBGrid](#)

Methods

■ Key methods

~~Eof~~{linkDelphi=Eof_Method}

~~RecordCount~~{linkDelphi=RecordCount_Method}

~~SelectedRecordsSupported~~{linkDelphi=SelectedRecordsSupported_Method}

~~FillColumns~~{linkDelphi=FillColumns_Method}

~~First~~{linkDelphi=First_Method}

~~Next~~{linkDelphi=Next_Method}

See also

<<< See also of TSMExportStyle object >>>

Properties

► Run-time only ■ Key properties

■ [EvenColor](#)

■ [OddColor](#)

■ [Style](#)

Methods

- Key methods

~~Assign~~{linkDelphi=Assign_Method}

ExportDS unit

This unit contains the declaration for basic export component and some related types.

Components

[TSMEDataLevel](#)

[TSMEDataLevels](#)

[TSMECustomBaseComponent](#)

[TSMEExportBaseComponent](#)

Objects

[TSMEExportStyle](#)

[TSMEDataFormats](#)

[TSMESendTo](#)

[TSMEStatistic](#)

Types

[TActionAfterExport](#)

[TCharacterSet](#)

[TColumnSource](#)

[TExportFormatTypes](#)

[TGetCellParamsEvent](#)

[TSMEAbout](#)

[TSMEAfterRecordEvent](#)

[TSMEBeforeRecordEvent](#)

[TSMEDateOrder](#)

[TSMEGetFileName](#)

[TSMEGetLanguageStringEvent](#)

[TSMEGetNextSelected](#)

[TSMEGetSelectedCount](#)

[TSMELayout](#)

[TSMEProgress](#)

[TSMEResult](#)

[TSMESection](#)

[TSMESpecificationEvent](#)

[TSMESStyle](#)

[TSMOption](#)

[TSMOptions](#)

[TTableTypeExport](#)

Routines

[ReplaceHTMLSystemChars](#)

[TblName](#)

Constants

AllFormats

arrStyleColor

clMoneyGreen

clSkyBlue

GeneratorVer

See also

[OddColor](#)

See also

[EvenColor](#)

See also

[ExportStyle](#)

TSMEStyle type

[See also](#)

Unit

[ExportDS](#)

The TSMEStyle enumerates the default color schemas of exported data.

Declaration

```
type TSMEStyle = (esNormal, esPriceList, esMSMoney, esBrick,  
esDesert, esEggplant, esLilac, esMaple, esMarine, esRose, esSpruce,  
esWheat);
```

Description

The default schemas allow to export data in two-color mode when odd and even rows have different background colors.

Each enumerated value defines the OddColor and Even Color subproperties of Style property for any export component.

See also

[DataFormats](#)

Properties

► Run-time only ■ Key properties

- [BooleanFalse](#)
- [BooleanTrue](#)
- [CurrencyString](#)
- [CustomDateTimeFormat](#)
- [DateOrder](#)
- [DateSeparator](#)
- [DecimalSeparator](#)
- [FourDigitYear](#)
- [LeadingZerosInDate](#)
- [ThousandSeparator](#)
- [TimeSeparator](#)
- [UseRegionalSettings](#)

Methods

- Key methods

- [GetDateFormat](#)

- [GetDateTimeFormat](#)

- [GetTimeFormat](#)

~~Assign~~{linkDelphi=Assign_Method}

- [LoadRegionalSettings](#)

See also

[DataFormats](#)

[BooleanTrue](#)

See also

[DataFormats](#)

[BooleanFalse](#)

See also

[DataFormats](#)

[DecimalSeparator](#)

[ThousandSeparator](#)

See also

[DataFormats](#)

[DateOrder](#)

[DateSeparator](#)

[FourDigitYear](#)

[LeadingZerosInDate](#)

[TimeSeparator](#)

See also

[DataFormats](#)

[CustomDateTimeFormat](#)

[DateSeparator](#)

[FourDigitYear](#)

[LeadingZerosInDate](#)

[TimeSeparator](#)

TSMEDateOrder type

[See also](#)

Unit

[ExportDS](#)

The TSMEDateOrder enumerates the supported order of date values for exported data.

Declaration

```
type TSMEDateOrder = (doMDY, doDMY, doYMD, doYDM, doDYM, doMYD);
```

Description

The available orders for date values allow to define a custom order for exported data.

By default date values will be formatted using defined regional settings in Control Panel of MS Windows but you can customize it.

See also

[DataFormats](#)

[CustomDateTimeFormat](#)

[DateOrder](#)

[FourDigitYear](#)

[LeadingZerosInDate](#)

[TimeSeparator](#)

See also

[DataFormats](#)

[CurrencyString](#)

[ThousandSeparator](#)

See also

[DataFormats](#)

[CustomDateTimeFormat](#)

[DateOrder](#)

[DateSeparator](#)

[LeadingZerosInDate](#)

[TimeSeparator](#)

See also

[DataFormats](#)

[CustomDateTimeFormat](#)

[DateOrder](#)

[DateSeparator](#)

[FourDigitYear](#)

[TimeSeparator](#)

See also

[DataFormats](#)

[CurrencyString](#)

[DecimalSeparator](#)

See also

[DataFormats](#)

[CustomDateTimeFormat](#)

[DateOrder](#)

[DateSeparator](#)

[FourDigitYear](#)

[LeadingZerosInDate](#)

See also

[DataFormats](#)

[LoadRegionalSettings](#)

See also

[DataFormats](#)

[GetDateTimeFormat](#)

[GetTimeFormat](#)

See also

[DataFormats](#)

[GetDateFormat](#)

[GetTimeFormat](#)

See also

[DataFormats](#)

[GetDateFormat](#)

[GetDateTimeFormat](#)

See also

[DataFormats](#)

[UseRegionalSettings](#)

See also

<<< See also of TSMESendTo object >>>

Properties

► Run-time only ■ Key properties

■ [EMailBCC](#)

■ [EMailBody](#).

■ [EMailCC](#)

■ [EMailOpenBeforeSend](#)

■ [EMailRecipient](#)

■ [EMailSubject](#)

Methods

- Key methods

~~Assign~~{linkDelphi=Assign_Method}

See also

[EMailCC](#)

[EMailRecipient](#)

See also

[EMailSubject](#)

See also

[EMailBCC](#)

[EMailRecipient](#)

See also

[EMailBCC](#)

[EMailCC](#)

See also

[EMailBody](#).

See also

<<< See also of TSMEDataLevel component >>>

Properties

► Run-time only ■ Key properties

- [Bands](#)
- [Caption](#)
- [Columns](#)
- [ColumnSource](#)
- [DataEngine](#)
- [DataSet](#)
- [DBGrid](#)
- [ShiftCount](#)

Methods

- Key methods

~~Create~~{linkDelphi=Create_Method}

~~Destroy~~{linkDelphi=Destroy_Method}

~~Assign~~{linkDelphi=Assign_Method}

- [CreateSourceDataEngine](#)

- [DestroySourceDataEngine](#)

See also

<<< See also of Bands property >>>

Bands property example

See also

<<< See also of Caption property >>>

Caption property example

See also

<<< See also of Columns property >>>

Columns property example

See also

<<< See also of ColumnSource property >>>

ColumnSource property example

See also

<<< See also of DataEngine property >>>

DataEngine property example

See also

<<< See also of DataSet property >>>

DataSet property example

See also

<<< See also of DBGrid property >>>

DBGrid property example

See also

<<< See also of ShiftCount property >>>

ShiftCount property example

See also

<<< See also of CreateSourceDataEngine method >>>

CreateSourceDataEngine method example

See also

<<< See also of DestroySourceDataEngine method >>>

DestroySourceDataEngine method example

See also

<<< See also of TSMEDataLevels component >>>

Properties

▶ Run-time only ■ Key properties

▶ ■ [Items](#)

▶ ■ [SMEControl](#)

Methods

- Key methods
- [Create](#)
- [Add](#)

About the TSMEDataLevels component

[See also TSMEDataLevels reference](#)

Purpose

<<< purpose of the TSMEDataLevels component >>>

Tasks

<<< how to use the TSMEDataLevels component >>>

See also

<<< See also of Items property >>>

Items property example

See also

<<< See also of SMEControl property >>>

SMEControl property example

See also

<<< See also of Create method >>>

Create method example

See also

<<< See also of Add method >>>

Add method example

See also

<<< See also of TSMStatistic object >>>

Properties

► Run-time only ■ Key properties

► ■ [CurrentCol](#)

► ■ [CurrentDataLevel](#)

► ■ [CurrentRow](#)

► ■ [CurrentSection](#)

■ [Result](#)

■ [TotalCount](#)

■ [TotalErrors](#)

■ [UpdateStep](#)

Methods

- Key methods

~~Assign~~{linkDelphi=Assign_Method}

See also

<<< See also of CurrentCol property >>>

CurrentCol property example

See also

<<< See also of CurrentDataLevel property >>>

CurrentDataLevel property example

See also

<<< See also of CurrentRow property >>>

CurrentRow property example

See also

<<< See also of CurrentSection property >>>

CurrentSection property example

See also

<<< See also of Result property >>>

Result property example

See also

<<< See also of TotalCount property >>>

TotalCount property example

See also

<<< See also of TotalErrors property >>>

TotalErrors property example

See also

<<< See also of UpdateStep property >>>

UpdateStep property example

See also

<<< See also of TSMECustomBaseComponent component >>>

Properties

► Run-time only ■ Key properties

■ [About](#)

► ■ [AddTitle](#)

■ [AnimatedStatus](#)

► ■ [AutoFitColumns](#)

■ [BlankIfZero](#)

► ■ [CharacterSet](#)

■ [DataFormats](#)

► ■ [ExportedRecordCount](#)

■ [ExportIfEmpty](#)

► ■ [ExportResult](#)

► ■ [ExportStyle](#)

► ■ [Fixed](#)

■ [KeyGenerator](#)

■ [Options](#)

► ■ [PageSetup](#)

► ■ [RecordSeparator](#)

■ [RightToLeft](#)

■ [SelectedRecord](#)

■ [SendTo](#)

► ■ [Separator](#)

■ [Statistic](#)

► ■ [TableName](#)

► ■ [TextQualifier](#)

Methods

- Key methods

~~Create~~{linkDelphi=Create_Method}

~~Destroy~~{linkDelphi=Destroy_Method}

- [Extension](#)

- [GetTitleRowCount](#)

- [SendMail](#)

- [AboutSME](#)

~~Assign~~{linkDelphi=Assign_Method}

Events

- Key events
- [OnAfterExecute](#)
- [OnAfterRecord](#)
- [OnBeforeExecute](#)
- [OnBeforeRecord](#)
- [OnGetCellParams](#)
- [OnGetLanguageString](#)
- [OnGetFileName](#)
- [OnProgress](#)

TSMEAbout type

Unit

[ExportDS](#)

Declaration

```
type TSMEAbout = string;
```

Description

It's a design-time type for About property.

See also

<<< See also of AutoFitColumns property >>>

AutoFitColumns property example

TCharacterSet type

Unit

[ExportDS](#)

Declaration

```
type TCharacterSet = (csANSI_WINDOWS, csASCII_MSDOS, csEBCDIC);
```

Description

This type specifies the available code pages for data exporting.

DataFormats property example

{mm/dd/yyyy as date format}

SMExportToText1.DataFormats.DateOrder := doMDY;

SMExportToText1.DataFormats.LeadingZerosInDate := True;

SMExportToText1.DataFormats.FourDigitYear := True;

{custom numeric formats}

SMExportToText1.DataFormats.DecimalSeparator := '.';

See also

<<< See also of ExportIfEmpty property >>>

ExportIfEmpty property example

See also

<<< See also of ExportResult property >>>

ExportResult property example

See also

<<< See also of ExportStyle property >>>

ExportStyle property example

See also

<<< See also of Fixed property >>>

See also

<<< See also of KeyGenerator property >>>

TSMEGetFileName type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TSMEGetFileName = procedure(Sender: TObject; CurFileNumber: Integer; var AFileName: string); of object;
```

Description

This procedure type is declared for batch exporting in a few files.

By default the file names will be generated as FileName001.ext, FileName002.ext etc where FileName.ext is a value of your FileName property.

For example, you specified the RowsPerFile=25 (so every 25 rows exported in new file) and using this event you may change the generated file name

See also

<<< See also of OnProgress property >>>

OnProgress property example

See also

<<< See also of Options property >>>

Options property example

See also

<<< See also of PageSetup property >>>

PageSetup property example

See also

<<< See also of RecordSeparator property >>>

RecordSeparator property example

See also

<<< See also of SendTo property >>>

SendTo property example

See also

<<< See also of Separator property >>>

See also

<<< See also of Statistic property >>>

Statistic property example

See also

<<< See also of TableName property >>>

See also

<<< See also of TextQualifier property >>>

TextQualifier property example

See also

<<< See also of Extension method >>>

Extension method example

See also

<<< See also of GetTitleRowCount method >>>

GetTitleRowCount method example

See also

<<< See also of SendMail method >>>

SendMail method example

See also

[OnAfterExecute](#)

See also

<<< See also of OnAfterRecord event >>>

OnAfterRecord event example

See also

[OnAfterExecute](#)

See also

<<< See also of OnBeforeRecord event >>>

OnBeforeRecord event example

See also

<<< See also of OnGetCellParams event >>>

OnGetCellParams event example

For example, in TSMExportToHTML you can export a data from "WWW_link" field as link:

```
procedure TForm1.SMExportToHTML1GetCellParams(Sender: TObject;  
Field: TField; var Text: String; AFont: TFont; var Alignment: TAlignment;  
var Background: TColor);  
begin  
if Assigned(Field) and  
  
(UpperCase(Field.FieldName) = 'WWW_LINK') then  
Text := '<a href="' + Text + '"'>' + Text + '</a>';  
end;
```

TGetCellParamsEvent type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TGetCellParamsEvent = procedure(Sender: TObject; Field:  
TField; var Text: string; AFont: TFont; var Alignment: TAlignment;  
var Background: TColor; var CellType: TCellType); of object;
```

Description

Occurs for each cell export (each field of each record and title).

Write an OnGetCellParams event handler to take specific actions before cell export has ended. Also in this handler you can change the exported data (some encoding or changing of text) and/or change the font/color/alignment values.

OnGetLanguageString event example

case id of

```
0: Result := 'Property %s wasn't specified.';
1: Result := 'Assigned DBGrid (%s) have the not supported type';
2: Result := 'The data export to the %s file successfully is finished.';
3: Result := 'Error of data export.';
4: Result := 'Data export is successfully completed.';
5: Result := 'It is impossible to create a file %s.';
6: Result := 'The Exported data by SMExport generator';
7: Result := 'File %s already exists. Do you want to replace it?';

8: Result := 'Exporting...';
9: Result := 'Can't create a process dialog';
10: Result := 'Setup of the data export';
11: Result := '&OK';
12: Result := '&Cancel';
13: Result := '< &Back';
14: Result := '&Next >';
15: Result := 'Execute';
16: Result := 'Data exporting was canceled by user.';
17: Result := 'Export Wizard';
18: Result := 'Step %d of %d';
19: Result := 'Wizard is not complete. If you quit the Wizard now, the new export will not be
created. Abort wizard?';

20: Result := 'This wizard allows you to specify details on how should export your data.
Which export format would you like?';
21: Result := '';
22: Result := 'What delimiter separates your fields? Select the appropriate delimiter.';
23: Result := 'You can define a custom properties of exported columns.';
24: Result := 'You can add a custom header and footer into generated file.';
25: Result := 'That''s all information the wizard needs to export your data.';

26: Result := 'Header';
27: Result := 'Footer';

28: Result := ' Table type ';
29: Result := 'Paradox file (*.db)';
30: Result := 'DBase file (*.dbf)';
31: Result := 'Text file (*.txt)';
32: Result := 'HTML file (*.htm)';
33: Result := 'Excel spreadsheet (*.xls)';
34: Result := 'Excel file (*.xls)';
35: Result := 'Word file (*.doc)';
36: Result := 'SYLK (Symbolic Link) (*.slk)';
37: Result := 'DIF (Data Interchange Format) (*.dif)';
38: Result := 'Lotus 1-2-3 file (*.wk1)';
39: Result := 'QuattroPro file (*.wq1)';
40: Result := 'SQL script file (*.sql)';
```


41: Result := 'XML file (*.xml)';
42: Result := 'MS Access database (*.mdb)';
43: Result := 'MS Windows clipboard';
44: Result := 'Rich Text format (*.rtf)';

50: Result := 'File Origin:';
51: Result := 'selected records only';
52: Result := 'include a column titles';
53: Result := 'Include Field Names on First Row';
54: Result := 'blank if zero';
55: Result := 'Action after exporting';
56: Result := 'none';
57: Result := 'open for file view';
58: Result := 'e-mail with file attachment';

59: Result := ' Field delimiter ';
60: Result := 'Tab';
61: Result := 'Semicolon';
62: Result := 'Comma';
63: Result := 'Space';
64: Result := 'Other symbol:';
65: Result := 'fixed length of the columns';
66: Result := '&Delimited - Characters such as comma or tab separate each field';
67: Result := 'Fixed &Width - Fields are aligned in columns with spaces between each field';

68: Result := 'Export to File:';
69: Result := 'Select a file for data exporting';
70: Result := 'All files (*.*)|*.*';
71: Result := 'Table name:';

72: Result := ' Title ';
73: Result := ' Data ';
74: Result := 'Caption:';
75: Result := 'Alignment:';
76: Result := 'Background:';
77: Result := 'Font:';
78: Result := 'Width:';
79: Result := 'characters';
80: Result := 'left';
81: Result := 'right';
82: Result := 'center';

83: Result := Specifications...
else
Result := ''
end;

TSMEGetLanguageStringEvent type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TSMEGetLanguageStringEvent = procedure(Sender: TObject; id: Integer; var Text: string); of object;
```

Description

This procedure type is declared for writing of multilingual applications.

See also

<<< See also of TSMExportBaseComponent component >>>

Properties

► Run-time only ■ Key properties

- ■ [ActionAfterExport](#)
- [Bands](#)
- [Columns](#)
- [ColumnSource](#)
- [DataEngine](#)
- [DataSet](#)
- [DBGrid](#)
- ■ [DetailSources](#)
- ■ [FileName](#)
- ■ [LastExportColumnIndex](#)
- [SpecificationDir](#)

Methods

- Key methods

~~Create~~{linkDelphi=Create_Method}

~~Destroy~~{linkDelphi=Destroy_Method}

- [GetDefExt](#)

- [GetDS](#)

~~Assign~~{linkDelphi=Assign_Method}

- [BuildDefaultColumns](#)

- [Execute](#)

- [LoadSpecification](#)

- [LoadSpecificationFromStream](#)

- [SaveSpecification](#)

- [SaveSpecificationToStream](#)

Events

- Key events
- [OnAfterLoadSpecification](#)
- [OnAfterSaveSpecification](#)
- [OnBeforeLoadSpecification](#)
- [OnBeforeSaveSpecification](#)
- [OnGetNextSelected](#)
- [OnGetSelectedCount](#)

TActionAfterExport type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TActionAfterExport = (aeNone, aeOpenView, aeEMail);
```

Description

You can define the default action which will process after export complete.

These are the possible values of ActionAfterExport:

Value Meaning

aeNone you don't want to do anything after export process finish;

aeOpenView to open the target export file in view/edit mode. For example, open MS Excel with exported spreadsheet);

aeEMail to send the exported file by e-mail as attachment.

See also

<<< See also of Bands property >>>

See also

[Bands](#)

[TSMEColumn](#)

[TSMEColumns](#)

TColumnSource type

Unit

[ExportDS](#)

Declaration

```
type TColumnSource = (csDBGrid, csDataSet, csDataEngine);
```

Description

This type specifies the available sources for data exporting.
These are the possible values of ColumnSource:

Value Meaning

csDBGrid export a data from linked DBGrid component and transfer a width, font, alignment for each visible columns (data and title).

csDataSet export a data from linked DataSet component.

caDataEngine export from specified DataEngine component

See also

[ColumnSource](#)

[TSMECustomDataEngine](#)

See also

[ColumnSource](#)

See also

[ColumnSource](#)

See also

<<< See also of DetailSources property >>>

DetailSources property example

See also

[OnGetSelectedCount](#)

OnGetNextSelected property example

For example, you use a third-party grid from InfoPower. This grid uses own specific method of work with selected records. In this case you must write the next code:

```
procedure TForm1.SMEWizardDlg1GetNextSelected(Sender: TObject;  
Index: Integer; var Selected: TBookmark);  
begin  
Selected := wwDBGrid1.SelectedList.Items[Index]  
end;
```

where wwDBGrid1 is name of InfoPower's grid.

TSMEGetNextSelected type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TSMEGetNextSelected = procedure(Sender: TObject; Index: Integer; var Selected: TBookmark); of object;
```

Description

If you use some third-party grid or you want to control a list of exported records in own code, you must use [OnGetSelectedCount](#) and [OnGetNextSelected](#) events.

In this event you must return a pointer on next selected records which must be exported.

See also

[OnGetNextSelected](#)

OnGetSelectedCount property example

For example, you use a third-party grid from InfoPower. This grid uses own specific method of work with selected records. In this case you must write the next code:

```
procedure TForm1.SMEWizardDlg1GetSelectedCount(Sender: TObject; var Count:
LongInt): Integer;
begin
Count := wwDBGrid1.SelectedList.Count
end;
```

where wwDBGrid1 is name of InfoPower's grid.

TSMEGetSelectedCount type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TSMEGetSelectedCount = procedure(Sender: TObject; var Count: LongInt); of object;
```

Description

If you use some third-party grid or you want to control a list of exported records in own code, you must use [OnGetSelectedCount](#) and [OnGetNextSelected](#) events.

In this event you must return a total number of records which must be exported in target file.

See also

[LoadSpecificationFromStream](#)

[LoadSpecification](#)

[SaveSpecification](#)

[SaveSpecificationToStream](#)

SpecificationDir property example

See also

[LoadSpecificationFromStream](#)

[SaveSpecification](#)

[SaveSpecificationToStream](#)

[OnBeforeLoadSpecification](#)

[OnAfterLoadSpecification](#)

See also

<<< See also of GetDefExt method >>>

GetDefExt method example

See also

<<< See also of GetDS method >>>

GetDS method example

See also

<<< See also of Execute method >>>

See also

[LoadSpecification](#)

[SaveSpecification](#)

[SaveSpecificationToStream](#)

[OnBeforeLoadSpecification](#)

[OnAfterLoadSpecification](#)

See also

[LoadSpecificationFromStream](#)

[LoadSpecification](#)

[SaveSpecificationToStream](#)

[OnBeforeSaveSpecification](#)

[OnAfterSaveSpecification](#)

See also

[LoadSpecificationFromStream](#)

[LoadSpecification](#)

[SaveSpecification](#)

[OnBeforeSaveSpecification](#)

[OnAfterSaveSpecification](#)

See also

<<< See also of OnAfterLoadSpecification event >>>

OnAfterLoadSpecification event example

See also

<<< See also of OnAfterSaveSpecification event >>>

OnAfterSaveSpecification event example

See also

<<< See also of OnBeforeLoadSpecification event >>>

OnBeforeLoadSpecification event example

See also

<<< See also of OnBeforeSaveSpecification event >>>

OnBeforeSaveSpecification event example

See also

<<< See also of SMEEngine unit >>>

CellType2FieldType function

Unit

[SMEEngine](#)

Declaration

```
function CellType2FieldType(ct: TCellType): TFieldType;
```

Description

This function converts the [TCellType](#) to TFieldType

GetValueType function

Unit

[SMEEngine](#)

Declaration

```
function GetValueType(DataType: TFieldType; Value: Variant;  
BlankIfZero: Boolean): TCellType;
```

Description

This function converts the TFieldType to [TCellType](#)

See also

[TSMEColumn.ColumnKind](#)

See also

[DataType](#)

[CellType2FieldType](#)

[GetValueType](#)

See also

[TSMEColumn.GetItemCaption](#)

See also

[Measure](#)

See also

[Orientation](#)

See also

[OnCount](#)

[RecordCount](#)

See also

[OnGetValue](#)

[GetFieldValue](#)

See also

[OnNext](#)

[Next](#)

TExportFormatTypes type

Unit

[ExportDS](#)

Declaration

```
type TExportFormatTypes = set of TTableTypeExport ;
```

Description

This type is list of available export formats.

This type is used when in the compound export components you can limit a list of available formats for end-users.

For example, if on client computers is not installed BDE, you can hide a Paradox and DBase formats which requires a BDE.

TSMEAAfterRecordEvent type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TSMEAAfterRecordEvent = procedure(Sender: TObject; var Abort: Boolean); of object;
```

Description

<<< Description of TSMEAAfterRecordEvent type >>>

TSMEBeforeRecordEvent type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TSMEBeforeRecordEvent = procedure(Sender: TObject; var Accept: Boolean); of object;
```

Description

<<< Description of TSMEBeforeRecordEvent type >>>

TSMELayout type

[See also](#)

Unit

[ExportDS](#)

The TSMELayout enumerates the possible layouts of forms for exported data.

Declaration

```
type TSMELayout = (elColumnar, elReversedColumnar, elTabularForm);
```

Description

The available layout for exported data allow to customize the external view of data.

elColumnar will generate a "table" with standard list of columns/rows

elReversedColumnar is a similar to elColumnar layout but with rotated view when rows are columns and columns are rows

elTabularForm will generate a few rows per each record and each row will contain a field name and field value

Default value is elColumnar.

Note that custom layout can be used for non-table formats like html, MS Excel, text etc but can't be applied to Paradox, MS Access etc

TSMEProgress type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TSMEProgress = procedure(Sender: TObject; CurValue, MaxValue: Integer; var Abort: Boolean); of object;
```

Description

<<< Description of TSMEProgress type >>>

TSMEResult type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TSMEResult = (erInProgress, erCompleted, erCanceled,  
erFailed);
```

Description

<<< Description of TSMEResult type >>>

TSMESection type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TSMESection = (esHeader, esBand, esTitle, esData, esFooter,  
esDetail);
```

Description

<<< Description of TSMESection type >>>

TSMESpecificationEvent type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TSMESpecificationEvent = procedure(Sender: TObject; var AFileName: string); of object;
```

Description

This type is used for event of specification loading in visual export dialogs.

TSMOption type

[See also](#)

Unit

[ExportDS](#)

Declaration

```
type TSMOption = (soFieldMask, soShowMessage,  
soBlankRowAfterCaptions, soMergeData, soWaitCursor,  
soDisableControls, soUseFieldNameAsCaption, soColLines, soRowLines,  
soColorsFonts, soExportBlankValues, soExportBands);
```

Description

This enumerated type is a list of available extended options.

soFieldMask - if option is included, the field value will be exported as string with applied DisplayFormat. Else will be exported a "pure" data.

soShowMessage - if included, during export process the some messages are available (with error messages, for example). Else the export will be in "silence" mode. This option is very useful for CGI application which must be executed on server side without any interaction with end-user.

soBlankRowAfterCaptions - if included, then after exported row with captions of fields, will be added a blank row before exported data rows. Very useful option for text or MS Excel exporting.

Note: will be affected only if AddTitle property is True and export format is not "table" format like Paradox, MS Access etc

soMergeData - if included and file is exists (see FileName property topic), then data will be merged with existing data (added at bottom of file or created in new sheet for MS Excel spreadsheet)

soWaitCursor - disable this option if your application is server-side (cgi, ISAPI, ASP etc)

soDisableControls - to increase the speed for export process you may disable repainting during dataset navigation.

soUseFieldNameAsCaption - to display in wizard the captions for fields instead field

names

soColLines - to export the vertical lines between columns

soRowLines - to export the horizontal lines between rows

soColorsFonts - exclude this option if you want to export the plain data values. Disabled option will crease the performance

soExportBlankValues - to export the empty values from fields or export the non-empty values only

soExportBands - export the column groupings or don't include the bands as first header line (before column headers)

TSMOptions type

Unit

[ExportDS](#)

Declaration

```
type TSMOptions = set of TSMOption;
```

Description

This type enumerate the available extended options.

TTableTypeExport type

Unit

[ExportDS](#)

TTableTypeExport values are available export formats which are supported by SMExport suite.

Declaration

```
type TTableTypeExport = (teParadox, teDBase, teText, teHTML, teXLS, teExcel, teWord, teSYLK, teDIF, teWKS, teQuattro, teSQL, teXML, teAccess, teClipboard, teRTF, teSPSS, tePDF, teLDIF, teADO);
```

Description

TTableTypeExport is a set of values which are passed to export components. SMExport suite contains a lot of different export component which supports each format. Also there you can find compound components which takes any formats in one place.

ReplaceHTMLSystemChars function

Unit

[ExportDS](#)

Declaration

```
function ReplaceHTMLSystemChars(const s: string): string;
```

Description

This useful function will convert the source string into correct formatted html-string

TblName function

Unit

[ExportDS](#)

Declaration

```
function TblName(DS: TDataset): string;
```

Description

This useful function will return the valid TableName from source dataset.

If dataset is T*Table component (TTable, TADOTable etc) then actual value for TableName property.

Else if dataset is query component, then SQL-statement will be parsed (SQL-property) to get the table name.

See also

[ExportStyle](#)

See also

[TSMEDataFormats](#)

See also

<<< See also of About the TSMEDataLevels component >>>

See also

[OnGetFileName](#)

See also

[OnGetCellParams](#)

See also

[OnGetLanguageString](#)

See also

[ActionAfterExport](#)

See also

[TSMEGetSelectedCount](#)

See also

[TSMEGetNextSelected](#)

See also

<<< See also of TSMEAfterRecordEvent type >>>

See also

<<< See also of TSMEBeforeRecordEvent type >>>

See also

Layout

See also

<<< See also of TSMEProgress type >>>

See also

<<< See also of TSMEResult type >>>

See also

<<< See also of TSMESession type >>>

See also

TOnGetSpecifications

See also

[Options](#)

[TSMOptions](#)